Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 120 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
- Non-Volatile Program and Data Memories
  - 2/4/8K Bytes of In-System Programmable Program Memory Flash
    - Endurance: 10,000 Write/Erase Cycles
  - 128/256/512 Bytes of In-System Programmable EEPROM
    - Endurance: 100,000 Write/Erase Cycles
  - 128/256/512 Bytes of Internal SRAM
  - Data retention: 20 years at 85°C / 100 years at 25°C
  - Programming Lock for Self-Programming Flash & EEPROM Data Security
- Peripheral Features
  - One 8-Bit and One 16-Bit Timer/Counter with Two PWM Channels, Each
  - 10-bit ADC
    - 8 Single-Ended Channels
    - 12 Differential ADC Channel Pairs with Programmable Gain (1x / 20x)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-Chip Analog Comparator
  - Universal Serial Interface
- Special Microcontroller Features
  - debugWIRE On-chip Debug System
  - In-System Programmable via SPI Port
  - Internal and External Interrupt Sources: Pin Change Interrupt on 12 Pins
  - Low Power Idle, ADC Noise Reduction, Standby and Power-Down Modes
  - Enhanced Power-on Reset Circuit
  - Programmable Brown-Out Detection Circuit
  - Internal Calibrated Oscillator
  - On-Chip Temperature Sensor
- I/O and Packages
  - Available in 20-Pin QFN/MLF & 14-Pin SOIC and PDIP
  - Twelve Programmable I/O Lines
- Operating Voltage:
  - 1.8 – 5.5V for ATtiny24V/44V/84V
  - 2.7 – 5.5V for ATtiny24/44/84
- Speed Grade
  - ATtiny24V/44V/84V
    - 0 – 4 MHz @ 1.8 – 5.5V
    - 0 – 10 MHz @ 2.7 – 5.5V
  - ATtiny24/44/84
    - 0 – 10 MHz @ 2.7 – 5.5V
    - 0 – 20 MHz @ 4.5 – 5.5V
- Industrial Temperature Range: -40°C to +85°C
- Low Power Consumption
  - Active Mode (1 MHz System Clock): 300 µA @ 1.8V
  - Power-Down Mode: 0.1 µA @ 1.8V

8-bit AVR® Microcontroller with 2/4/8K Bytes In-System Programmable Flash

ATtiny24/44/84

Preliminary

ATtiny24/44 not recommended for new designs. Use:

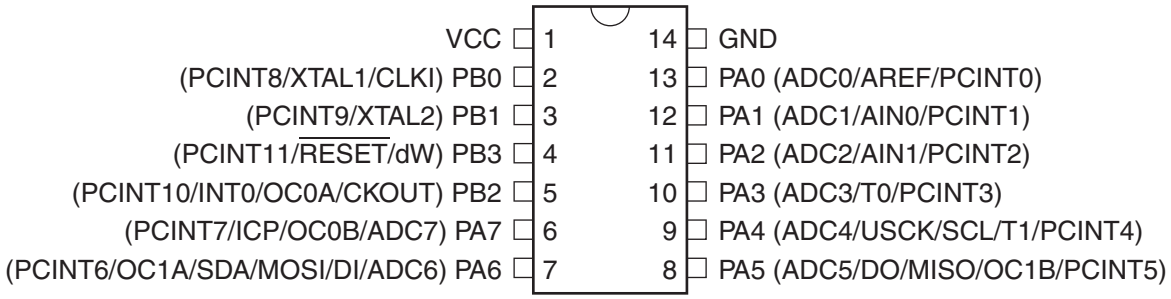
ATtiny24A
ATtiny44A

Rev. 8006H–AVR–10/09

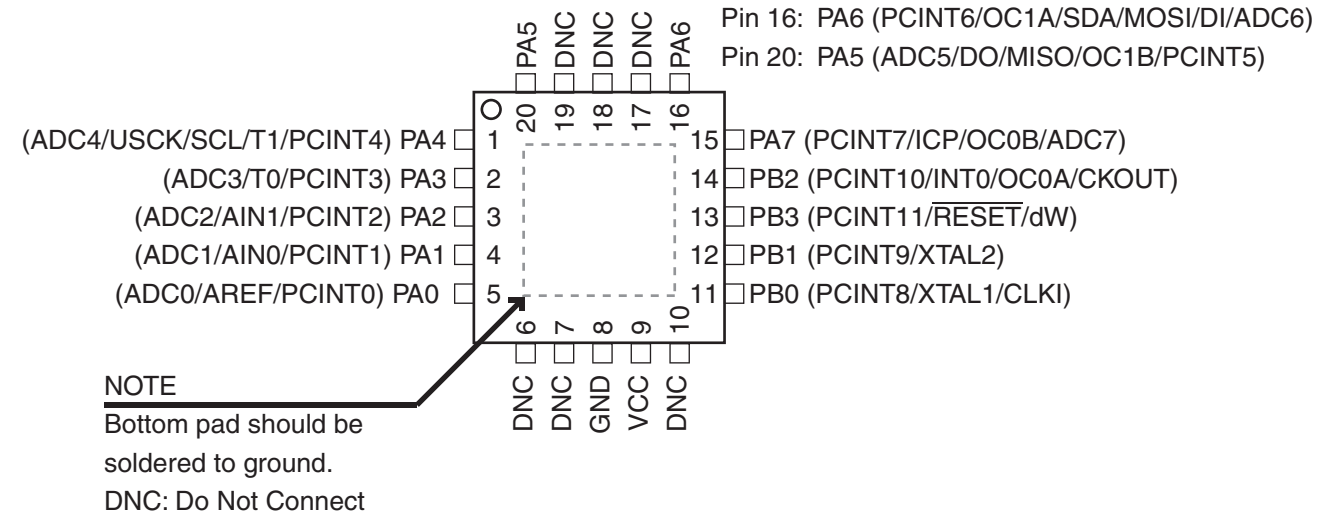
1. Pin Configurations

Figure 1-1. Pinout ATtiny24/44/84

PDIP/SOIC



QFN/MLF



1.1 Pin Descriptions

1.1.1 VCC

Supply voltage.

1.1.2 GND

Ground.

1.1.3 Port B (PB3...PB0)

Port B is a 4-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability except PB3 which has the $\overline{\text{RESET}}$ capability. To use pin PB3 as an I/O pin, instead of RESET pin, program (‘0’) RSTDISBL fuse. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATtiny24/44/84 as listed in Section 10.2 “Alternate Port Functions” on page 58.

1.1.4 $\overline{\text{RESET}}$

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running and provided the reset pin has not been disabled. The minimum pulse length is given in Table 20-4 on page 177. Shorter pulses are not guaranteed to generate a reset.

The reset pin can also be used as a (weak) I/O pin.

1.1.5 Port A (PA7...PA0)

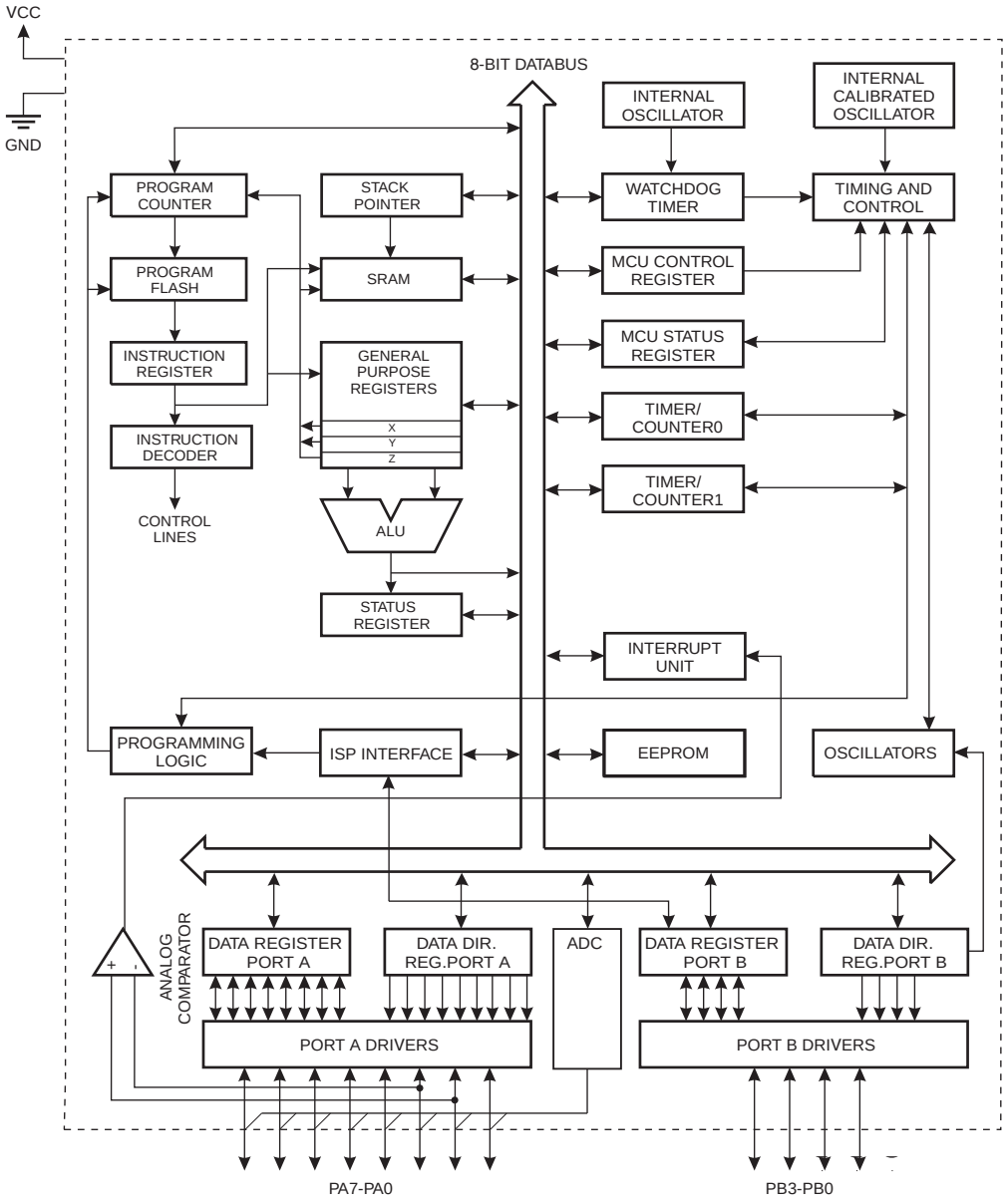
Port A is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A has alternate functions as analog inputs for the ADC, analog comparator, timer/counter, SPI and pin change interrupt as described in “Alternate Port Functions” on page 58.

2. Overview

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

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATtiny24/44/84 provides the following features: 2/4/8K byte of In-System Programmable Flash, 128/256/512 bytes EEPROM, 128/256/512 bytes SRAM, 12 general purpose I/O lines, 32 general purpose working registers, an 8-bit Timer/Counter with two PWM channels, a 16-bit timer/counter with two PWM channels, Internal and External Interrupts, a 8-channel 10-bit ADC, programmable gain stage (1x, 20x) for 12 differential ADC channel pairs, a programmable Watchdog Timer with internal oscillator, internal calibrated oscillator, and four software selectable power saving modes. Idle mode stops the CPU while allowing the SRAM, Timer/Counter, ADC, Analog Comparator, and Interrupt system to continue functioning. ADC Noise Reduction mode minimizes switching noise during ADC conversions by stopping the CPU and all I/O modules except the ADC. In Power-down mode registers keep their contents and all chip functions are disbaled until the next interrupt or hardware reset. In Standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping, allowing very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The on-chip ISP Flash allows the Program memory to be re-programmed in-system through an SPI serial interface, by a conventional non-volatile memory programmer or by an on-chip boot code running on the AVR core.

The ATtiny24/44/84 AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators and Evaluation kits.

3. About

3.1 Resources

A comprehensive set of drivers, application notes, data sheets and descriptions on development tools are available for download at http://www.atmel.com/avr.

3.2 Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

For I/O Registers located in the extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically, this means “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”. Note that not all AVR devices include an extended I/O map.

3.3 Data Retention

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

3.4 Disclaimer

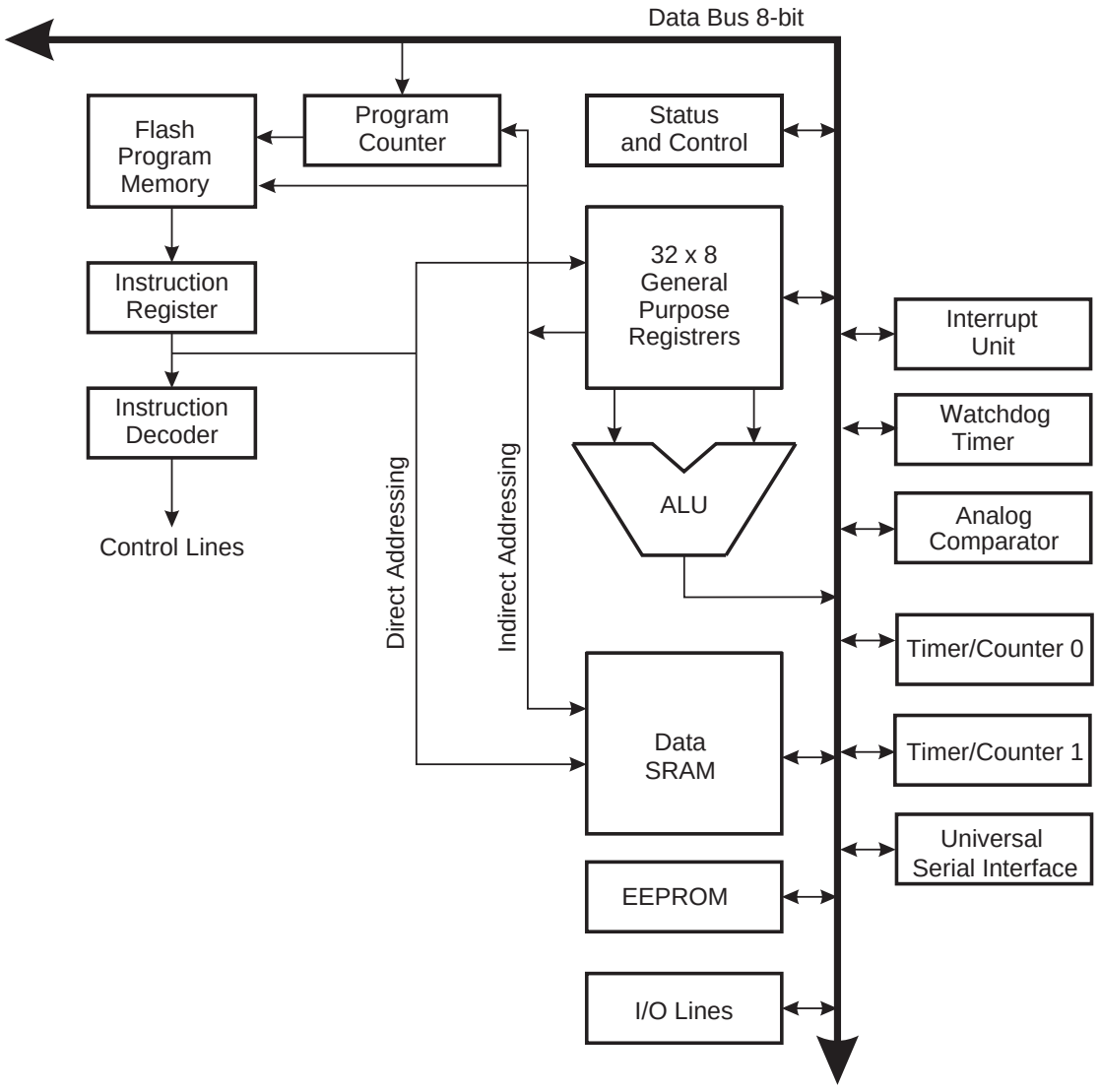
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

4. CPU Core

This section discusses the 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.

4.1 Architectural Overview

Figure 4-1. 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.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, capable of directly addressing the whole address space. Most AVR instructions have a single 16-bit word format but 32-bit wide instructions also exist. The actual instruction set varies, as some devices only implement a part of the instruction set.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F.

4.2 ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

4.3 Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is neither automatically stored when entering an interrupt routine, nor restored when returning from an interrupt. This must be handled by software.

4.3.1 SREG – AVR Status Register

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
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 – I: Global Interrupt Enable**

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

- **Bit 6 – T: Bit Copy Storage**

The Bit Copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the Register File can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half Carry is useful in BCD arithmetic. See the “Instruction Set Description” for detailed information.

- **Bit 4 – S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the Negative Flag N and the Two’s Complement Overflow Flag V. See the “Instruction Set Description” for detailed information.

- **Bit 3 – V: Two’s Complement Overflow Flag**

The Two’s Complement Overflow Flag V supports two’s complement arithmetics. See the “Instruction Set Description” for detailed information.

- **Bit 2 – N: Negative Flag**

The Negative Flag N indicates a negative result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 1 – Z: Zero Flag**

The Zero Flag Z indicates a zero result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 0 – C: Carry Flag**

The Carry Flag C indicates a carry in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

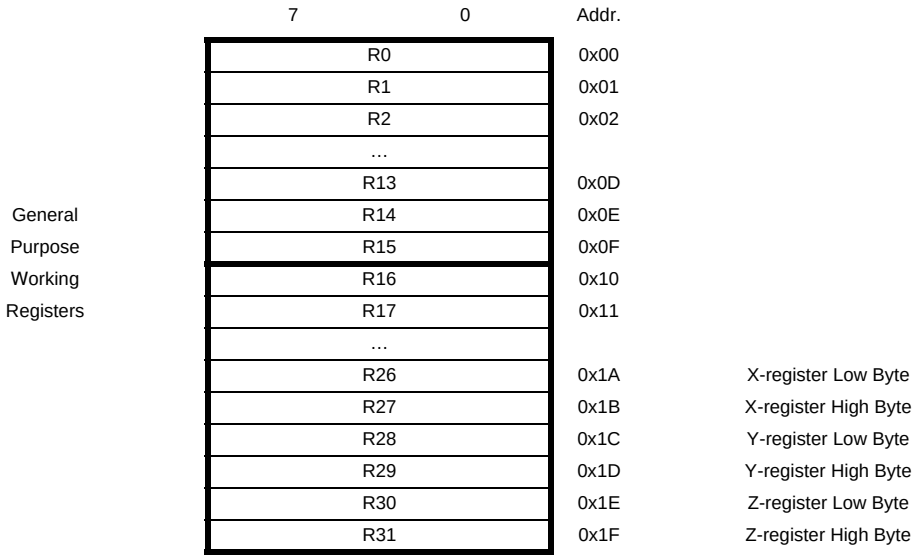
4.4 General Purpose Register File

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

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

Figure 4-2 below shows the structure of the 32 general purpose working registers in the CPU.

Figure 4-2. AVR CPU General Purpose Working Registers



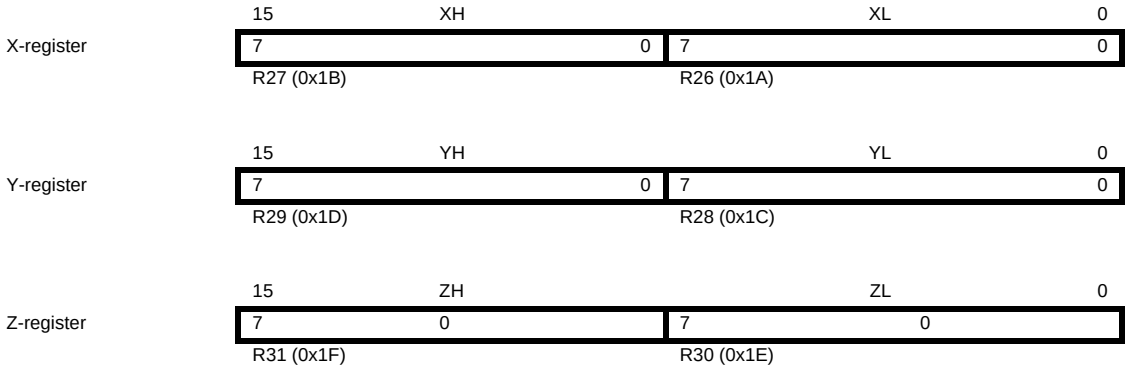
Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 4-2, each register is also assigned a Data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y- and Z-pointer registers can be set to index any register in the file.

4.4.1 The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 4-3 below.

Figure 4-3. The X-, Y-, and Z-registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the instruction set reference for details).

4.5 Stack Pointer

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. The Stack Pointer Register always points to the top of the Stack. Note that the Stack is implemented as growing from higher memory locations to lower memory locations. This implies that a Stack PUSH command decreases the Stack Pointer.

The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The Stack Pointer must be set to point above 0x60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when the return address is pushed onto the Stack with subroutine call or interrupt. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction, and it is incremented by two when data is popped from the Stack with return from subroutine RET or return from interrupt RETI.

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.

4.5.1 SPH and SPL — Stack Pointer Register

Bit	15	14	13	12	11	10	9	8	
0x3E (0x5E)	**SP15**	**SP14**	**SP13**	**SP12**	**SP11**	**SP10**	**SP9**	**SP8**	**SPH**
0x3D (0x5D)	**SP7**	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**	**SPL**
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	
Initial Value	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	

4.6 Instruction Execution Timing

This section describes the general access timing concepts for instruction execution. The AVR CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

Figure 4-4 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast access Register File concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 4-4. The Parallel Instruction Fetches and Instruction Executions

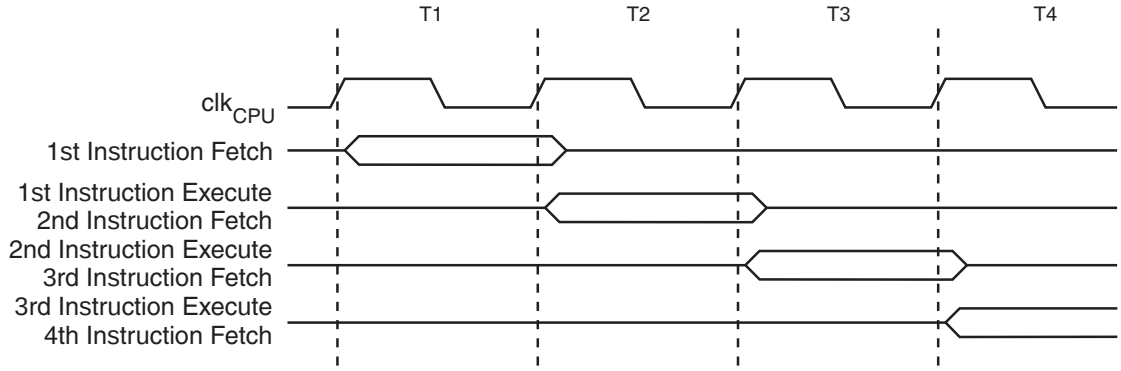
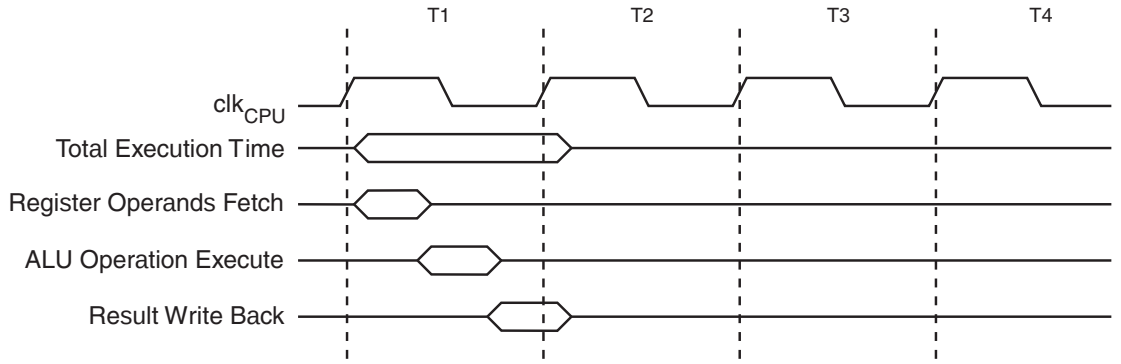


Figure 4-5 shows the internal timing concept for the Register File. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 4-5. Single Cycle ALU Operation



4.7 Reset and Interrupt Handling

The AVR provides several different interrupt sources. These interrupts and the separate Reset Vector each have a separate Program Vector in the Program memory space. All interrupts are assigned individual enable bits which must be written logic one together with the Global Interrupt Enable bit in the Status Register in order to enable the interrupt.

The lowest addresses in the Program memory space are by default defined as the Reset and Interrupt Vectors. The complete list of vectors is shown in “Interrupts” on page 48. The list also determines the priority levels of the different interrupts. The lower the address the higher is the

priority level. RESET has the highest priority, and next is INT0 – the External Interrupt Request 0.

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared and all interrupts are disabled. The user software can write logic one to the I-bit to enable nested interrupts. All enabled interrupts can then interrupt the current interrupt routine. The I-bit is automatically set when a Return from Interrupt instruction – RETI – is executed.

There are basically two types of interrupts. The first type is triggered by an event that sets the Interrupt Flag. For these interrupts, the Program Counter is vectored to the actual Interrupt Vector in order to execute the interrupt handling routine, and hardware clears the corresponding Interrupt Flag. Interrupt Flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared. If an interrupt condition occurs while the corresponding interrupt enable bit is cleared, the Interrupt Flag will be set and remembered until the interrupt is enabled, or the flag is cleared by software. Similarly, if one or more interrupt conditions occur while the Global Interrupt Enable bit is cleared, the corresponding Interrupt Flag(s) will be set and remembered until the Global Interrupt Enable bit is set, and will then be executed by order of priority.

The second type of interrupts will trigger as long as the interrupt condition is present. These interrupts do not necessarily have Interrupt Flags. If the interrupt condition disappears before the interrupt is enabled, the interrupt will not be triggered.

When the AVR exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

Note that the Status Register is not automatically stored when entering an interrupt routine, nor restored when returning from an interrupt routine. This must be handled by software.

When using the CLI instruction to disable interrupts, the interrupts will be immediately disabled. No interrupt will be executed after the CLI instruction, even if it occurs simultaneously with the CLI instruction. The following example shows how this can be used to avoid interrupts during the timed EEPROM write sequence.

Assembly Code Example

```
in  r16, SREG     ; store SREG value
cli    ; disable interrupts during timed sequence
sbi EECR, EEMPE   ; start EEPROM write
sbi EECR, EEPE
out SREG, r16     ; restore SREG value (I-bit)
```

C Code Example

```
char cSREG;
cSREG = SREG; /* store SREG value */
/* disable interrupts during timed sequence */
_CLI();
EECR |= (1<<EEMPE); /* start EEPROM write */
EECR |= (1<<EEPE);
SREG = cSREG; /* restore SREG value (I-bit) */
```

Note: See “Code Examples” on page 6.

When using the SEI instruction to enable interrupts, the instruction following SEI will be executed before any pending interrupts, as shown in the following example.

Assembly Code Example

```
sei  ; set Global Interrupt Enable
sleep; enter sleep, waiting for interrupt
; note: will enter sleep before any pending
; interrupt(s)
```

C Code Example

```
_SEI(); /* set Global Interrupt Enable */
_SLEEP(); /* enter sleep, waiting for interrupt */
/* note: will enter sleep before any pending interrupt(s) */
```

Note: See “Code Examples” on page 6.

4.7.1 Interrupt Response Time

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles the Program Vector address for the actual interrupt handling routine is executed. During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.

A return from an interrupt handling routine takes four clock cycles. During these four clock cycles, the Program Counter (two bytes) is popped back from the Stack, the Stack Pointer is incremented by two, and the I-bit in SREG is set.

5. Memories

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

5.1 In-System Re-programmable Flash Program Memory

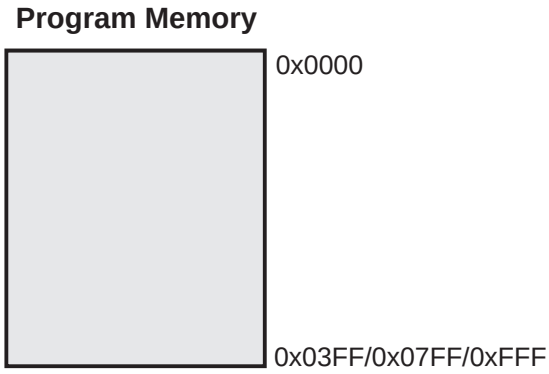
The ATtiny24/44/84 contains 2/4/8K byte On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 1024/2048/4096 x 16.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATtiny24/44/84 Program Counter (PC) is 10/11/12 bits wide, thus addressing the 1024/2048/4096 Program memory locations. “Memory Programming” on page 159 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 instructions LPM – Load Program Memory and SPM – Store Program Memory).

Timing diagrams for instruction fetch and execution are presented in “Instruction Execution Timing” on page 12.

Figure 5-1. Program Memory Map



5.2 SRAM Data Memory

Figure 5-2 on page 16 shows how the ATtiny24/44/84 SRAM Memory is organized.

The lower data memory locations address both 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 cover: 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.

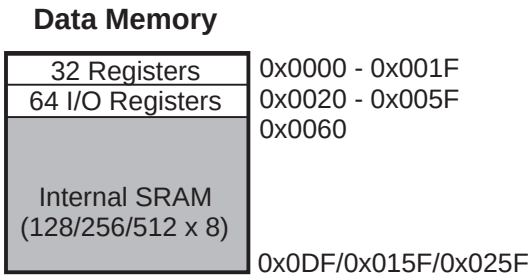
The direct addressing reaches the entire data space.

The Indirect with Displacement mode reaches 63 address locations 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 the 128/256/512 bytes of internal data SRAM in the ATtiny24/44/84 are all accessible through all these addressing modes. The Register File is described in “General Purpose Register File” on page 10.

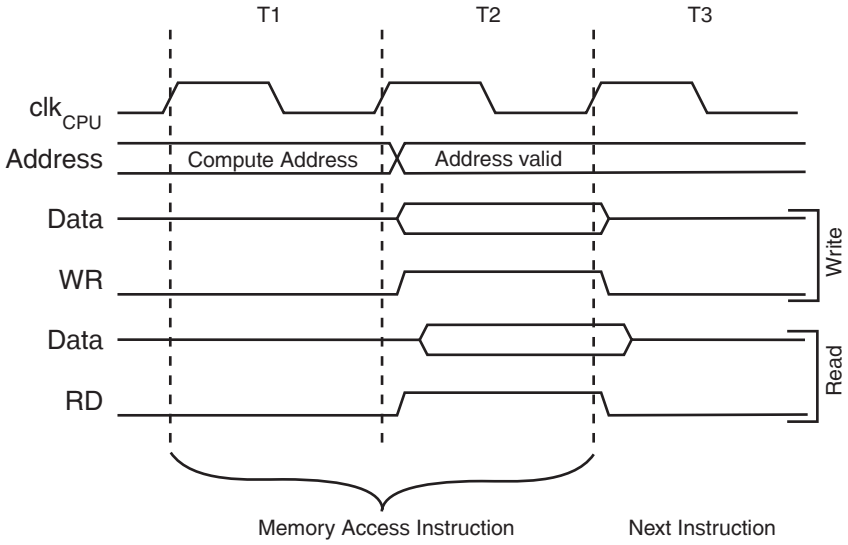
Figure 5-2. Data Memory Map



5.2.1 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 illustrated in Figure 5-3.

Figure 5-3. On-chip Data SRAM Access Cycles



5.3 EEPROM Data Memory

The ATtiny24/44/84 contains 128/256/512 bytes of data EEPROM 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, 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 “Serial Programming” on page 163.

5.3.1 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 5-1 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 the EEPROM, some precautions must be taken. In heavily filtered power supplies, V_{CC} 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 specified as minimum for the clock frequency used. See "Preventing EEPROM Corruption" on page 19 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 "Atomic Byte Programming" on page 17 and "Split Byte Programming" on page 17 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.

5.3.2 Atomic Byte Programming

Using Atomic Byte Programming is the simplest mode. When writing a byte to the EEPROM, the user must write the address into register EEAR and data into register EEDR. If the EEPMn bits are zero, writing EEPE (within four cycles after EEMPE is written) will trigger the erase/write operation. Both the erase and write cycle are done in one operation and the total programming time is given in Table 5-1 on page 22. The EEPE bit remains set until the erase and write operations are completed. While the device is busy with programming, it is not possible to do any other EEPROM operations.

5.3.3 Split Byte Programming

It is possible to split the erase and write cycle in two different operations. This may be useful if the system requires short access time for some limited period of time (typically if the power supply voltage falls). In order to take advantage of this method, it is required that the locations to be written have been erased before the write operation. But since the erase and write operations are split, it is possible to do the erase operations when the system allows doing time-critical operations (typically after Power-up).

5.3.4 Erase

To erase a byte, the address must be written to EEAR. If the EEPMn bits are 0b01, writing the EEPE (within four cycles after EEMPE is written) will trigger the erase operation only (programming time is given in Table 5-1 on page 22). The EEPE bit remains set until the erase operation completes. While the device is busy programming, it is not possible to do any other EEPROM operations.

5.3.5 Write

To write a location, the user must write the address into EEAR and the data into EEDR. If the EEPMn bits are 0b10, writing the EEPE (within four cycles after EEMPE is written) will trigger the write operation only (programming time is given in Table 5-1 on page 22). The EEPE bit remains set until the write operation completes. If the location to be written has not been erased before write, the data that is stored must be considered as lost. While the device is busy with programming, it is not possible to do any other EEPROM operations.

The calibrated Oscillator is used to time the EEPROM accesses. Make sure the Oscillator frequency is within the requirements described in "OSCCAL – Oscillator Calibration Register" on page 30.

5.3.6 Program Examples

The following code examples show one assembly and one C function for erase, write, or atomic write of the EEPROM. The examples assume that interrupts are controlled (e.g., by disabling interrupts globally) so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_write:
  ; Wait for completion of previous write
  sbic EECR, EEPE
  rjmp EEPROM_write
  ; Set Programming mode
  ldi  r16, (0<<EEPM1)|(0<<EEPM0)
  out  EECR, r16
  ; Set up address (r18:r17) in address registers
  out EEARH, r18
  out EEARL, r17
  ; Write data (r19) to data register
  out EEDR, r19
  ; Write logical one to EEMPE
  sbi EECR, EEMPE
  ; Start eeprom write by setting EEPE
  sbi EECR, EEPE
  ret
```

C Code Example

```
void EEPROM_write(unsigned int ucAddress, unsigned char ucData)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE))
    ;
  /* Set Programming mode */
  EECR = (0<<EEPM1)|(0<<EEPM0)
  /* Set up address and data registers */
  EEAR = ucAddress;
  EEDR = ucData;
  /* Write logical one to EEMPE */
  EECR |= (1<<EEMPE);
  /* Start eeprom write by setting EEPE */
  EECR |= (1<<EEPE);
}
```

Note: See "Code Examples" on page 6.

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 (r18:r17) in address registers
  out  EEARH, r18
  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 int ucAddress)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE))
    ;
  /* Set up address register */
  EEAR = ucAddress;
  /* Start eeprom read by writing EERE */
  EECR |= (1<<EERE);
  /* Return data from data register */
  return EEDR;
}
```

Note: See “Code Examples” on page 6.

5.3.7 Preventing EEPROM Corruption

During periods of low V_{CC}, the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. These issues are the same as for board level systems using EEPROM, and the same design solutions should be applied.

An EEPROM data corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the EEPROM requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage is too low.

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 that the power supply voltage is sufficient.

5.4 I/O Memory

The I/O space definition of the ATtiny24/44/84 is shown in “Register Summary” on page 213.

All 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 section 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 to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the Status Flags are cleared by writing a logical one to them. Note that 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.

5.4.1 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.

5.5 Register Description

5.5.1 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/0	

- **Bits 7..1 – Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bit 0 – EEAR8: EEPROM Address**

This is the most significant EEPROM address bit of ATtiny84. In devices with less EEPROM, i.e. ATtiny24/ATtiny44, this bit is reserved and will always read zero. The initial value of the EEPROM Address Register (EEAR) is undefined and a proper value must therefore be written before the EEPROM is accessed.

5.5.2 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	

- **Bit 7 – EEAR7: EEPROM Address**

This is the most significant EEPROM address bit of ATtiny44. In devices with less EEPROM, i.e. ATtiny24, this bit is reserved and will always read zero. The initial value of the EEPROM Address Register (EEAR) is undefined and a proper value must therefore be written before the EEPROM is accessed.

- **Bits 6..0 – EEAR6..0: EEPROM Address**

These are the (low) bits of the EEPROM Address Register. The EEPROM data bytes are addressed linearly in the range 0...128/256/512. The initial value of EEAR is undefined and a proper value must be therefore be written before the EEPROM may be accessed.

5.5.3 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.

5.5.4 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 ATtiny24/44/84. For compatibility with future AVR devices, always write this bit to zero. After reading, mask out this bit.

- **Bit 6 – Res: Reserved Bit**

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

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

The EEPROM Programming mode bits setting defines which programming action that 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 to split the Erase and Write operations in two different operations. The Programming times for the different modes are shown in Table 5-1.

Table 5-1. EEPROM Programming Mode Bits and Programming Times

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

When 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.

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

Writing EERIE to one enables the EEPROM Ready Interrupt if the I-bit in SREG is set. Writing EERIE to 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 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 zero, setting EEPE will have no effect. When EEMPE has been written to one by software, hardware clears the bit to zero after four clock cycles.

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

The EEPROM Program Enable Signal 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 one before a logical one is written to EEPE, otherwise no EEPROM write takes 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 to the EEPROM. When the correct address is set up in the EEAR Register, the EERE bit must be written to one to trigger the EEPROM read. The EEPROM read access takes one instruction, and the requested data is 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 neither possible to read the EEPROM, nor to change the EEAR Register.

5.5.5 GPIOR2 – General Purpose I/O Register 2

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	MSB							LSB	GPIOR2
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	

5.5.6 GPIOR1 – General Purpose I/O Register 1

Bit	7	6	5	4	3	2	1	0	
0x14 (0x34)	MSB							LSB	GPIOR1
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	

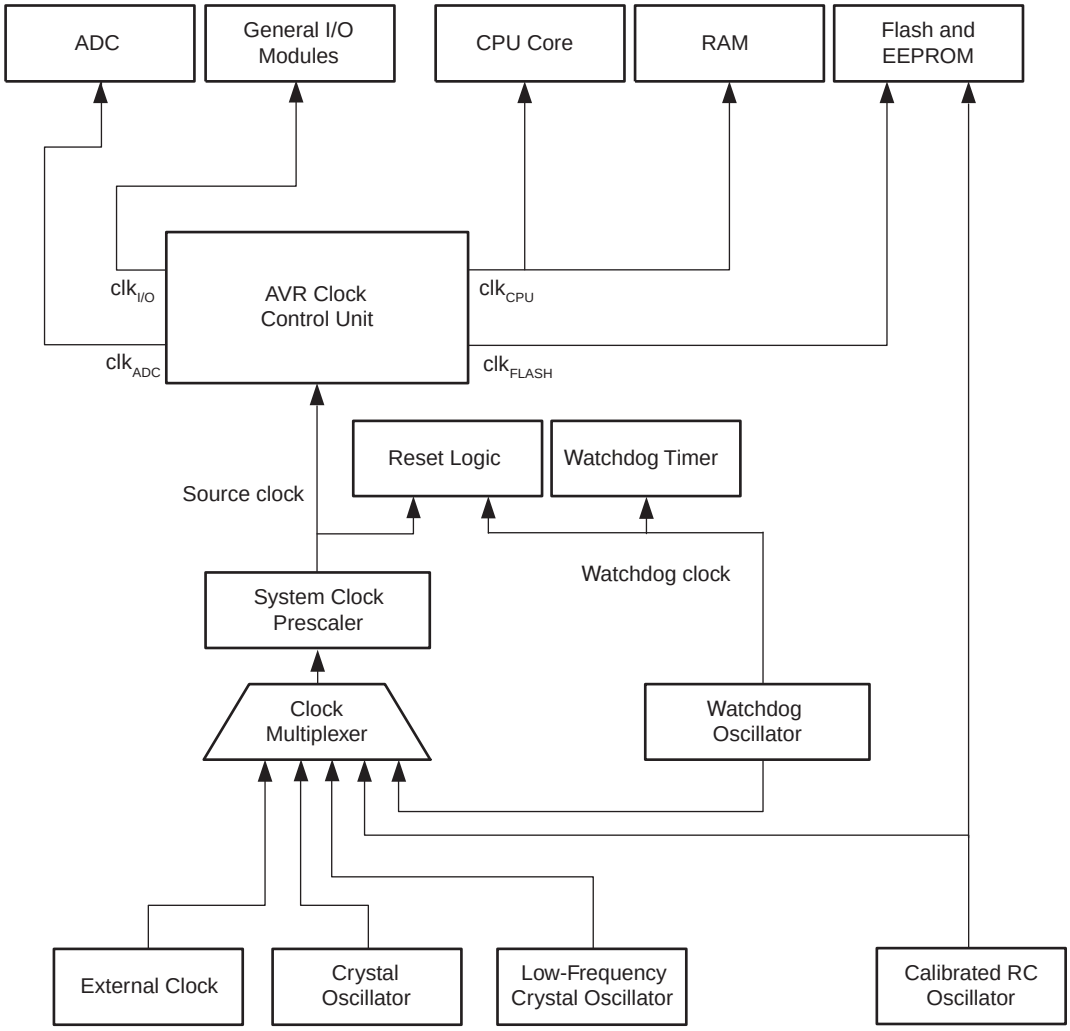
5.5.7 GPIOR0 – General Purpose I/O Register 0

Bit	7	6	5	4	3	2	1	0	
0x13 (0x33)	MSB							LSB	GPIOR0
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	

6. Clock System

Figure 6-1 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in "Power Management and Sleep Modes" on page 33.

Figure 6-1. Clock Distribution



6.1 Clock Subsystems

The clock subsystems are detailed in the sections below.

6.1.1 CPU Clock – clk_{CPU}

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

6.1.2 I/O Clock – $clk_{I/O}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counter. The I/O clock is also used by the External Interrupt module, but note that some external interrupts are detected by asynchronous logic, allowing such interrupts to be detected even if the I/O clock is halted.

6.1.3 Flash Clock – clk_{FLASH}

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

6.1.4 ADC Clock – clk_{ADC}

The ADC is provided with a dedicated clock domain. This allows halting the CPU and I/O clocks in order to reduce noise generated by digital circuitry. This gives more accurate ADC conversion results.

6.2 Clock Sources

The device has the following clock source options, selectable by Flash Fuse bits as shown below. The clock from the selected source is input to the AVR clock generator, and routed to the appropriate modules.

Table 6-1. Device Clocking Options Select

Device Clocking Option	CKSEL3:0(1)
External Clock (see page 26)	0000
Reserved	0001
Calibrated Internal 8 MHz Oscillator (see page 26)	0010
Reserved	0011
Internal 128 kHz Oscillator (see page 27)	0100
Reserved	0101
Low-Frequency Crystal Oscillator (see page 28)	0110
Reserved	0111
Crystal Oscillator / Ceramic Resonator (see page 28)	1000-1111

Note: 1. For all fuses “1” means unprogrammed and “0” means programmed.

The various choices for each clocking option is given in the following sections. When the CPU wakes up from Power-down the selected clock source is used to time the start-up, ensuring stable Oscillator operation before instruction execution starts. When the CPU starts from reset, there is an additional delay allowing the power to reach a stable level before commencing normal operation. The Watchdog Oscillator is used for timing this real-time part of the start-up time. The number of WDT Oscillator cycles used for each time-out is shown in Table 6-2.

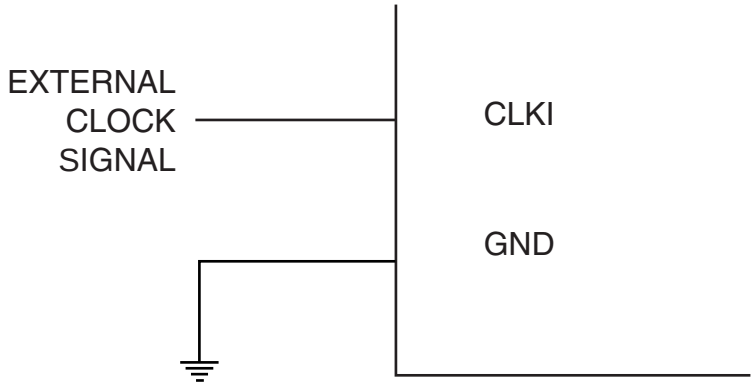
Table 6-2. Number of Watchdog Oscillator Cycles

Typ Time-out	Number of Cycles
4 ms	512
64 ms	8K (8,192)

6.2.1 External Clock

To drive the device from an external clock source, CLKI should be driven as shown in Figure 6-2 on page 26. To run the device on an external clock, the CKSEL Fuses must be programmed to "0000".

Figure 6-2. External Clock Drive Configuration



When this clock source is selected, start-up times are determined by the SUT Fuses as shown in Table 6-3.

Table 6-3. Start-up Times for the External Clock Selection

SUT1..0	Start-up Time from Power-down	Additional Delay from Reset	Recommended Usage
00	6 CK	14CK	BOD enabled
01	6 CK	14CK + 4 ms	Fast rising power
10	6 CK	14CK + 64 ms	Slowly rising power
11	Reserved		

When applying an external clock, it is required to avoid sudden changes in the applied clock frequency to ensure stable operation of the MCU. A variation in frequency of more than 2% from one clock cycle to the next can lead to unpredictable behavior. It is required to ensure that the MCU is kept in Reset during such changes in the clock frequency.

Note that the System Clock Prescaler can be used to implement run-time changes of the internal clock frequency while still ensuring stable operation. See "System Clock Prescaler" on page 30 for details.

6.2.2 Calibrated Internal 8 MHz Oscillator

By default, the Internal Oscillator provides an approximate 8 MHz clock. Though voltage and temperature dependent, this clock can be very accurately calibrated by the user. See Table 20-2 on page 176 and "Internal Oscillator Speed" on page 205 for more details. The device is shipped with the CKDIV8 Fuse programmed. See "System Clock Prescaler" on page 30 for more details.

This clock may be selected as the system clock by programming the CKSEL Fuses as shown in Table 6-4. If selected, it will operate with no external components. During reset, hardware loads the pre-programmed calibration value into the OSCCAL Register and thereby automatically cal-

ibrates the RC Oscillator. The accuracy of this calibration is shown as Factory calibration in Table 20-2 on page 176.

By changing the OSCCAL register from SW, see "OSCCAL – Oscillator Calibration Register" on page 30, it is possible to get a higher calibration accuracy than by using the factory calibration. The accuracy of this calibration is shown as User calibration in Table 20-2 on page 176.

When this Oscillator is used as the chip clock, the Watchdog Oscillator will still be used for the Watchdog Timer and for the Reset Time-out. For more information on the pre-programmed calibration value, see the section "Calibration Byte" on page 162.

Table 6-4. Internal Calibrated RC Oscillator Operating Modes

CKSEL3..0	Nominal Frequency
0010 (1)	8.0 MHz

Note: 1. The device is shipped with this option selected.

When this oscillator is selected, start-up times are determined by the SUT Fuses as shown in Table 6-5..

Table 6-5. Start-up Times for the Internal Calibrated RC Oscillator Clock Selection

SUT1..0	Start-up Time from Power-down	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	6 CK	14CK(2)	BOD enabled
01	6 CK	14CK + 4 ms	Fast rising power
10(1)	6 CK	14CK + 64 ms	Slowly rising power
11	Reserved		

Note: 1. The device is shipped with this option selected.
2. If the RSTDISBL fuse is programmed, this start-up time will be increased to 14CK + 4 ms to ensure programming mode can be entered.

6.2.3 Internal 128 kHz Oscillator

The 128 kHz internal oscillator is a low power oscillator providing a clock of 128 kHz. The frequency depends on supply voltage, temperature and batch variations. This clock may be selected as the system clock by programming the CKSEL Fuses to "0100".

When this clock source is selected, start-up times are determined by the SUT Fuses as shown in Table 6-6.

Table 6-6. Start-up Times for the 128 kHz Internal Oscillator

SUT1..0	Start-up Time from Power-down	Additional Delay from Reset	Recommended Usage
00	6 CK	14CK(1)	BOD enabled
01	6 CK	14CK + 4 ms	Fast rising power
10	6 CK	14CK + 64 ms	Slowly rising power
11	Reserved		

Note: 1. If the RSTDISBL fuse is programmed, this start-up time will be increased to 14CK + 4 ms to ensure programming mode can be entered.

6.2.4 Low-Frequency Crystal Oscillator

To use a 32.768 kHz watch crystal as the clock source for the device, the low-frequency crystal oscillator must be selected by setting CKSEL fuses to '0110'. The crystal should be connected as shown in Figure 6-3. To find suitable capacitors please consult the manufacturer's datasheet.

For this oscillator start-up times can be set with the SUT fuses, as shown in Table 6-7.

Table 6-7. Start-up Times for the Low-Frequency Crystal Oscillator Clock Selection

SUT1..0	Start-up Time from Power Down	Additional Delay from Reset	Recommended usage
00	1K CK(1)	4 ms	Fast rising power or BOD enabled
01	1K CK(1)	64 ms	Slowly rising power
10	32K CK	64 ms	Stable frequency at start-up
11	Reserved		

Notes: 1. These options should be used only if frequency stability at start-up is not important.

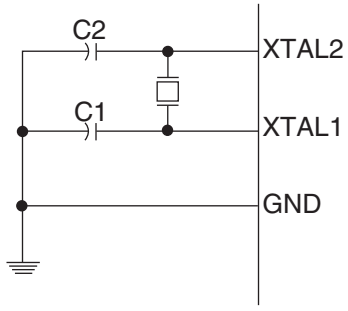
Table 6-8. Capacitance for the Low-Frequency Crystal Oscillator

Device	32 kHz Osc. Type	Cap (Xtal1/Tosc1)	Cap (Xtal2/Tosc2)
ATtiny24/44/88	System Osc.	16 pF	6 pF

6.2.5 Crystal Oscillator / Ceramic Resonator

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 6-3 Either a quartz crystal or a ceramic resonator may be used.

Figure 6-3. Crystal Oscillator Connections



C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 6-9 below. For ceramic resonators, the capacitor values given by the manufacturer should be used.

Table 6-9. Crystal Oscillator Operating Modes

CKSEL3..1	Frequency Range (MHz)	Recommended C1 and C2 Value (pF)
100[1]	0.4 - 0.9	–
101	0.9 - 3.0	12 - 22
110	3.0 - 8.0	12 - 22
111	8.0 -	12 - 22

Notes: 1. This option should not be used with crystals, only with ceramic resonators.

The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by fuses CKSEL3..1 as shown in Table 6-9.

The CKSEL0 Fuse together with the SUT1..0 Fuses select the start-up times as shown in Table 6-10.

Table 6-10. Start-up Times for the Crystal Oscillator Clock Selection

CKSEL0	SUT1..0	Start-up Time from Power-down[1]	Additional Delay from Reset	Recommended Usage
0	00	258 CK[2]	14CK + 4 ms	Ceramic resonator, fast rising power
0	01	258 CK[2]	14CK + 64 ms	Ceramic resonator, slowly rising power
0	10	1K CK[3]	14CK	Ceramic resonator, BOD enabled
0	11	1K CK[3]	14CK + 4 ms	Ceramic resonator, fast rising power
1	00	1K CK[3]	14CK + 64 ms	Ceramic resonator, slowly rising power
1	01	16K CK	14CK	Crystal Oscillator, BOD enabled
1	10	16K CK	14CK + 4 ms	Crystal Oscillator, fast rising power
1	11	16K CK	14CK + 64 ms	Crystal Oscillator, slowly rising power

Notes:
1. When the BOD has been disabled by software, the wake-up time from sleep mode will be approximately 60µs to ensure that the BOD is working correctly before the MCU continues executing code.
2. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.
3. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

6.2.6 Default Clock Source

The device is shipped with CKSEL = “0010”, SUT = “10”, and CKDIV8 programmed. The default clock source setting is therefore the Internal Oscillator running at 8.0 MHz with longest start-up time and an initial system clock prescaling of 8, resulting in 1.0 MHz system clock. This default setting ensures that all users can make their desired clock source setting using an in-system or high-voltage programmer.

For low-voltage devices (ATtiny24V/44V/84V) it should be noted that unprogramming the CKDIV8 fuse may result in overclocking. At low voltages (below 2.7V) the devices are rated for maximum 4 MHz operation (see Section 20.3 on page 175), but routing the clock signal from the internal oscillator directly to the system clock line will run the device at 8 MHz.

6.3 System Clock Prescaler

The ATtiny24/44/84 system clock can be divided by setting the “CLKPR – Clock Prescale Register” on page 31. 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 6-11 on page 32.

6.3.1 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 neither the clock frequency corresponding to the previous setting, nor 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, 2 active clock edges are produced. Here, T1 is the previous clock period, and T2 is the period corresponding to the new prescaler setting.

6.4 Clock Output Buffer

The device can output the system clock on the CLKO pin. To enable the output, the CKOUT fuse has to be programmed. This mode is suitable when the chip clock is used to drive other circuits on the system. Note that the clock will not be output during reset and that the normal operation of the I/O pin will be overridden when the fuse is programmed. Any clock source, including the internal RC Oscillator, can be selected when the clock is output on CLKO. If the System Clock Prescaler is used, it is the divided system clock that is output.

6.5 Register Description

6.5.1 OSCCAL – Oscillator Calibration Register

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								

- **Bits 7:0 – CAL7:0: Oscillator 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 20-2 on page 176. The application software can write this register to change the oscillator frequency. The oscillator can be calibrated to frequencies as specified in Table 20-2 on page 176. 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 8.8 MHz. Otherwise, the EEPROM or Flash write may fail.

The CAL7 bit determines the range of operation for the oscillator. Setting this bit to 0 gives the lowest frequency range, setting this bit to 1 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. See "Calibrated 8 MHz RC Oscillator Frequency vs, OSCCAL Value" on page 207 for typical frequencies.

To ensure stable operation of the MCU the calibration value should be changed in small. A variation in frequency of more than 2% from one cycle to the next can lead to unpredicatble behavior. Changes in OSCCAL should not exceed 0x20 for each calibration. It is required to ensure that the MCU is kept in Reset during such changes in the clock frequency.

6.5.2 CLKPR – Clock Prescale Register

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 logic one to enable change of the CLKPS bits. The CLKPCE bit is only updated when the other bits in CLKPR are simultaniosly written to 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.

- **Bits 6..4 – Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bits 3..0 – CLKPS3..0: Clock Prescaler Select Bits 3 - 0**

These bits define the division factor between the selected clock source and the internal system clock. These bits can be written run-time to vary the clock frequency to suit the application requirements. As the divider divides the master clock input to the MCU, the speed of all synchronous peripherals is reduced when a division factor is used. The division factors are given in Table 6-11 on page 32.

To avoid unintentional changes of clock frequency, a special write procedure must be followed to change the CLKPS bits:

1. Write the Clock Prescaler Change Enable (CLKPCE) bit to one and all other bits in CLKPR to zero.
2. Within four cycles, write the desired value to CLKPS while writing a zero to CLKPCE.

Interrupts must be disabled when changing prescaler setting to make sure the write procedure is not interrupted.

The CKDIV8 Fuse determines the initial value of the CLKPS bits. If CKDIV8 is unprogrammed, the CLKPS bits will be reset to “0000”. If CKDIV8 is programmed, CLKPS bits are reset to “0011”, giving a division factor of eight at start up. This feature should be used if the selected clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. Note that any value can be written to the CLKPS bits regardless of the CKDIV8 Fuse setting. The Application software must ensure that a sufficient division factor is chosen if the selcted clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. The device is shipped with the CKDIV8 Fuse programmed.

Table 6-11. Clock Prescaler Select

CLKPS3	CLKPS2	CLKPS1	CLKPS0	Clock Division Factor
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	Reserved
1	0	1	0	Reserved
1	0	1	1	Reserved
1	1	0	0	Reserved
1	1	0	1	Reserved
1	1	1	0	Reserved
1	1	1	1	Reserved

7. Power Management and Sleep Modes

The high performance and industry leading code efficiency makes the AVR microcontrollers an ideal choise for low power applications. In addition, sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the application's requirements.

7.1 Sleep Modes

Figure 6-1 on page 24 presents the different clock systems and their distribution in ATtiny24/44/84. The figure is helpful in selecting an appropriate sleep mode. Table 7-1 shows the different sleep modes and their wake up sources.

Table 7-1. Active Clock Domains and Wake-up Sources in Different Sleep Modes

	Active Clock Domains				Oscillators	Wake-up Sources				
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	Main Clock Source Enabled	INT0 and Pin Change	SPM/EEPROM Ready Interrupt	ADC Interrupt	Other I/O	Watchdog Interrupt
Idle			X	X	X	X	X	X	X	X
ADC Noise Reduction				X	X	X(1)	X	X		X
Power-down						X(1)				X
Stand-by						X(1)				X

Note: 1. For INT0, only level interrupt.

To enter any of the three sleep modes, the SE bit in MCUCR must be written to logic one and a SLEEP instruction must be executed. The SM1..0 bits in the MCUCR Register select which sleep mode (Idle, ADC Noise Reduction, Standby or Power-down) will be activated by the SLEEP instruction. See Table 7-2 on page 37 for a summary.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the Register File and SRAM are unaltered when the device wakes up from sleep. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset Vector.

Note that if a level triggered interrupt is used for wake-up the changed level must be held for some time to wake up the MCU (and for the MCU to enter the interrupt service routine). See "External Interrupts" on page 49 for details.

7.1.1 Idle Mode

When the SM1..0 bits are written to 00, the SLEEP instruction makes the MCU enter Idle mode, stopping the CPU but allowing Analog Comparator, ADC, Timer/Counter, Watchdog, and the interrupt system to continue operating. This sleep mode basically halts clk_{CPU} and clk_{FLASH}, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow. If wake-up from the Analog Comparator interrupt is not required,

the Analog Comparator can be powered down by setting the ACD bit in “ACSR – Analog Comparator Control and Status Register” on page 130. This will reduce power consumption in Idle mode. If the ADC is enabled, a conversion starts automatically when this mode is entered.

7.1.2 ADC Noise Reduction Mode

When the SM1..0 bits are written to 01, the SLEEP instruction makes the MCU enter ADC Noise Reduction mode, stopping the CPU but allowing the ADC, the external interrupts, and the Watchdog to continue operating (if enabled). This sleep mode halts $clk_{I/O}$, clk_{CPU}, and clk_{FLASH}, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart form the ADC Conversion Complete interrupt, only an External Reset, a Watchdog Reset, a Brown-out Reset, an SPM/EEPROM ready interrupt, an external level interrupt on INT0 or a pin change interrupt can wake up the MCU from ADC Noise Reduction mode.

7.1.3 Power-Down Mode

When the SM1:0 bits are written to 10, the SLEEP instruction makes the MCU enter Power-down mode. In this mode, the Oscillator is stopped, while the external interrupts, and the Watchdog continue operating (if enabled). Only an External Reset, a Watchdog Reset, a Brown-out Reset, an external level interrupt on INT0, or a pin change interrupt can wake up the MCU. This sleep mode halts all generated clocks, allowing operation of asynchronous modules only.

7.1.4 Standby Mode

When the SM1..0 bits are 11 and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Standby mode. This mode is identical to Power-down with the exception that the Oscillator is kept running. From Standby mode, the device wakes up in six clock cycles.

7.2 Software BOD Disable

When the Brown-out Detector (BOD) is enabled by BODLEVEL fuses (see Table 19-4 on page 160), the BOD is actively monitoring the supply voltage during a sleep period. In some devices it is possible to save power by disabling the BOD by software in Power-Down and Stand-By sleep modes. The sleep mode power consumption will then be at the same level as when BOD is globally disabled by fuses.

If BOD is disabled by software, the BOD function is turned off immediately after entering the sleep mode. Upon wake-up from sleep, BOD is automatically enabled again. This ensures safe operation in case the V_{CC} level has dropped during the sleep period.

When the BOD has been disabled, the wake-up time from sleep mode will be approximately 60µs to ensure that the BOD is working correctly before the MCU continues executing code.

BOD disable is controlled by the BODS (BOD Sleep) bit of MCU Control Register, see “MCUCR – MCU Control Register” on page 36. Writing this bit to one turns off BOD in Power-Down and Stand-By, while writing a zero keeps the BOD active. The default setting is zero, i.e. BOD active.

Writing to the BODS bit is controlled by a timed sequence and an enable bit, see “MCUCR – MCU Control Register” on page 36.

7.2.1 Limitations

BOD disable functionality has been implemented in the following devices, only:

- ATtiny24, revision E, and newer
- ATtiny44, revision D, and newer
- ATtiny84, revision B, and newer

Revisions are marked on the device package and can be located as follows:

- Bottom side of packages 14P3 and 14S1
- Top side of package 20M1

7.3 Power Reduction Register

The Power Reduction Register (PRR), see “PRR – Power Reduction Register” on page 37, provides a method to reduce power consumption by stopping the clock to individual peripherals. The current state of the peripheral is frozen and the I/O registers can not be read or written. Resources used by the peripheral when stopping the clock will remain occupied, hence the peripheral should in most cases be disabled before stopping the clock. Waking up a module, which is done by clearing the bit in PRR, puts the module in the same state as before shutdown.

Module shutdown can be used in Idle mode and Active mode to significantly reduce the overall power consumption. In all other sleep modes, the clock is already stopped. See “Supply Current of I/O Modules” on page 185 for examples.

7.4 Minimizing Power Consumption

There are several issues to consider when trying to minimize the power consumption in an AVR controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the device’s functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

7.4.1 Analog to Digital Converter

If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. See “Analog to Digital Converter” on page 132 for details on ADC operation.

7.4.2 Analog Comparator

When entering Idle mode, the Analog Comparator should be disabled if not used. When entering ADC Noise Reduction mode, the Analog Comparator should be disabled. In the other sleep modes, the Analog Comparator is automatically disabled. However, if the Analog Comparator is set up to use the Internal Voltage Reference as input, the Analog Comparator should be disabled in all sleep modes. Otherwise, the Internal Voltage Reference will be enabled, independent of sleep mode. See “Analog Comparator” on page 129 for details on how to configure the Analog Comparator.

7.4.3 Brown-out Detector

If the Brown-out Detector is not needed in the application, this module should be turned off. If the Brown-out Detector is enabled by the BODLEVEL Fuses, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. See “Brown-out Detection” on page 41 and “Software BOD Disable” on page 34 for details on how to configure the Brown-out Detector.

7.4.4 Internal Voltage Reference

The Internal Voltage Reference will be enabled when needed by the Brown-out Detection, the Analog Comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. See “Internal Voltage Reference” on page 42 for details on the start-up time.

7.4.5 Watchdog Timer

If the Watchdog Timer is not needed in the application, this module should be turned off. If the Watchdog Timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. See “Watchdog Timer” on page 42 for details on how to configure the Watchdog Timer.

7.4.6 Port Pins

When entering a sleep mode, all port pins should be configured to use minimum power. The most important thing is then to ensure that no pins drive resistive loads. In sleep modes where both the I/O clock ($clk_{I/O}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. See the section “Digital Input Enable and Sleep Modes” on page 58 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or has an analog signal level close to $V_{CC}/2$, the input buffer will use excessive power.

For analog input pins, the digital input buffer should be disabled at all times. An analog signal level close to $V_{CC}/2$ on an input pin can cause significant current even in active mode. Digital input buffers can be disabled by writing to the Digital Input Disable Register (DIDR0). See “DIDR0 – Digital Input Disable Register 0” on page 150 for details.

7.5 Register Description

7.5.1 MCUCR – MCU Control Register

The MCU Control Register contains control bits for power management.

Bit	7	6	5	4	3	2	1	0	
	BODS	PUD	SE	SM1	SM0	BODSE	ISC01	ISC00	MCUCR
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 – BODS: BOD Sleep**

BOD disable functionality is available in some devices, only. See “Limitations” on page 35.

In order to disable BOD during sleep (see Table 7-1 on page 33) the BODS bit must be written to logic one. This is controlled by a timed sequence and the enable bit, BODSE in MCUCR. First,

both BODS and BODSE must be set to one. Second, within four clock cycles, BODS must be set to one and BODSE must be set to zero. The BODS bit is active three clock cycles after it is set. A sleep instruction must be executed while BODS is active in order to turn off the BOD for the actual sleep mode. The BODS bit is automatically cleared after three clock cycles.

In devices where Sleeping BOD has not been implemented this bit is unused and will always read zero.

- **Bit 5 – SE: Sleep Enable**

The SE bit must be written to logic one to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode unless it is the programmer's purpose, it is recommended to write the Sleep Enable (SE) bit to one just before the execution of the SLEEP instruction and to clear it immediately after waking up.

- **Bits 4, 3 – SM1..0: Sleep Mode Select Bits 2..0**

These bits select between the three available sleep modes as shown in Table 7-2.

Table 7-2. Sleep Mode Select

SM1	SM0	Sleep Mode
0	0	Idle
0	1	ADC Noise Reduction
1	0	Power-down
1	1	Standby(1)

Note: 1. Only recommended with external crystal or resonator selected as clock source

- **Bit 2 – BODSE: BOD Sleep Enable**

BOD disable functionality is available in some devices, only. See "Limitations" on page 35.

The BODSE bit enables setting of BODS control bit, as explained on BODS bit description. BOD disable is controlled by a timed sequence.

This bit is unused in devices where software BOD disable has not been implemented and will read as zero in those devices.

7.5.2 PRR – Power Reduction Register

The Power Reduction Register provides a method to reduce power consumption by allowing peripheral clock signals to be disabled.

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	PRTIM1	PRTIM0	PRUSI	PRADC	PRR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7, 6, 5, 4 – Res: Reserved Bits**

These bits are reserved and will always read zero.

- **Bit 3 – PRTIM1: Power Reduction Timer/Counter1**

Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 2 – PRTIM0: Power Reduction Timer/Counter0**

Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 1 – PRUSI: Power Reduction USI**

Writing a logic one to this bit shuts down the USI by stopping the clock to the module. When waking up the USI again, the USI should be re initialized to ensure proper operation.

- **Bit 0 – PRADC: Power Reduction ADC**

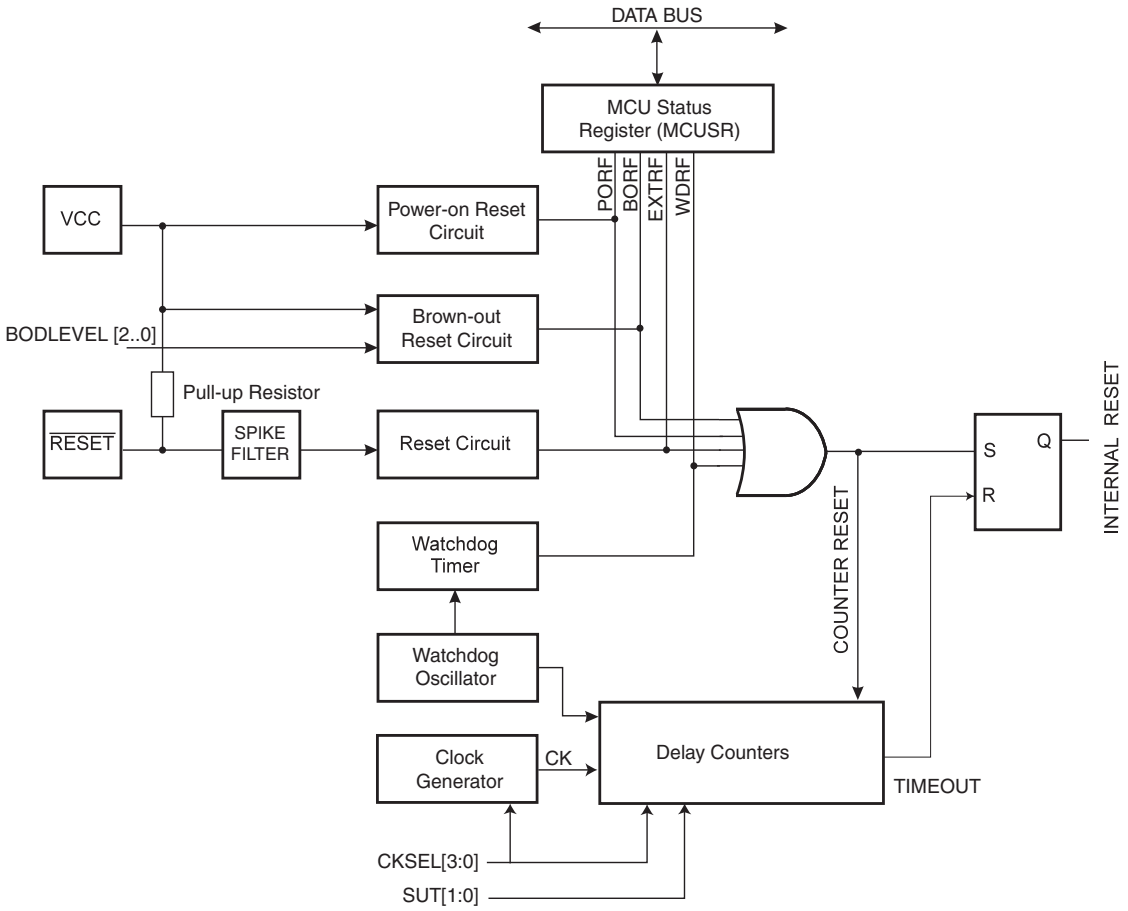
Writing a logic one to this bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot be used when the ADC is shut down.

8. System Control and Reset

8.1 Resetting the AVR

During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. The instruction placed at the Reset Vector must be a RJMP – Relative Jump – instruction to the reset handling routine. 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 circuit diagram in Figure 8-1 shows the reset logic. Electrical parameters of the reset circuitry are given in Table 20-4 on page 177.

Figure 8-1. Reset Logic



The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the SUT and CKSEL Fuses. The different selections for the delay period are presented in "Clock Sources" on page 25.

8.2 Reset Sources

The ATtiny24/44/84 has four sources of reset:

- Power-on Reset. The MCU is reset when the supply voltage is below the Power-on Reset threshold (V_{POT}).
- External Reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length when $\overline{\text{RESET}}$ function is enabled.
- Watchdog Reset. The MCU is reset when the Watchdog Timer period expires and the Watchdog is enabled.
- Brown-out Reset. The MCU is reset when the supply voltage V_{CC} is below the Brown-out Reset threshold (V_{BOT}) and the Brown-out Detector is enabled.

8.2.1 Power-on Reset

A Power-on Reset (POR) pulse is generated by an On-chip detection circuit. The detection level is defined in "System and Reset Characteristics" on page 177. 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 Power-on Reset (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 8-2. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}

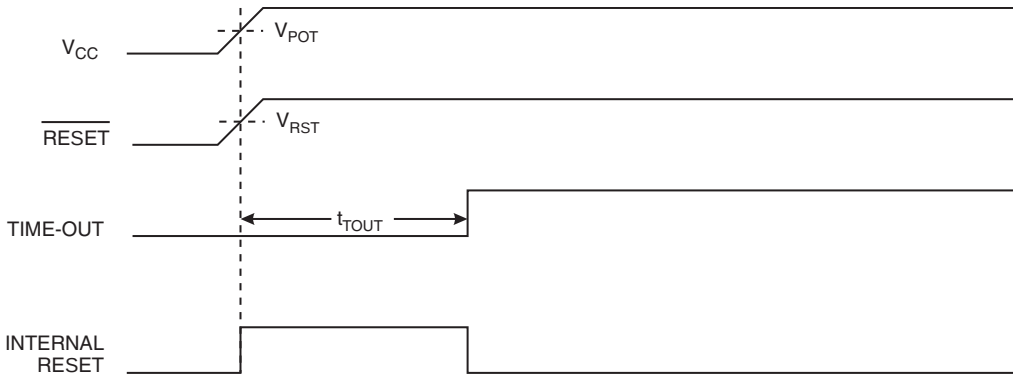
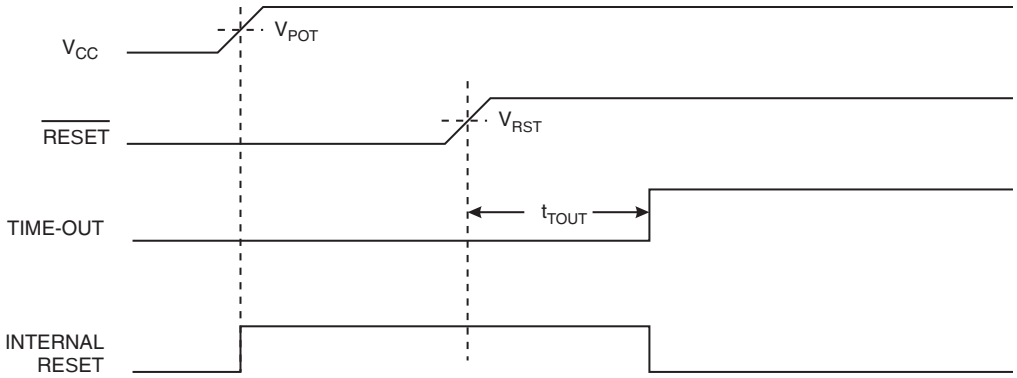


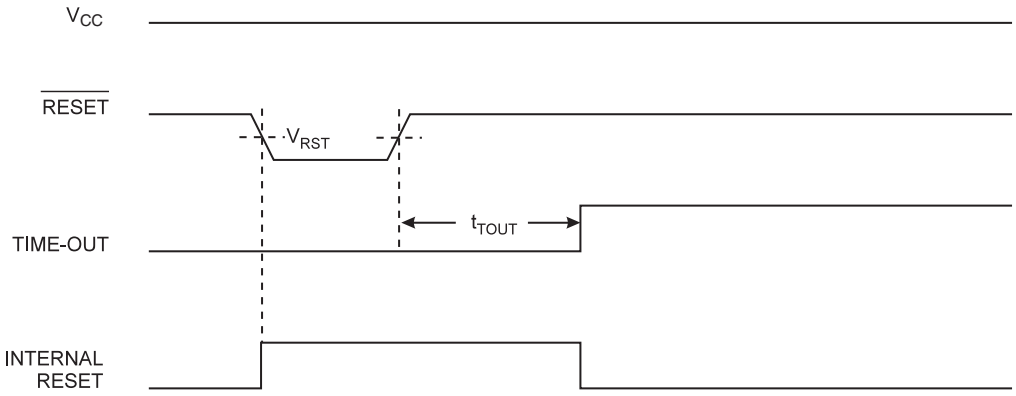
Figure 8-3. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally



8.2.2 External Reset

An External Reset is generated by a low level on the $\overline{\text{RESET}}$ pin if enabled. Reset pulses longer than the minimum pulse width (see "System and Reset Characteristics" on page 177) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} – on its positive edge, the delay counter starts the MCU after the Time-out period – t_{TOUT} – has expired.

Figure 8-4. External Reset During Operation



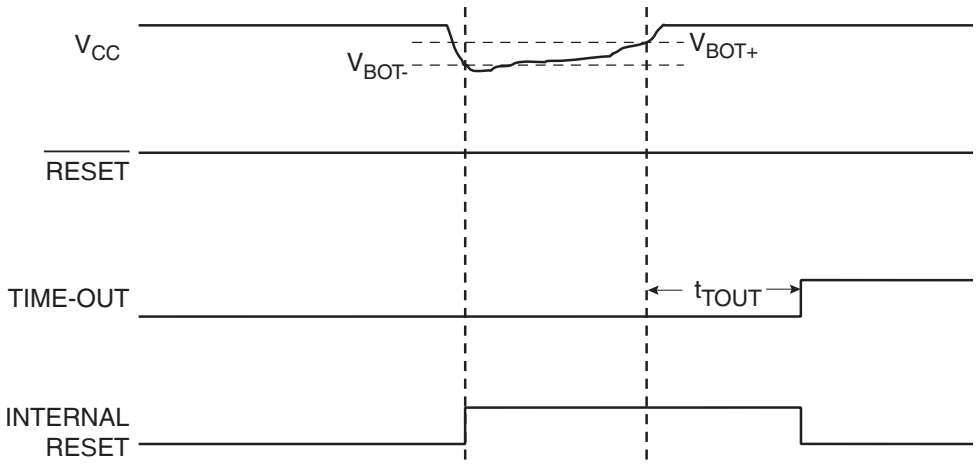
8.2.3 Brown-out Detection

ATtiny24/44/84 has an On-chip Brown-out Detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the BODLEVEL Fuses. The trigger level has a hysteresis to ensure spike free Brown-out Detection. The hysteresis on the detection level should be interpreted as $V_{BOT+} = V_{BOT} + V_{HYST}/2$ and $V_{BOT-} = V_{BOT} - V_{HYST}/2$.

When the BOD is enabled, and V_{CC} decreases to a value below the trigger level (V_{BOT-} in Figure 8-5 on page 41), the Brown-out Reset is immediately activated. When V_{CC} increases above the trigger level (V_{BOT+} in Figure 8-5 on page 41), the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in "System and Reset Characteristics" on page 177.

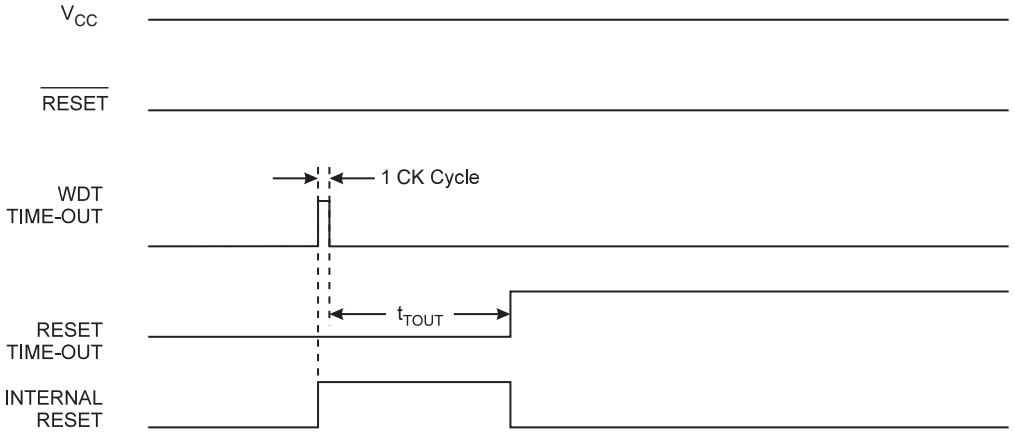
Figure 8-5. Brown-out Reset During Operation



8.2.4 Watchdog Reset

When the Watchdog times out, it will generate a short reset pulse of one CK cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period t_{TOUT}. See "Watchdog Timer" on page 42 for details on operation of the Watchdog Timer.

Figure 8-6. Watchdog Reset During Operation



8.3 Internal Voltage Reference

ATtiny24/44/84 features an internal bandgap reference. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC. The bandgap voltage varies with supply voltage and temperature.

8.3.1 Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in "System and Reset Characteristics" on page 177. To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODLEVEL [2..0] Fuse).
2. When the internal reference is connected to the Analog Comparator (by setting the ACBG bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or ADC is used. To reduce power consumption in Power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering Power-down mode.

8.4 Watchdog Timer

The Watchdog Timer is clocked from an On-chip Oscillator which runs at 128 kHz. By controlling the Watchdog Timer prescaler, the Watchdog Reset interval can be adjusted as shown in Table 8-3 on page 47. The WDR – Watchdog Reset – instruction resets the Watchdog Timer. The Watchdog Timer is also reset when it is disabled and when a Chip Reset occurs. Ten different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog Reset, the ATtiny24/44/84 resets and executes from the Reset Vector. For timing details on the Watchdog Reset, refer to Table 8-3 on page 47.

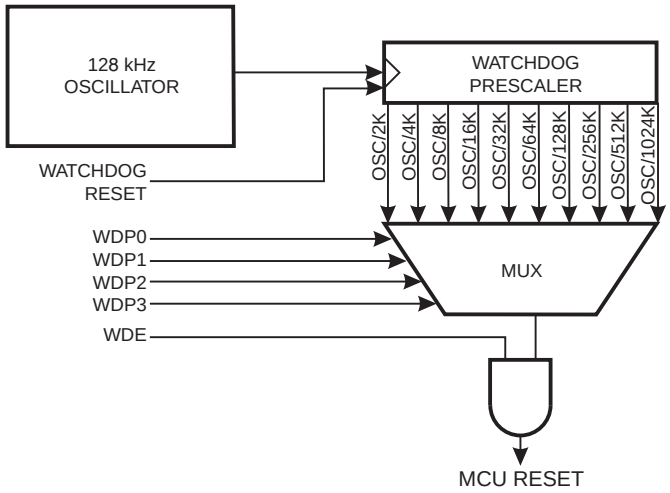
The Wathdog Timer can also be configured to generate an interrupt instead of a reset. This can be very helpful when using the Watchdog to wake-up from Power-down.

To prevent unintentional disabling of the Watchdog or unintentional change of time-out period, two different safety levels are selected by the fuse WDTON as shown in Table 8-1 See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 43 for details.

Table 8-1. WDT Configuration as a Function of the Fuse Settings of WDTON

WDTON	Safety Level	WDT Initial State	How to Disable the WDT	How to Change Time-out
Unprogrammed	1	Disabled	Timed sequence	No limitations
Programmed	2	Enabled	Always enabled	Timed sequence

Figure 8-7. Watchdog Timer



8.4.1 Timed Sequences for Changing the Configuration of the Watchdog Timer

The sequence for changing configuration differs slightly between the two safety levels. Separate procedures are described for each level.

8.4.1.1 Safety Level 1

In this mode, the Watchdog Timer is initially disabled, but can be enabled by writing the WDE bit to one without any restriction. A timed sequence is needed when disabling an enabled Watchdog Timer. To disable an enabled Watchdog Timer, the following procedure must be followed:

1. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE regardless of the previous value of the WDE bit.
2. Within the next four clock cycles, in the same operation, write the WDE and WDP bits as desired, but with the WDCE bit cleared.

8.4.1.2 Safety Level 2

In this mode, the Watchdog Timer is always enabled, and the WDE bit will always read as one. A timed sequence is needed when changing the Watchdog Time-out period. To change the Watchdog Time-out, the following procedure must be followed:

1. In the same operation, write a logical one to WDCE and WDE. Even though the WDE always is set, the WDE must be written to one to start the timed sequence.
2. Within the next four clock cycles, in the same operation, write the WDP bits as desired, but with the WDCE bit cleared. The value written to the WDE bit is irrelevant.

8.4.2 Code Example

The following code example shows one assembly and one C function for turning off the WDT. The example assumes that interrupts are controlled (e.g., by disabling interrupts globally) so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
WDT_off:
  wdr
  ; Clear WDRF in MCUSR
  ldi  r16, (0<<WDRF)
  out  MCUSR, r16
  ; Write logical one to WDCE and WDE
  ; Keep old prescaler setting to prevent unintentional Watchdog Reset
  in   r16, WDTCSR
  ori  r16, (1<<WDCE)|(1<<WDE)
  out  WDTCSR, r16
  ; Turn off WDT
  ldi  r16, (0<<WDE)
  out  WDTCSR, r16
  ret
```

C Code Example

```
void WDT_off(void)
{
  _WDR();
  /* Clear WDRF in MCUSR */
  MCUSR = 0x00
  /* Write logical one to WDCE and WDE */
  WDTCSR |= (1<<WDCE) | (1<<WDE);
  /* Turn off WDT */
  WDTCSR = 0x00;
}
```

Note: See “Code Examples” on page 6.

8.5 Register Description

8.5.1 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 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 logic 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 logic 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 logic 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.

8.5.2 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 Timeout 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 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 timeout 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.

Table 8-2. Watchdog Timer Configuration

WDE	WDIE	Watchdog Timer State	Action on Time-out
0	0	Stopped	None
0	1	Running	Interrupt
1	0	Running	Reset
1	1	Running	Interrupt

- **Bit 4 – WDCE: Watchdog Change Enable**

This bit must be set when the WDE bit is written to logic zero. Otherwise, the Watchdog will not be disabled. Once written to one, hardware will clear this bit after four clock cycles. See the description of the WDE bit for a Watchdog disable procedure. This bit must also be set when changing the prescaler bits. See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 43.

- **Bit 3 – WDE: Watchdog Enable**

When the WDE is written to logic one, the Watchdog Timer is enabled, and if the WDE is written to logic zero, the Watchdog Timer function is disabled. WDE can only be cleared if the WDCE bit has logic level one. To disable an enabled Watchdog Timer, the following procedure must be followed:

1. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE even though it is set to one before the disable operation starts.
2. Within the next four clock cycles, write a logic 0 to WDE. This disables the Watchdog.

In safety level 2, it is not possible to disable the Watchdog Timer, even with the algorithm described above. See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 43.

In safety level 1, WDE is overridden by WDRF in MCUSR. See “MCUSR – MCU Status Register” on page 45 for description of WDRF. This means that WDE is always set when WDRF is set. To clear WDE, WDRF must be cleared before disabling the Watchdog with the procedure described above. This feature ensures multiple resets during conditions causing failure, and a safe start-up after the failure.

Note: If the watchdog timer is not going to be used in the application, it is important to go through a watchdog disable procedure in the initialization of the device. If the Watchdog is accidentally enabled, for example by a runaway pointer or brown-out condition, the device will be reset, which in turn will lead to a new watchdog reset. To avoid this situation, the application software should always clear the WDRF flag and the WDE control bit in the initialization routine.

- **Bits 5, 2..0 – WDP3..0: Watchdog Timer Prescaler 3, 2, 1, and 0**

The WDP3..0 bits determine the Watchdog Timer prescaling when the Watchdog Timer is enabled. The different prescaling values and their corresponding Timeout Periods are shown in Table 8-3.

Table 8-3. Watchdog Timer Prescale Select

WDP3	WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at V_{CC} = 5.0V
0	0	0	0	2K cycles	16 ms
0	0	0	1	4K cycles	32 ms
0	0	1	0	8K cycles	64 ms
0	0	1	1	16K cycles	0.125 s
0	1	0	0	32K cycles	0.25 s
0	1	0	1	64K cycles	0.5 s
0	1	1	0	128K cycles	1.0 s
0	1	1	1	256K cycles	2.0 s
1	0	0	0	512K cycles	4.0 s
1	0	0	1	1024K cycles	8.0 s
1	0	1	0	Reserved[1]	
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

Note: 1. If selected, one of the valid settings below 0b1010 will be used.

9. Interrupts

This section describes the specifics of the interrupt handling as performed in ATtiny24/44/84. For a general explanation of the AVR interrupt handling, see “Reset and Interrupt Handling” on page 12.

9.1 Interrupt Vectors

The interrupt vectors of ATtiny24/44/84 are described in Table 9-1 below.

Table 9-1. 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	TIM1_CAPT	Timer/Counter1 Capture Event
7	0x0006	TIM1_COMPA	Timer/Counter1 Compare Match A
8	0x0007	TIM1_COMPB	Timer/Counter1 Compare Match B
9	0x0008	TIM1_OVF	Timer/Counter1 Overflow
10	0x0009	TIM0_COMPA	Timer/Counter0 Compare Match A
11	0x000A	TIM0_COMPB	Timer/Counter0 Compare Match B
12	0x000B	TIM0_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_STR	USI START
17	0x0010	USI_OVF	USI Overflow

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 setup for Reset and Interrupt Vector Addresses in ATtiny24/44/84 is shown in the program example below.

```
Address  Labels Code                 Comments

0x0000         rjmp   RESET          ; Reset Handler
0x0001         rjmp   INT0           ; IRQ0 Handler
0x0002         rjmp   PCINT0         ; PCINT0 Handler
0x0003         rjmp   PCINT1         ; PCINT1 Handler
0x0004         rjmp   WDT            ; 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>
...            ...
```

9.2 External Interrupts

The External Interrupts are triggered by the INT0 pin or any of the PCINT11..0 pins. Observe that, if enabled, the interrupts will trigger even if the INT0 or PCINT11..0 pins are configured as outputs. This feature provides a way of generating a software interrupt. Pin change 0 interrupts PCI0 will trigger if any enabled PCINT7..0 pin toggles. Pin change 1 interrupts PCI1 will trigger if any enabled PCINT11..8 pin toggles. The PCMSK0 and PCMSK1 Registers control which pins contribute to the pin change interrupts. Pin change interrupts on PCINT11..0 are detected asynchronously. This implies that these interrupts can be used for waking the part also from sleep modes other than Idle mode.

The INT0 interrupts can be triggered by a falling or rising edge or a low level. This is set up as indicated in the specification for the MCU Control Register – MCUCR. When the INT0 interrupt is enabled and is configured as level triggered, the interrupt will trigger as long as the pin is held low. Note that recognition of falling or rising edge interrupts on INT0 requires the presence of an I/O clock, as described in “Clock Sources” on page 25.

9.2.1 Low Level Interrupt

A low level interrupt on INT0 is detected asynchronously. This means that the interrupt source can be used for waking the part also from sleep modes other than Idle (the I/O clock is halted in all sleep modes except Idle).

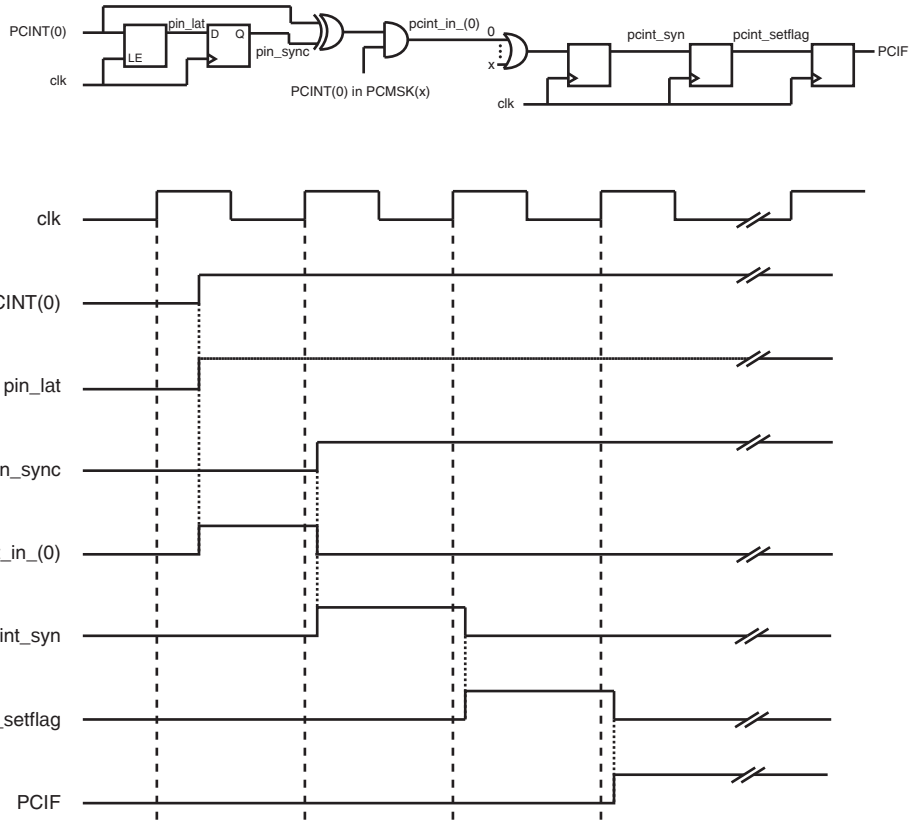
Note that if a level triggered interrupt is used for wake-up from Power-down, the required level must be held long enough for the MCU to complete the wake-up to trigger the level interrupt. If the level disappears before the end of the Start-up Time, the MCU will still wake up, but no interrupt will be generated. The start-up time is defined by the SUT and CKSEL fuses, as described in “Clock System” on page 24.

If the low level on the interrupt pin is removed before the device has woken up then program execution will not be diverted to the interrupt service routine but continue from the instruction following the SLEEP command.

9.2.2 Pin Change Interrupt Timing

An example of timing of a pin change interrupt is shown in Figure 9-1.

Figure 9-1. Timing of pin change interrupts



9.3 Register Description

9.3.1 MCUCR – MCU Control Register

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	BODS	PUD	SE	SM1	SM0	BODSE	ISC01	ISC00	MCUCR
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 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 9-2. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 9-2. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

9.3.2 GIMSK – General Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x3B (0x5B)	–	INT0	PCIE1	PCIE0	–	–	–	–	GIMSK
Read/Write	R	R/W	R/W	R/w	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7, 3..0 – Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bit 6 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control bits (ISC01 and ISC00) in the External Interrupt Control Register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of External Interrupt Request 0 is executed from the INT0 Interrupt Vector.

- **Bit 5 – PCIE1: Pin Change Interrupt Enable 1**

When the PCIE1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 1 is enabled. Any change on any enabled PCINT11..8 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI1 Interrupt Vector. PCINT11..8 pins are enabled individually by the PCMSK1 Register.

- **Bit 4– PCIE0: Pin Change Interrupt Enable 0**

When the PCIE0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 0 is enabled. Any change on any enabled PCINT7..0 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI0 Interrupt Vector. PCINT7..0 pins are enabled individually by the PCMSK0 Register.

9.3.3 GIFR – General Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x3A (0x5A	–	INTF0	PCIF1	PCIF0	–	–	–	–	GIFR
Read/Write	R	R/W	R/W	R/W	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7, 3..0 – Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bit 6 – INTF0: External Interrupt Flag 0**

When an edge or logic change on the INT0 pin triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in GIMSK are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it. This flag is always cleared when INT0 is configured as a level interrupt.

- **Bit 5 – PCIF1: Pin Change Interrupt Flag 1**

When a logic change on any PCINT11..8 pin triggers an interrupt request, PCIF1 becomes set (one). If the I-bit in SREG and the PCIE1 bit in GIMSK are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 4– PCIF0: Pin Change Interrupt Flag 0**

When a logic change on any PCINT7..0 pin triggers an interrupt request, PCIF becomes set (one). If the I-bit in SREG and the PCIE0 bit in GIMSK are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

9.3.4 PCMSK1 – Pin Change Mask Register 1

Bit	7	6	5	4	3	2	1	0	
0x20 (0x40)	–	–	–	–	PCINT11	PCINT10	PCINT9	PCINT8	PCMSK1
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7, 4– Res: Reserved Bits**

These bits are reserved and will always read as zero.

- **Bits 3..0 – PCINT11..8: Pin Change Enable Mask 11..8**

Each PCINT11..8 bit selects whether pin change interrupt is enabled on the corresponding I/O pin. If PCINT11..8 is set and the PCIE1 bit in GIMSK is set, pin change interrupt is enabled on the corresponding I/O pin. If PCINT11..8 is cleared, pin change interrupt on the corresponding I/O pin is disabled.

9.3.5 PCMSK0 – Pin Change Mask Register 0

Bit	7	6	5	4	3	2	1	0	
0x12 (0x32)	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	PCMSK0
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	

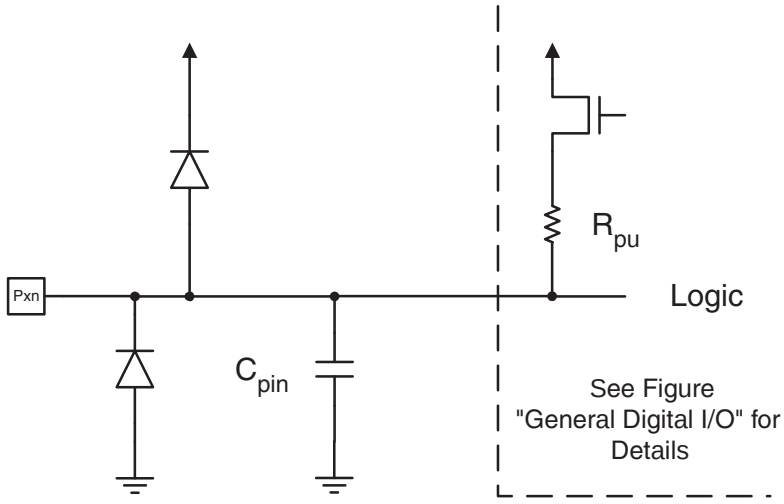
- **Bits 7..0 – PCINT7..0: Pin Change Enable Mask 7..0**

Each PCINT7..0 bit selects whether pin change interrupt is enabled on the corresponding I/O pin. If PCINT7..0 is set and the PCIE0 bit in GIMSK is set, pin change interrupt is enabled on the corresponding I/O pin. If PCINT7..0 is cleared, pin change interrupt on the corresponding I/O pin is disabled.

10. I/O Ports

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both V_{CC} and Ground as indicated in Figure 10-1 on page 54. See "Electrical Characteristics" on page 174 for a complete list of parameters.

Figure 10-1. I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case "x" represents the numbering letter for the port, and a lower case "n" represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. For example, PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in "Register Description" on page 67.

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

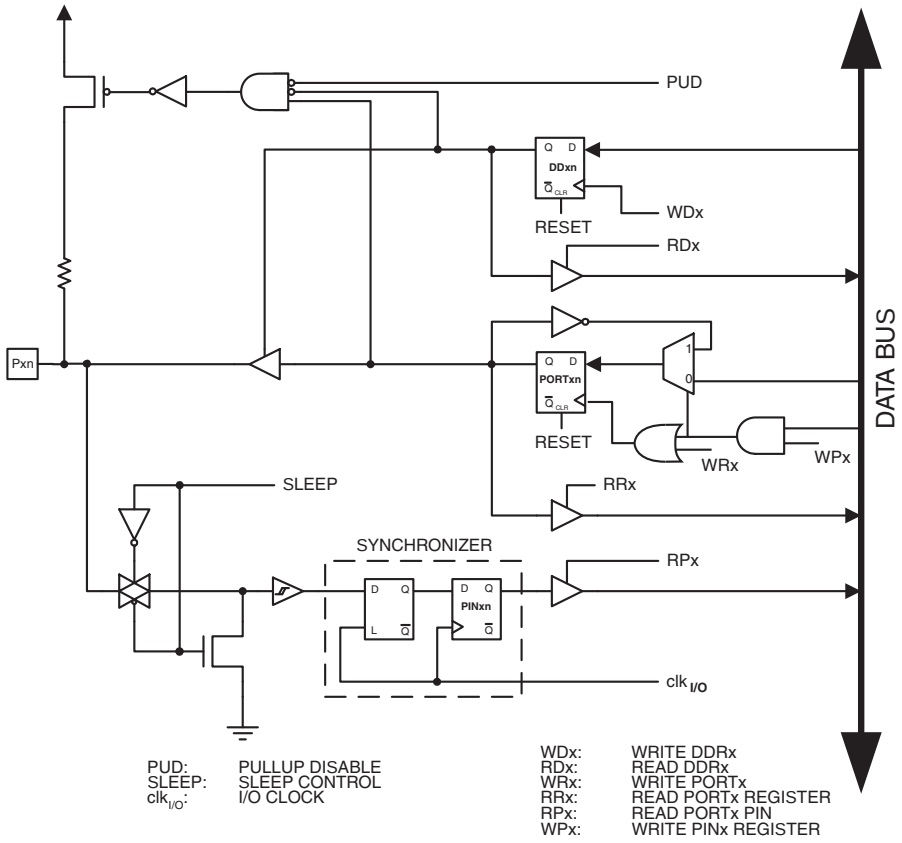
Using the I/O port as General Digital I/O is described in "Ports as General Digital I/O" on page 55. Most port pins are multiplexed with alternate functions for the peripheral features on the device. How each alternate function interferes with the port pin is described in "Alternate Port Functions" on page 58. Refer to the individual module sections for a full description of the alternate functions.

Note that enabling the alternate function of some of the port pins does not affect the use of the other pins in the port as general digital I/O.

10.1 Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 10-2 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 10-2. General Digital I/O[1]



Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

10.1.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in "Register Description" on page 67, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

10.1.2 Toggling the Pin

Writing a logic one to PINxn toggles the value of PORTxn, independent on the value of DDRxn. Note that the SBI instruction can be used to toggle one single bit in a port.

10.1.3 Switching Between Input and Output

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled {DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the MCUCR Register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b10) as an intermediate step.

Table 10-1 summarizes the control signals for the pin value.

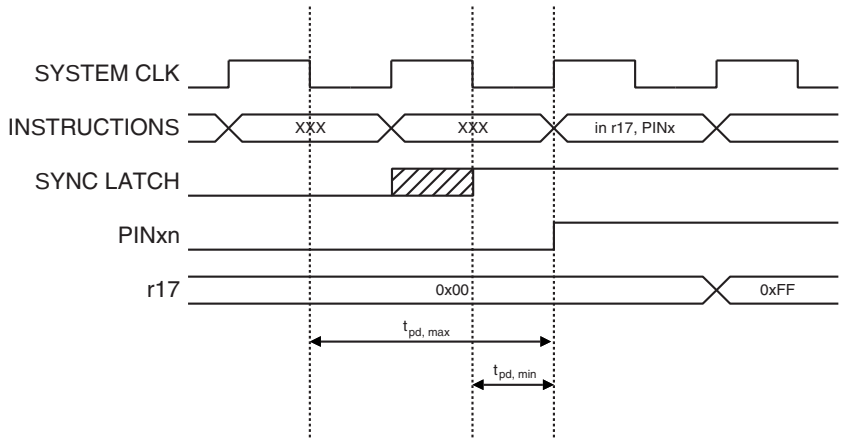
Table 10-1. Port Pin Configurations

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

10.1.4 Reading the Pin Value

Independent of the setting of Data Direction bit DDxn, the port pin can be read through the PINxn Register bit. As shown in Figure 10-2 on page 55, the PINxn Register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 10-3 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$ respectively.

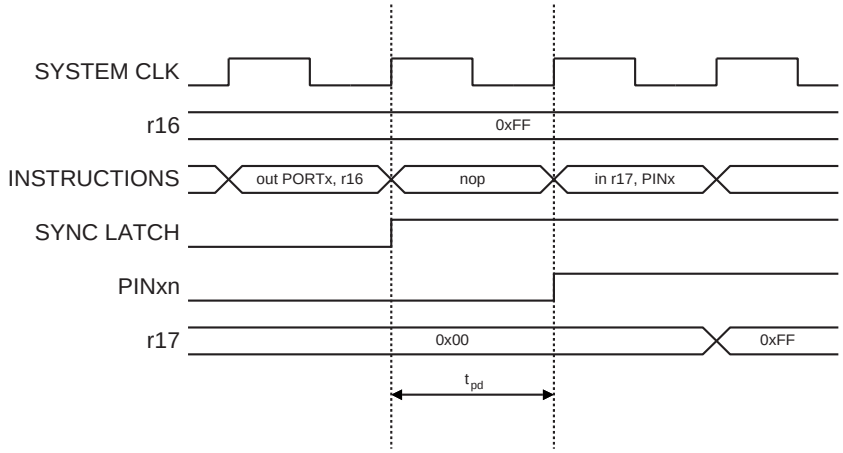
Figure 10-3. Synchronization when Reading an Externally Applied Pin value



Consider the clock period starting shortly after the first falling edge of the system clock. The latch is closed when the clock is low, and goes transparent when the clock is high, as indicated by the shaded region of the “SYNC LATCH” signal. The signal value is latched when the system clock goes low. It is clocked into the PINxn Register at the succeeding positive clock edge. As indicated by the two arrows tpd,max and tpd,min, a single signal transition on the pin will be delayed between ½ and 1½ system clock period depending upon the time of assertion.

When reading back a software assigned pin value, a nop instruction must be inserted as indicated in Figure 10-4 on page 57. The out instruction sets the “SYNC LATCH” signal at the positive edge of the clock. In this case, the delay tpd through the synchronizer is one system clock period.

Figure 10-4. Synchronization when Reading a Software Assigned Pin Value



The following code example shows how to set port A pins 0 and 1 high, 2 and 3 low, and define the port pins from 4 to 5 as input with a pull-up assigned to port pin 4. The resulting pin values are read back again, but as previously discussed, a *nop* instruction is included to be able to read back the value recently assigned to some of the pins.

Assembly Code Example

```
...
; Define pull-ups and set outputs high
; Define directions for port pins
ldi  r16,(1<<PA4)|(1<<PA1)|(1<<PA0)
ldi  r17,(1<<DDA3)|(1<<DDA2)|(1<<DDA1)|(1<<DDA0)
out  PORTA,r16
out  DDRA,r17
; Insert nop for synchronization
nop
; Read port pins
in   r16,PINA
...
```

Note: Two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1 and 4, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

C Code Example

```
unsigned char i;
  ...
  /* Define pull-ups and set outputs high */
  /* Define directions for port pins */
  PORTA = (1<<PA4)|(1<<PA1)|(1<<PA0);
  DDRA = (1<<DDA3)|(1<<DDA2)|(1<<DDA1)|(1<<DDA0);
  /* Insert nop for synchronization*/
  _NOP();
  /* Read port pins */
  i = PINA;
  ...
```

Note: See “Code Examples” on page 6.

10.1.5 Digital Input Enable and Sleep Modes

As shown in Figure 10-2 on page 55, the digital input signal can be clamped to ground at the input of the schmitt-trigger. The signal denoted SLEEP in the figure, is set by the MCU Sleep Controller in Power-down and Standby modes to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

SLEEP is overridden for port pins enabled as external interrupt pins. If the external interrupt request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in “Alternate Port Functions” on page 58.

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

10.1.6 Unconnected Pins

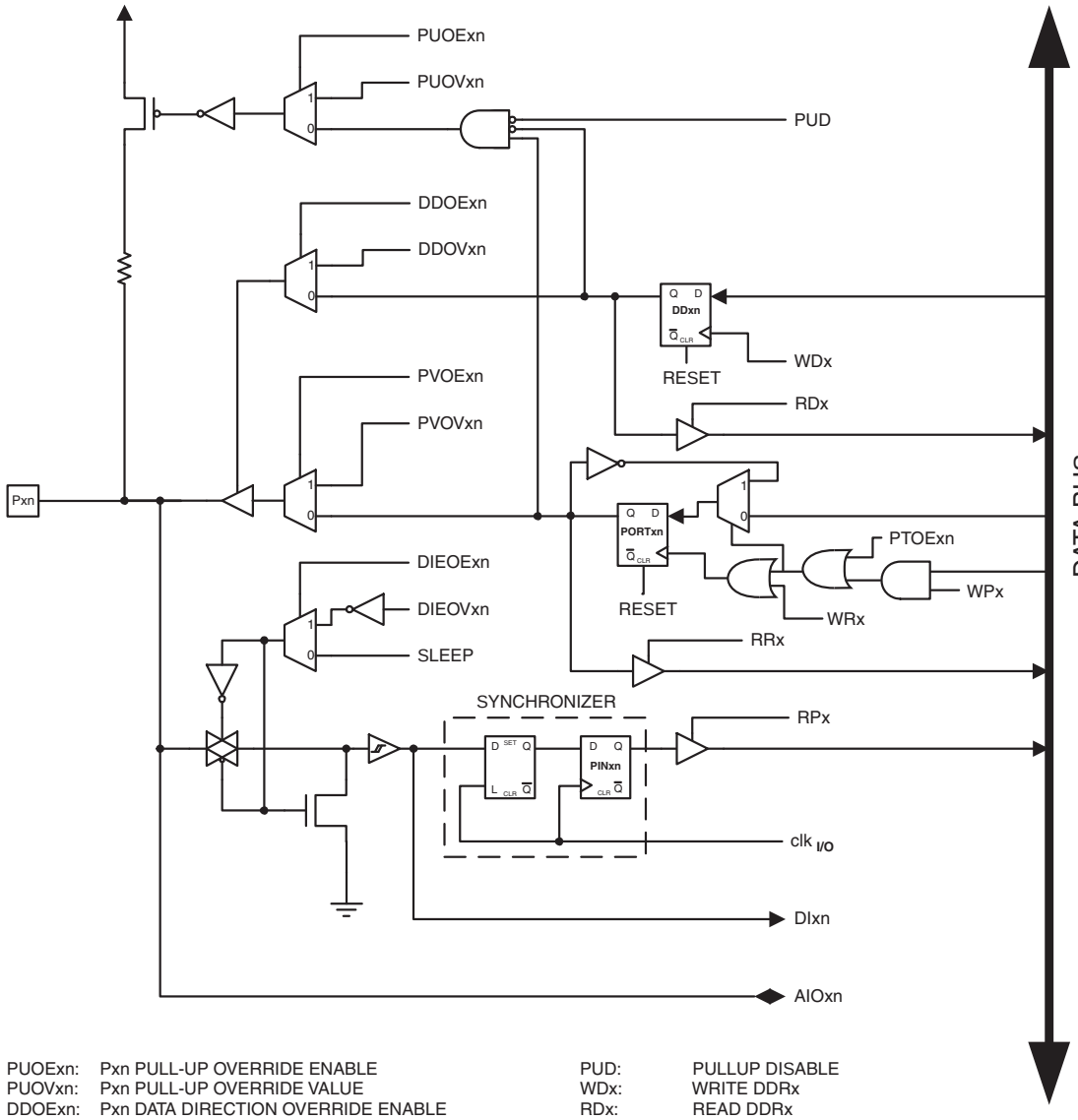
If some pins are unused, it is recommended to ensure that these pins have a defined level. Even though most of the digital inputs are disabled in the deep sleep modes as described above, floating inputs should be avoided to reduce current consumption in all other modes where the digital inputs are enabled (Reset, Active mode and Idle mode).

The simplest method to ensure a defined level of an unused pin, is to enable the internal pull-up. In this case, the pull-up will be disabled during reset. If low power consumption during reset is important, it is recommended to use an external pull-up or pulldown. Connecting unused pins directly to V_{CC} or GND is not recommended, since this may cause excessive currents if the pin is accidentally configured as an output.

10.2 Alternate Port Functions

Most port pins have alternate functions in addition to being general digital I/Os. In Figure 10-5 below is shown how the port pin control signals from the simplified Figure 10-2 on page 55 can be overridden by alternate functions.

Figure 10-5. Alternate Port Functions[1]



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE
SLEEP:	SLEEP CONTROL
PTOExn:	Pxn, PORT TOGGLE OVERRIDE ENABLE

PUD:	PULLUP DISABLE
WDx:	WRITE DDRx
RDx:	READ DDRx
RRx:	READ PORTx REGISTER
WRx:	WRITE PORTx
RPx:	READ PORTx PIN
WPx:	WRITE PINx
$clk_{I/O}$:	I/O CLOCK
DIxn:	DIGITAL INPUT PIN n ON PORTx
AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

The overriding signals may not be present in all port pins, but the figure serves as a generic description applicable to all port pins in the AVR microcontroller family.

Table 10-2 summarizes the function of the overriding signals. The pin and port indexes from Figure 10-5 are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

Table 10-2. Generic Description of Overriding Signals for Alternate Functions

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
PTOE	Port Toggle Override Enable	If PTOE is set, the PORTxn Register bit is inverted.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU state (Normal mode, sleep mode).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, sleep mode).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt-trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/Output	This is the Analog Input/Output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

10.2.1 Alternate Functions of Port A

The Port A pins with alternate function are shown in Table 10-3.

Table 10-3. Port A Pins Alternate Functions

Port Pin	Alternate Function
PA0	ADC0: ADC Input Channel 0 AREF: External Analog Reference PCINT0: Pin Change Interrupt 0, Source 0
PA1	ADC1: ADC Input Channel 1 AIN0: Analog Comparator, Positive Input PCINT1: Pin Change Interrupt 0, Source 1
PA2	ADC2: ADC Input Channel 2 AIN1: Analog Comparator, Negative Input PCINT2: Pin Change Interrupt 0, Source 2
PA3	ADC3: ADC Input Channel 3 T0: Timer/Counter0 Clock Source. PCINT3: Pin Change Interrupt 0, Source 3
PA4	ADC4: ADC Input Channel 4 USCK: USI Clock (Three Wire Mode) SCL : USI Clock (Two Wire Mode) T1: Timer/Counter1 Clock Source PCINT4: Pin Change Interrupt 0, Source 4
PA5	ADC5: ADC Input Channel 5 DO: USI Data Output (Three Wire Mode) MISO: SPI Master Data Input / Slave Data Output OC1B: Timer/Counter1 Compare Match B Output PCINT5: Pin Change Interrupt 0, Source 5
PA6	ADC6: ADC Input Channel 6 DI: USI Data Input (Three Wire Mode) SDA: USI Data Input (Two Wire Mode) MOSI: SPI Master Data Output / Slave Data Input OC1A: Timer/Counter1 Compare Match A Output PCINT6: Pin Change Interrupt 0, Source 6
PA7	ADC7: ADC Input Channel 7 OC0B: Timer/Counter0 Compare Match B Output ICP1: Timer/Counter1 Input Capture Pin PCINT7: Pin Change Interrupt 0, Source 7

- **Port A, Bit 0 – ADC0/AREF/PCINT0**
  - ADC0: Analog to Digital Converter, Channel 0.
  - AREF: External Analog Reference for ADC. Pullup and output driver are disabled on PA0 when the pin is used as an external reference or Internal Voltage Reference with external capacitor at the AREF pin by setting (one) the bit REFS0 in the ADC Multiplexer Selection Register (ADMUX).
  - PCINT0: Pin Change Interrupt source 0. The PA0 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 1 – ADC1/AIN0/PCINT1**
  - ADC1: Analog to Digital Converter, Channel 1.
  - AIN0: Analog Comparator Positive Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.
  - PCINT1: Pin Change Interrupt source 1. The PA1 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 2 – ADC2/AIN1/PCINT2**
  - ADC2: Analog to Digital Converter, Channel 2.
  - AIN1: Analog Comparator Negative Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.
  - PCINT2: Pin Change Interrupt source 2. The PA2 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 3 – ADC3/T0/PCINT3**
  - ADC3: Analog to Digital Converter, Channel 3.
  - T0: Timer/Counter0 counter source.
  - PCINT3: Pin Change Interrupt source 3. The PA3 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 4 – ADC4/USCK/SCL/T1/PCINT4**
  - ADC4: Analog to Digital Converter, Channel 4.
  - USCK: Three-wire mode Universal Serial Interface Clock.
  - SCL: Two-wire mode Serial Clock for USI Two-wire mode.
  - T1: Timer/Counter1 counter source.
  - PCINT4: Pin Change Interrupt source 4. The PA4 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 5 – ADC5/DO/MISO/OC1B/PCINT5**
  - ADC5: Analog to Digital Converter, Channel 5.
  - DO: Data Output in USI Three-wire mode. Data output (DO) overrides PORTA5 value and it is driven to the port when the data direction bit DDA5 is set (one). However the PORTA5 bit still controls the pullup, enabling pullup if direction is input and PORTA5 is set(one).
  - MISO: Master Data input, Slave Data output pin for SPI channel. When the SPI is enabled as a Master, this pin is configured as an input regardless of the setting of DDA5. When the SPI is enabled as a Slave, the data direction of this pin is controlled by DDA5. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTA5 bit.
  - OC1B: Output Compare Match output: The PA5 pin can serve as an external output for the Timer/Counter1 Compare Match B. The PA5 pin has to be configured as an output (DDA5 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.
  - PCINT5: Pin Change Interrupt source 5. The PA5 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 6 – ADC6/DI/SDA/MOSI/OC1A/PCINT6**
  - ADC6: Analog to Digital Converter, Channel 6.
  - SDA: Two-wire mode Serial Interface Data.
  - DI: Data Input in USI Three-wire mode. USI Three-wire mode does not override normal port functions, so pin must be configure as an input for DI function.
  - MOSI: Master Data output, Slave Data input for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDA6. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDA6. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTA6 bit.
  - OC1A, Output Compare Match output: The PA6 pin can serve as an external output for the Timer/Counter1 Compare Match A. The pin has to be configured as an output (DDA6 set (one)) to serve this function. This is also the output pin for the PWM mode timer function.
  - PCINT6: Pin Change Interrupt source 6. The PA6 pin can serve as an external interrupt source for pin change interrupt 0.

- **Port A, Bit 7 – ADC7/OC0B/ICP1/PCINT7**
  - ADC7: Analog to Digital Converter, Channel 7.
  - OC0B, Output Compare Match output: The PA7 pin can serve as an external output for the Timer/Counter0 Compare Match B. The pin has to be configured as an output (DDA7 set (one)) to serve this function. This is also the output pin for the PWM mode timer function.
  - ICP1, Input Capture Pin: The PA7 pin can act as an Input Capture Pin for Timer/Counter1.
  - PCINT7: Pin Change Interrupt source 7. The PA7 pin can serve as an external interrupt source for pin change interrupt 0.

Table 10-4 and Table 10-6 relate the alternate functions of Port A to the overriding signals shown in Figure 10-5 on page 59.

Table 10-4. Overriding Signals for Alternate Functions in PA7..PA5

Signal Name	PA7/ADC7/OC0B/ICP1/ PCINT7	PA6/ADC6/DI/SDA/MOSI/ OC1A/ PCINT6	PA5/ADC5/MISO/DO/ OC1B/ PCINT5
PUOE	0	0	0
PUOV	0	0	0
DDOE	0	USIWM1	0
DDOV	0	($\overline{\text{SDA}}$ + $\overline{\text{PORTA6}}$) • DDA6	0
PVOE	OC0B enable	(USIWM1 • DDA6) + OC1A enable	($\overline{\text{USIWM1}}$ • USIWM0) + OC1B enable
PVOV	OC0B	$\overline{(\text{USIWM1} \bullet \text{DDA6})}$ • OC1A	$\overline{\text{USIWM1}}$ • USIWM0 • DO + (USIWM1 + $\overline{\text{USIWM0}}$) • OC1B
PTOE	0	0	0
DIEOE	PCINT7 • PCIE0 + ADC7D	USISIE + (PCINT6 • PCIE0) + ADC6D	PCINT5 • PCIE + ADC5D
DIEOV	PCINT7 • PCIE0	USISIE + PCINT7 • PCIE0	PCINT5 • PCIE
DI	PCINT7/ICP1 Input	DI/SDA/PCINT6 Input	PCINT5 Input
AIO	ADC7 Input	ADC6 Input	ADC5 Input

Table 10-5. Overriding Signals for Alternate Functions in PA4..PA2

Signal Name	PA4/ADC4/USCK/SCL/T1/ PCINT4	PA3/ADC3/T0/PCINT3	PA2/ADC2/AIN1/PCINT2
PUOE	0	0	0
PUOV	0	0	0
DDOE	USIWM1	0	0
DDOV	USI_SCL_HOLD + $\overline{\text{PORTA4}}$) • DDA4	0	0
PVOE	USIWM1 • DDA4	0	0
PVOV	0	0	0
PTOE	USI_PTOE	0	0
DIEOE	USISIE + (PCINT4 • PCIE0) + ADC4D	(PCINT3 • PCIE0) + ADC3D	PCINT2 • PCIE + ADC2D
DIEOV	USISIE + (PCINT4 • PCIE0)	PCINT3 • PCIE0	PCINT3 • PCIE0
DI	USCK/SCL/T1/PCINT4 input	PCINT1 Input	PCINT0 Input
AIO	ADC4 Input	ADC3 Input	ADC2/Analog Comparator Negative Input

Table 10-6. Overriding Signals for Alternate Functions in PA1..PA0

Signal Name	PA1/ADC1/AIN0/PCINT1	PA0/ADC0/AREF/PCINT0
PUOE	0	$\overline{\text{RESET}}$ • ($\overline{\text{REFS1}}$ • REFS0 + REFS1 • REFS0)
PUOV	0	0
DDOE	0	$\overline{\text{RESET}}$ • ($\overline{\text{REFS1}}$ • REFS0 + REFS1 • REFS0)
DDOV	0	0
PVOE	0	$\overline{\text{RESET}}$ • ($\overline{\text{REFS1}}$ • 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

10.2.2 Alternate Functions of Port B

The Port B pins with alternate function are shown in Table 10-7.

Table 10-7. Port B Pins Alternate Functions

Port Pin	Alternate Function
PB0	XTAL1: Crystal Oscillator Input PCINT8: Pin Change Interrupt 1, Source 8 CLKI: External Clock Input
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	$\overline{\text{RESET}}$: 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 internal calibrateble RC oscillator. When used as a clock pin, the pin can not be used as an I/O pin. When using internal calibratable 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.
  - CLKI: Clock Input from an external clock source, see “External Clock” on page 26.

- **Port B, Bit 1 – XTAL2/PCINT9**
  - XTAL2: Chip Clock Oscillator pin 2. Used as clock pin for all chip clock sources except internal calibrateble RC Oscillator and external clock. When used as a clock pin, the pin can not be used as an I/O pin. When using internal calibratable 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.

- **Port B, Bit 2 – INT0/OC0A/CKOUT/PCINT10**
  - INT0: External Interrupt Request 0.
  - OC0A: Output Compare Match output: The PB2 pin can serve as an external output for the Timer/Counter0 Compare Match A. The PB2 pin has to be configured as an output (DDB2 set (one)) to serve this function. The OC0A pin is also the output pin for the PWM mode timer function.
  - CKOUT - System Clock Output: The system clock can be output on the PB2 pin. The system clock will be output if the CKOUT Fuse is programmed, regardless of the PORTB2 and DDB2 settings. It will also be output during reset.
  - PCINT10: Pin Change Interrupt source 10. The PB2 pin can serve as an external interrupt source for pin change interrupt 1.

- **Port B, Bit 3 – $\overline{\text{RESET}}$/dW/PCINT11**
  - $\overline{\text{RESET}}$: External $\overline{\text{Reset}}$ input is active low and enabled by unprogramming (“1”) the RSTDISBL Fuse. Pullup is activated and output driver and digital input are deactivated when the pin is used as the $\overline{\text{RESET}}$ pin.
  - dW: When the debugWIRE Enable (DWEN) Fuse is programmed and Lock bits are unprogrammed, the debugWIRE system within the target device is activated. The RESET port pin is configured as a wire-AND (open-drain) bi-directional I/O pin with pull-up enabled and becomes the communication gateway between target and emulator.
  - PCINT11: Pin Change Interrupt source 11. The PB3 pin can serve as an external interrupt source for pin change interrupt 1.

Table 10-8 on page 66 and Table 10-9 on page 67 relate the alternate functions of Port B to the overriding signals shown in Figure 10-5 on page 59.

Table 10-8. Overriding Signals for Alternate Functions in PB3..PB2

Signal Name	PB3/$\overline{\text{RESET}}$/dW/ PCINT11	PB2/INT0/OC0A/CKOUT/PCINT10
PUOE	$\overline{\text{RSTDISBL}}$ [1] + DEBUGWIRE_ENABLE [2]	CKOUT
PUOV	1	0
DDOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2]	CKOUT
DDOV	DEBUGWIRE_ENABLE[2] • $\overline{\text{debugWire Transmit}}$	1
PVOE	$\overline{\text{RSTDISBL}}$[1] + DEBUGWIRE_ENABLE[2]	CKOUT + OC0A enable
PVOV	0	CKOUT • System Clock + $\overline{\text{CKOUT}}$ • OC0A
PTOE	0	0
DIEOE	RSTDISBL[1] + DEBUGWIRE_ENABLE[2] + PCINT11 • PCIE1	PCINT10 • PCIE1 + INT0
DIEOV	DEBUGWIRE_ENABLE[2] + (RSTDISBL[1] • PCINT11 • PCIE1)	PCINT10 • PCIE1 + INT0
DI	dW/PCINT11 Input	INT0/PCINT10 Input
AIO		

1. RSTDISBL is 1 when the Fuse is “0” (Programmed).
2. DebugWIRE is enabled wheb DWEN Fuse is programmed and Lock bits are unprogrammed.

Table 10-9. Overriding Signals for Alternate Functions in PB1..PB0

Signal Name	PB1/XTAL2/PCINT9	PB0/XTAL1/PCINT8
PUOE	EXT_OSC (1)	EXT_CLOCK (2) + EXT_OSC(1)
PUOV	0	0
DDOE	EXT_OSC(1)	EXT_CLOCK(2) + EXT_OSC(1)
DDOV	0	0
PVOE	EXT_OSC(1)	EXT_CLOCK(2) + EXT_OSC(1)
PVOV	0	0
PTOE	0	0
DIEOE	EXT_OSC (1)+ PCINT9 • PCIE1	EXT_CLOCK(2) + EXT_OSC(1) + (PCINT8 • PCIE1)
DIEOV	$\overline{\text{EXT\_OSC}}$(1) • PCINT9 • PCIE1	(EXT_CLOCK(2) • $\overline{\text{PWR\_DOWN}}$) + ($\overline{\text{EXT\_CLOCK}}$(2) • $\overline{\text{EXT\_OSC}}$(1) • PCINT8 • PCIE1)
DI	PCINT9 Input	CLOCK/PCINT8 Input
AIO	XTAL2	XTAL1

1. EXT_OSC = crystal oscillator or low frequency crystal oscillator is selected as system clock.
2. EXT_CLOCK = external clock is selected as system clock.

10.3 Register Description

10.3.1 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
	BODS	PUD	SE	SM1	SM0	BODSE	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 6 – PUD: Pull-up Disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn Registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See “Configuring the Pin” on page 55 for more details about this feature.

10.3.2 PORTA – Port A Data Register

Bit	7	6	5	4	3	2	1	0	
0x1B (0x3B)	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
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	

10.3.3 DDRA – Port A Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x1A (0x3A)	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRB
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	

10.3.4 PINA – Port A Input Pins

Bit	7	6	5	4	3	2	1	0	
0x19 (0x39)	**PINA7**	**PINA6**	**PINA5**	**PINA4**	**PINA3**	**PINA2**	**PINA1**	**PINA0**	**PINB**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

10.3.5 PORTB – Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x18 (0x38)	–	–	–	–	**PORTB3**	**PORTB2**	**PORTB1**	**PORTB0**	**PORTB**
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.6 DDRB – Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x17 (0x37)	–	–	–	–	**DDB3**	**DDB2**	**DDB1**	**DDB0**	**DDRB**
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

10.3.7 PINB – Port B Input Pins

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	–	–	**PINB3**	**PINB2**	**PINB1**	**PINB0**	**PINB**
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	N/A	N/A	N/A	N/A	

11. 8-bit Timer/Counter0 with PWM

11.1 Features

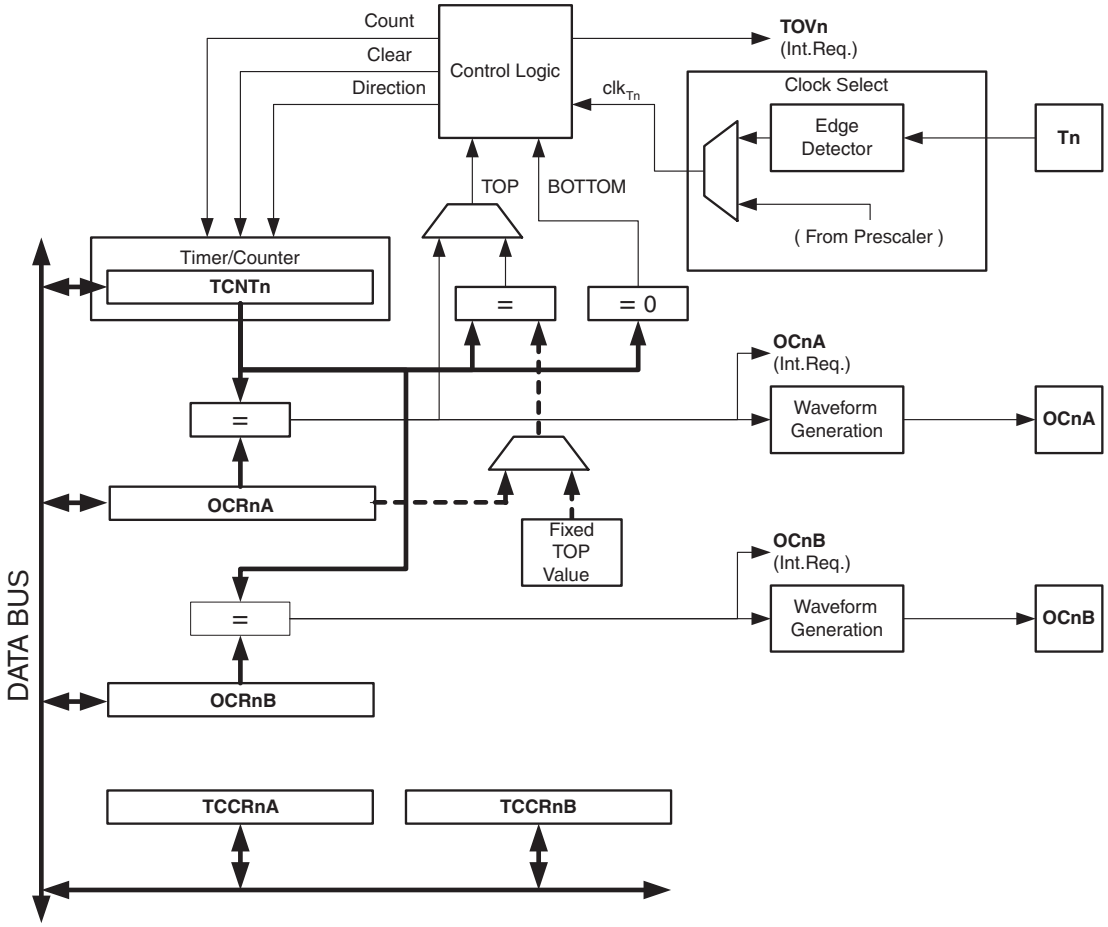
- **Two Independent Output Compare Units**
- **Double Buffered Output Compare Registers**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch Free, Phase Correct Pulse Width Modulator (PWM)**
- **Variable PWM Period**
- **Frequency Generator**
- **Three Independent Interrupt Sources (TOV0, OCF0A, and OCF0B)**

11.2 Overview

Timer/Counter0 is a general purpose 8-bit Timer/Counter module, with two independent Output 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 11-1 on page 69. For the actual placement of I/O pins, refer to Figure 1-1 on page 2. 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 "Register Description" on page 80.

Figure 11-1. 8-bit Timer/Counter Block Diagram



11.2.1 Registers

The Timer/Counter (TCNT0) and Output Compare Registers (OCR0A and OCR0B) are 8-bit registers. Interrupt request (abbreviated to Int.Req. in Figure 11-1) signals are all visible in the Timer Interrupt Flag Register (TIFR0). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK0). TIFR0 and TIMSK0 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T0 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{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 “Output Compare Unit” on page 71 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.

11.2.2 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/Counter0 counter value and so on.

The definitions in Table 11-1 are also used extensively throughout the document.

Table 11-1. Definitions

Constant	Description
BOTTOM	The counter reaches 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 depends on the mode of operation

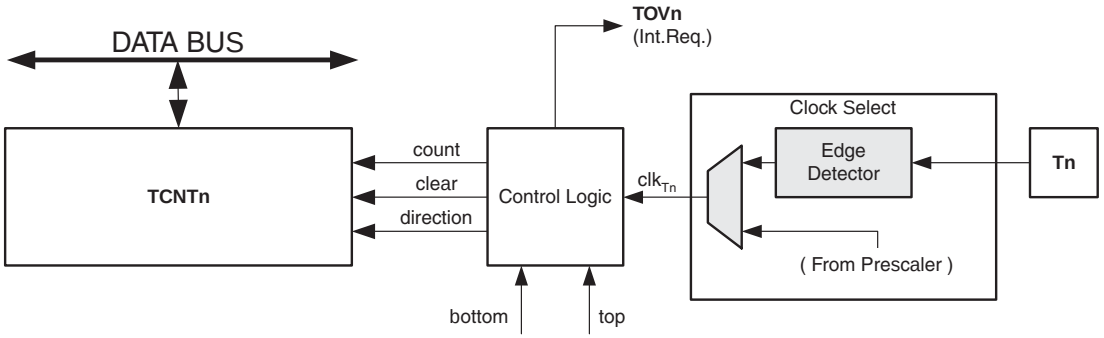
11.3 Clock Sources

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 “Timer/Counter Prescaler” on page 115.

11.4 Counter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 11-2 on page 71 shows a block diagram of the counter and its surroundings.

Figure 11-2. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT0 by 1.
direction	Select between increment and decrement.
clear	Clear TCNT0 (set all bits to zero).
clk_{Tn}	Timer/Counter clock, referred to as clk_{T0} in the following.
top	Signalize that TCNT0 has reached maximum value.
bottom	Signalize that TCNT0 has reached minimum value (zero).

Depending of the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clk_{T0}). clk_{T0} can be generated from an external or internal clock source, selected by the Clock Select bits (CS02:0). When no clock source is selected (CS02:0 = 0) the timer is stopped. However, the TCNT0 value can be accessed by the CPU, regardless of whether clk_{T0} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the WGM01 and WGM00 bits located in the Timer/Counter Control Register (TCCR0A) and the WGM02 bit located in the Timer/Counter Control Register B (TCCR0B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare output OC0A. For more details about advanced counting sequences and waveform generation, see “Modes of Operation” on page 74.

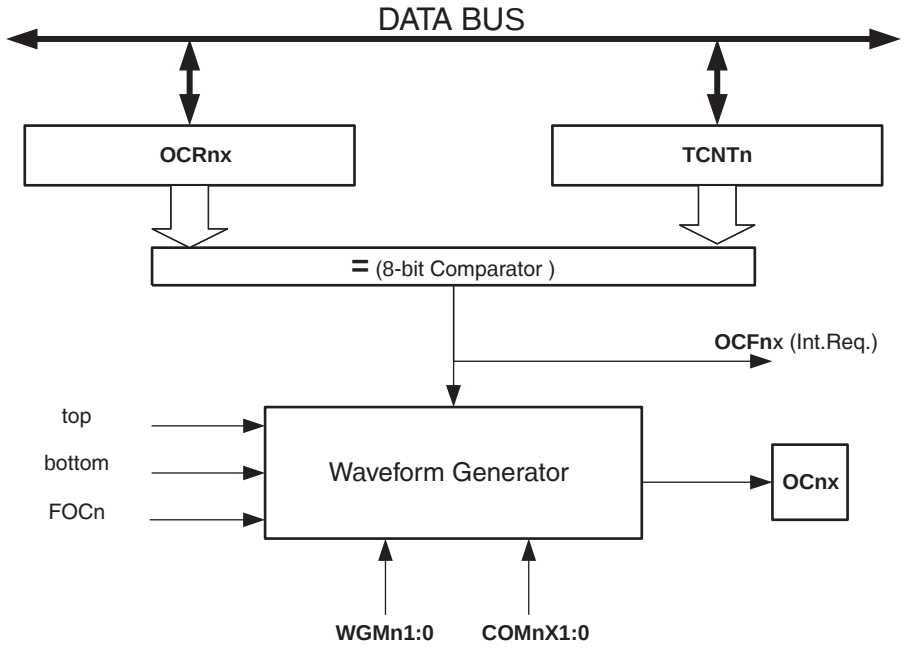
The Timer/Counter Overflow Flag (TOV0) is set according to the mode of operation selected by the WGM01:0 bits. TOV0 can be used for generating a CPU interrupt.

11.5 Output Compare Unit

The 8-bit comparator continuously compares TCNT0 with the Output Compare Registers (OCR0A and OCR0B). Whenever TCNT0 equals OCR0A or OCR0B, the comparator signals a match. A match will set the Output Compare Flag (OCF0A or OCF0B) at the next timer clock cycle. If the corresponding interrupt is enabled, the Output Compare Flag generates an Output Compare interrupt. The Output Compare Flag is automatically cleared when the interrupt is executed. Alternatively, the flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the WGM02:0 bits and Compare Output mode (COM0x1:0) bits. The max and bottom signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation. See “Modes of Operation” on page 74.

Figure 11-3 shows a block diagram of the Output Compare unit.

Figure 11-3. Output Compare Unit, Block Diagram



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 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.

11.5.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a 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).

11.5.2 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.

11.5.3 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 when 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.

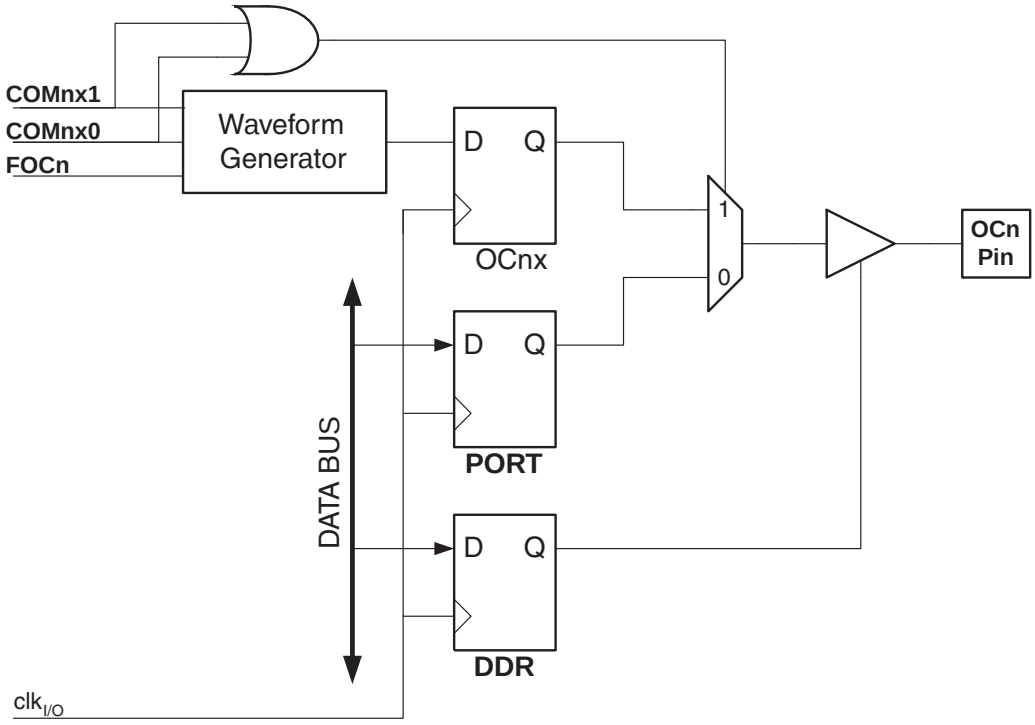
The setup of the OC0x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC0x value is to use the Force Output Compare (0x) strobe bits in Normal mode. The OC0x Registers keep their values even when changing between Waveform Generation modes.

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

11.6 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 11-4 on page 73 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 occur, the OC0x Register is reset to “0”.

Figure 11-4. 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 “Register Description” on page 80

11.6.1 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 11-2 on page 80. For fast PWM mode, refer to Table 11-3 on page 81, and for phase correct PWM refer to Table 11-4 on page 81.

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

11.7 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 “Modes of Operation” on page 74).

For detailed timing information refer to Figure 11-8 on page 79, Figure 11-9 on page 79, Figure 11-10 on page 79 and Figure 11-11 on page 80 in “Timer/Counter Timing Diagrams” on page 78.

11.7.1 Normal Mode

The simplest mode of operation is the Normal mode (WGM02:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the Timer/Counter Overflow Flag (TOV0) will be set in the same timer clock cycle as the TCNT0 becomes zero. The TOV0 Flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV0 Flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

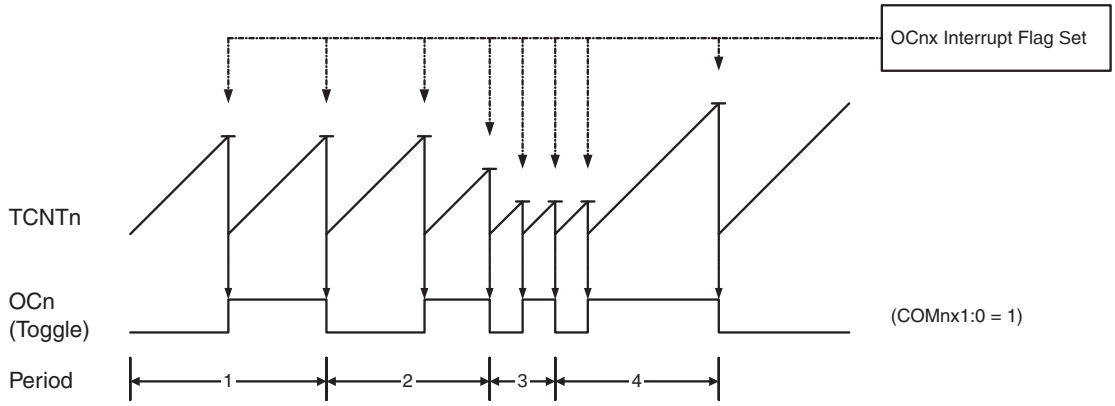
The Output Compare Unit can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

11.7.2 Clear Timer on Compare Match (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM02:0 = 2), the OCR0A Register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT0) matches the OCR0A. The OCR0A defines the top value for the counter, hence also its resolution. This mode allows greater control of the Compare Match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 11-5 on page 75. The counter value (TCNT0) increases until a Compare Match occurs between TCNT0 and OCR0A, and then counter (TCNT0) is cleared.

Figure 11-5. CTC Mode, Timing Diagram



An interrupt can be generated each time the counter value reaches the TOP value by using the OCF0A Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR0A is lower than the current value of TCNT0, the counter will miss the Compare Match. The counter will then have to count to its maximum value (0xFF) and wrap around starting at 0x00 before the Compare Match can occur.

For generating a waveform output in CTC mode, the OC0A output can be set to toggle its logical level on each Compare Match by setting the Compare Output mode bits to toggle mode (COM0A1:0 = 1). The OC0A value will not be visible on the port pin unless the data direction for the pin is set to output. The waveform generated will have a maximum frequency of $_0 = f_{clk\_I/O}/2$ when OCR0A is set to zero (0x00). The waveform frequency is defined by the following equation:

$$f_{OCnx} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnx)}$$

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

As for the Normal mode of operation, the TOV0 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x00.

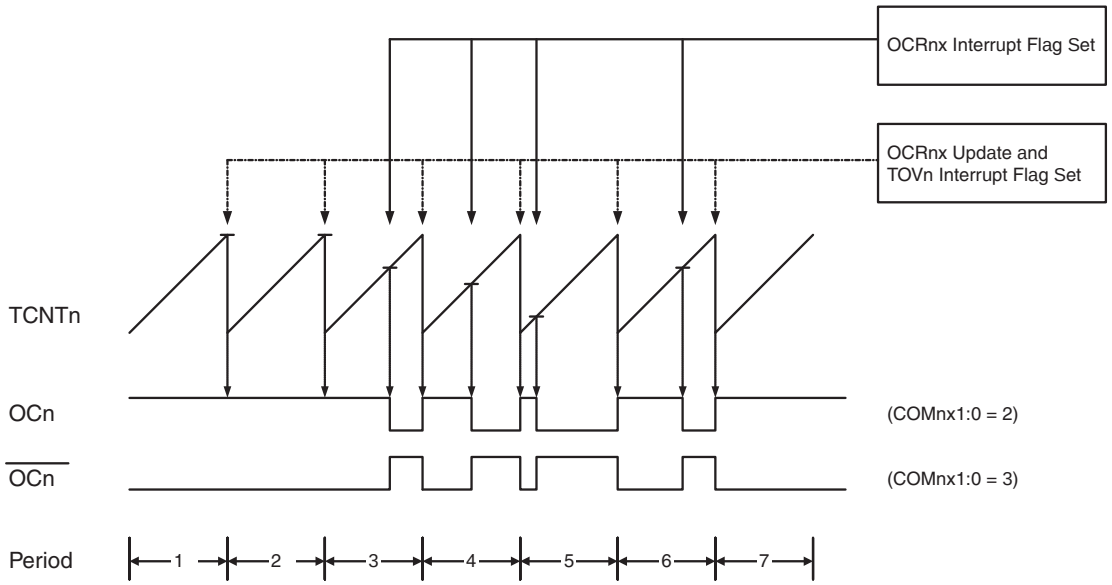
11.7.3 Fast PWM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM02:0 = 3 or 7) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. TOP is defined as 0xFF when WGM2:0 = 3, and OCR0A when WGM2:0 = 7. In non-inverting Compare Output mode, the Output Compare (OC0x) is cleared on the Compare Match between TCNT0 and OCR0x, and set at BOTTOM. In inverting Compare Output mode, the output is set on Compare Match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct PWM mode that use dual-slope operation. This high frequency makes the fast PWM mode well suited

for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the TOP value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 11-6 on page 76. The TCNT0 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT0 slopes represent Compare Matches between OCR0x and TCNT0.

Figure 11-6. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches TOP. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC0x pins. Setting the COM0x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM0x1:0 to three: Setting the COM0A1:0 bits to one allowes the AC0A pin to toggle on Compare Matches if the WGM02 bit is set. This option is not available for the OC0B pin (See Table 11-3 on page 81). The actual OC0x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC0x Register at the Compare Match between OCR0x and TCNT0, and clearing (or setting) the OC0x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

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

$$f_{OCnxPWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

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

The extreme values for the OCR0A Register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0A is set equal to BOTTOM, the output 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.)

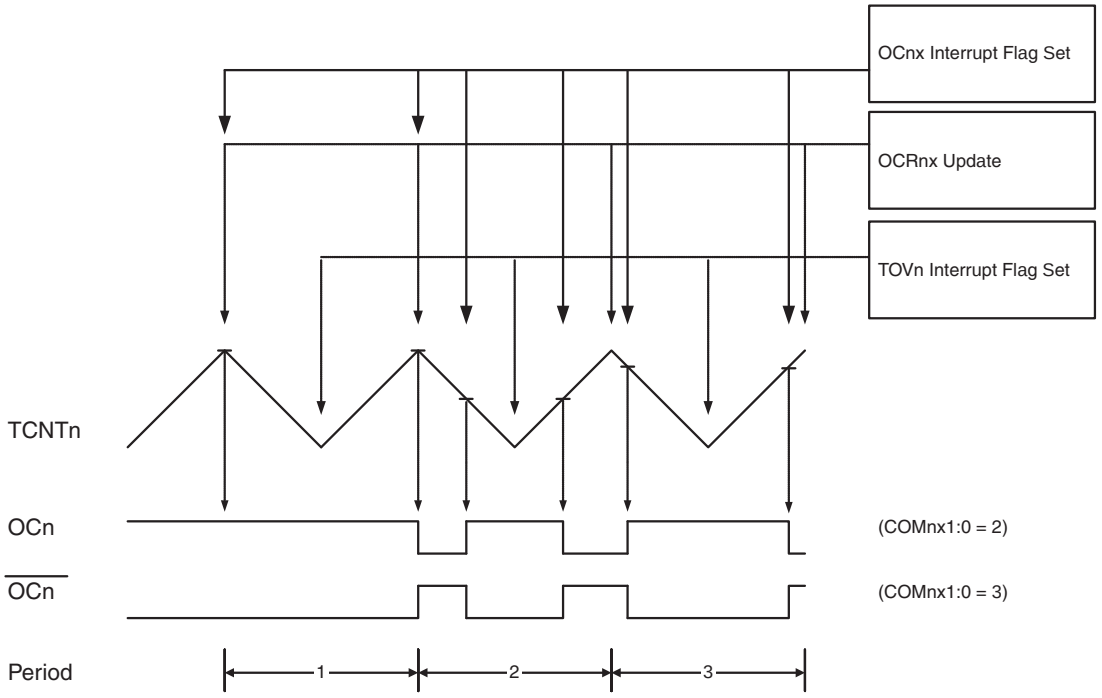
A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC0x to toggle its logical level on each Compare Match (COM0x1:0 = 1). The waveform generated will have a maximum frequency of $_0 = f_{clk\_I/O}/2$ when OCR0A is set to zero. This feature is similar to the OC0A toggle in CTC mode, except the double buffer feature of the Output Compare unit is enabled in the fast PWM mode.

11.7.4 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 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 upcounting, 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 11-7 on page 77. The TCNT0 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT0 slopes represent Compare Matches between OCR0x and TCNT0.

Figure 11-7. 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.

In phase correct PWM mode, the compare unit allows generation of PWM waveforms on the OC0x pins. Setting the COM0x1:0 bits to two will produce a non-inverted PWM. An inverted PWM output can be generated by setting the COM0x1:0 to three: Setting the COM0A0 bits to one allows the OC0A pin to toggle on Compare Matches if the WGM02 bit is set. This option is not available for the OC0B pin (See Table 11-4 on page 81). The actual OC0x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by clearing (or setting) the OC0x Register at the Compare Match between OCR0x and TCNT0 when the counter increments, and setting (or clearing) the OC0x Register at Compare Match between OCR0x and TCNT0 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

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

The extreme values for the OCR0A Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR0A is set equal to BOTTOM, the output will be continuously low and if set equal to MAX the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

At the very start of period 2 in Figure 11-7 on page 77 OCn has a transition from high to low even though there is no Compare Match. The point of this transition is to guaratee symmetry around BOTTOM. There are two cases that give a transition without Compare Match.

- OCR0A changes its value from MAX, like in Figure 11-7 on page 77. When the OCR0A value is MAX the OCn pin value is the same as the result of a down-counting Compare Match. To ensure symmetry around BOTTOM the OCn value at MAX must correspond to the result of an up-counting Compare Match.
- The timer starts counting from a value higher than the one in OCR0A, and for that reason misses the Compare Match and hence the OCn change that would have happened on the way up.

11.8 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T0}) is therefore shown as a clock enable signal in the following figures. The figures include information on when Interrupt Flags are set. Figure 11-8 on page 79 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value in all modes other than phase correct PWM mode.

Figure 11-8. Timer/Counter Timing Diagram, no Prescaling

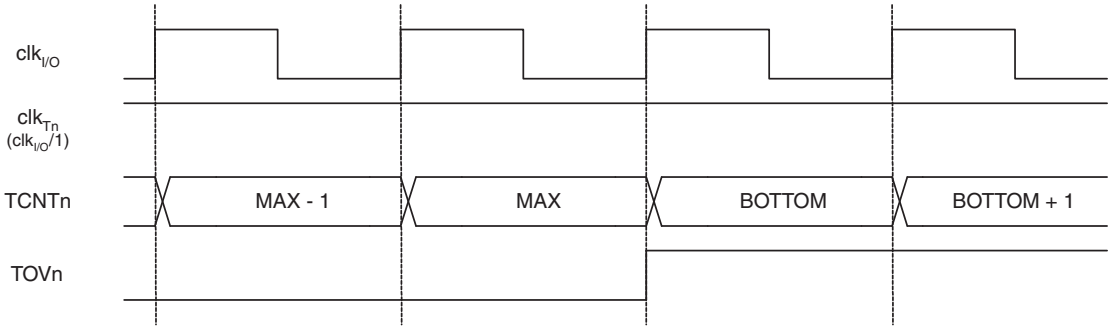


Figure 11-9 on page 79 shows the same timing data, but with the prescaler enabled.

Figure 11-9. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)

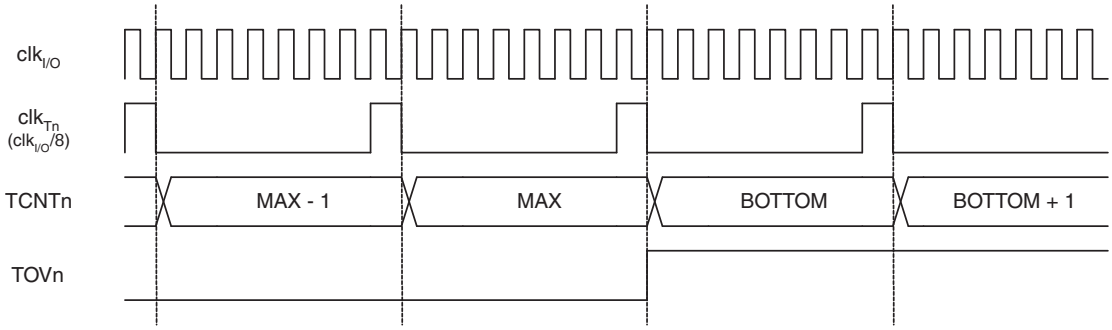


Figure 11-10 on page 79 shows the setting of OCF0B in all modes and OCF0A in all modes except CTC mode and PWM mode, where OCR0A is TOP.

Figure 11-10. Timer/Counter Timing Diagram, Setting of OCF0x, with Prescaler ($f_{clk\_I/O}/8$)

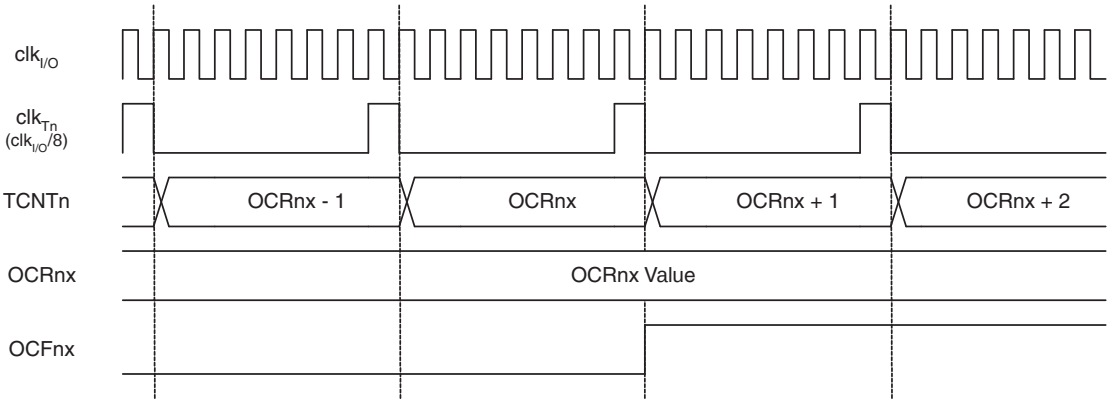
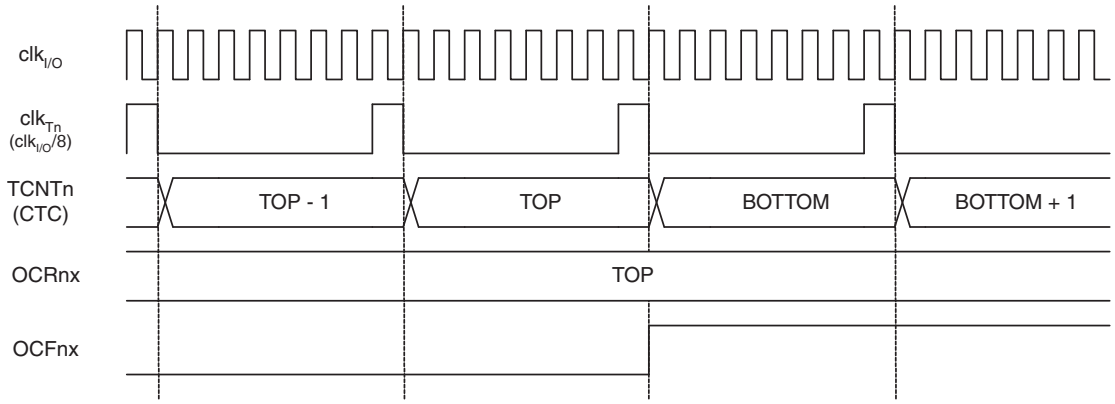


Figure 11-11 on page 80 shows the setting of OCF0A and the clearing of TCNT0 in CTC mode and fast PWM mode where OCR0A is TOP.

Figure 11-11. Timer/Counter Timing Diagram, Clear Timer on Compare Match mode, with Prescaler ($f_{clk\_I/O}/8$)



11.9 Register Description

11.9.1 TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x30 (0x50)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	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 11-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 11-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	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 11-3 shows COM0A1:0 bit functionality when WGM01:0 bits are set to fast PWM mode.

Table 11-3. Compare Output Mode, Fast 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 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 “Fast PWM Mode” on page 75 for more details.

Table 11-4 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 11-4. 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 “Phase Correct PWM Mode” on page 77 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 11-5 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 11-5. Compare Output Mode, non-PWM Mode

COM0B1	COM0B0	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 11-6 shows COM0B1:0 bit functionality when WGM02:0 bits are set to fast PWM mode.

Table 11-6. Compare Output Mode, Fast PWM Mode[1]

COM0B1	COM0B0	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 "Fast PWM Mode" on page 75 for more details.

Table 11-7 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 11-7. Compare Output Mode, Phase Correct PWM Mode[1]

COM0B1	COM0B0	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 "Phase Correct PWM Mode" on page 77 for more details.

- **Bits 3, 2 – Res: Reserved Bits**

These bits are reserved and will always read 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 11-8. 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 "Modes of Operation" on page 74).

Table 11-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Note: 1. MAX = 0xFF
BOTTOM = 0x00

11.9.2 TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x33 (0x53)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC0A: Force Output Compare A**

The FOC0A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0A output is changed according to its COM0A1:0 bits setting. Note that the FOC0A bit is implemented as a strobe. Therefore it is the value present in the COM0A1:0 bits that determines the effect of the forced compare.

A FOC0A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0A as TOP.

The FOC0A bit is always read as zero.

- **Bit 6 – FOC0B: Force Output Compare B**

The FOC0B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0B output is changed according to its COM0B1:0 bits setting. Note that the FOC0B bit is implemented as a strobe. Therefore it is the value present in the COM0B1:0 bits that determines the effect of the forced compare.

A FOC0B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0B as TOP.

The FOC0B bit is always read as zero.

- **Bits 5:4 – Res: Reserved Bits**

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

- **Bit 3 – WGM02: Waveform Generation Mode**

See the description in the “TCCR0A – Timer/Counter Control Register A” on page 80.

- **Bits 2:0 – CS02:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter.

Table 11-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

11.9.3 TCNT0 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
0x32 (0x52)	TCNT0[7:0]								TCNT0
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 Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a Compare Match between TCNT0 and the OCR0x Registers.

11.9.4 OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x36 (0x56)	OCR0A[7:0]								OCR0A
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 Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

11.9.5 OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x3C (0x5C)	OCR0B[7:0]								OCR0B
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 Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0B pin.

11.9.6 TIMSK0 – Timer/Counter 0 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x39 (0x59)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

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

- **Bit 2– OCIE0B: Timer/Counter Output Compare Match B Interrupt Enable**

When the OCIE0B bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter Compare Match B interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter occurs, i.e., when the OCF0B bit is set in the Timer/Counter Interrupt Flag Register – TIFR0.

- **Bit 1– OCIE0A: Timer/Counter0 Output Compare Match A Interrupt Enable**

When the OCIE0A bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter0 occurs, i.e., when the OCF0A bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

- **Bit 0– TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

11.9.7 TIFR0 – Timer/Counter 0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x38 (0x58)	–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

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

- **Bit 2– OCF0B: Output Compare Flag 0 B**

The OCF0B bit is set when a Compare Match occurs between the Timer/Counter and the data in OCR0B – Output Compare Register0 B. OCF0B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0B is cleared by writing a logic one to

the flag. When the I-bit in SREG, OCIE0B (Timer/Counter Compare B Match Interrupt Enable), and OCF0B are set, the Timer/Counter Compare Match Interrupt is executed.

- **Bit 1– OCF0A: Output Compare Flag 0 A**

The OCF0A bit is set when a Compare Match occurs between the Timer/Counter0 and the data in OCR0A – Output Compare Register0. OCF0A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0A (Timer/Counter0 Compare Match Interrupt Enable), and OCF0A are set, the Timer/Counter0 Compare Match Interrupt is executed.

- **Bit 0– TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set, the Timer/Counter0 Overflow interrupt is executed.

The setting of this flag is dependent of the WGM02:0 bit setting. See Table 11-8 on page 83 and “Waveform Generation Mode Bit Description” on page 83.

12. 16-bit Timer/Counter1

12.1 Features

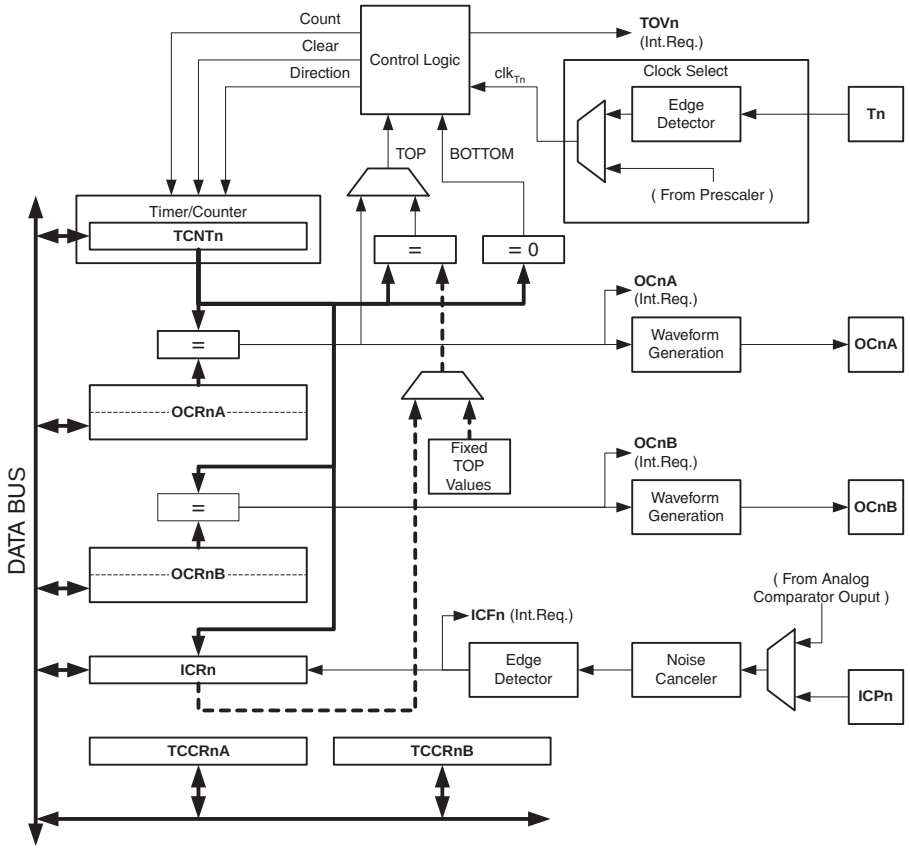
- **True 16-bit Design (i.e., Allows 16-bit PWM)**
- **Two independent Output Compare Units**
- **Double Buffered Output Compare Registers**
- **One Input Capture Unit**
- **Input Capture Noise Canceler**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch-free, Phase Correct Pulse Width Modulator (PWM)**
- **Variable PWM Period**
- **Frequency Generator**
- **External Event Counter**
- **Four independent interrupt Sources (TOV1, OCF1A, OCF1B, and ICF1)**

12.2 Overview

The 16-bit Timer/Counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement.

A simplified block diagram of the 16-bit Timer/Counter is shown in Figure 12-1 on page 87. For actual placement of I/O pins, refer to "Pinout ATtiny24/44/84" on page 2. 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 "Register Description" on page 108.

Figure 12-1. 16-bit Timer/Counter Block Diagram



Most register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, and a lower case “x” replaces the Output Compare unit channel. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT1 for accessing Timer/Counter1 counter value and so on.

12.2.1 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 “Accessing 16-bit Registers” on page 105. The Timer/Counter Control Registers (TCCR1A/B) are 8-bit registers and have no CPU access restrictions. Interrupt requests (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 time. 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 “Output Compare Units” on page 92. 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 Input Capture Register can capture the Timer/Counter value at a given external (edge triggered) event on either the Input Capture pin (ICP1) or on the Analog Comparator pins (See “Analog Comparator” on page 129). The Input Capture unit includes a digital filtering unit (Noise Canceler) for reducing the chance of capturing noise spikes.

The TOP value, or maximum Timer/Counter value, can in some modes of operation be defined by either the OCR1A Register, the ICR1 Register, or by a set of fixed values. When using OCR1A as TOP value in a PWM mode, the OCR1A Register can not be used for generating a PWM output. However, the TOP value will in this case be double buffered allowing the TOP value to be changed in run time. If a fixed TOP value is required, the ICR1 Register can be used as an alternative, freeing the OCR1A to be used as PWM output.

12.2.2 Definitions

The following definitions are used extensively throughout the section:

Table 12-1. Definitions

Constant	Description
BOTTOM	The counter reaches 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 depends on the mode of operation

12.2.3 Compatibility

The 16-bit Timer/Counter has been updated and improved from previous versions of 16-bit AVR Timer/Counters. This 16-bit Timer/Counter is fully compatible with the earlier version regarding:

- All 16-bit Timer/Counter related I/O Register address locations, including Timer Interrupt Registers.
- Bit locations inside all 16-bit Timer/Counter Registers, including Timer Interrupt Registers.
- Interrupt Vectors.

The following control bits have changed name, but have same functionality and register location:

- PWM10 is changed to WGM10.
- PWM11 is changed to WGM11.
- CTC1 is changed to WGM12.

The following bits are added to the 16-bit Timer/Counter Control Registers:

- 1A and 1B are added to TCCR1A.
- WGM13 is added to TCCR1B.

The 16-bit Timer/Counter has improvements that will affect backward compatibility in some special cases.

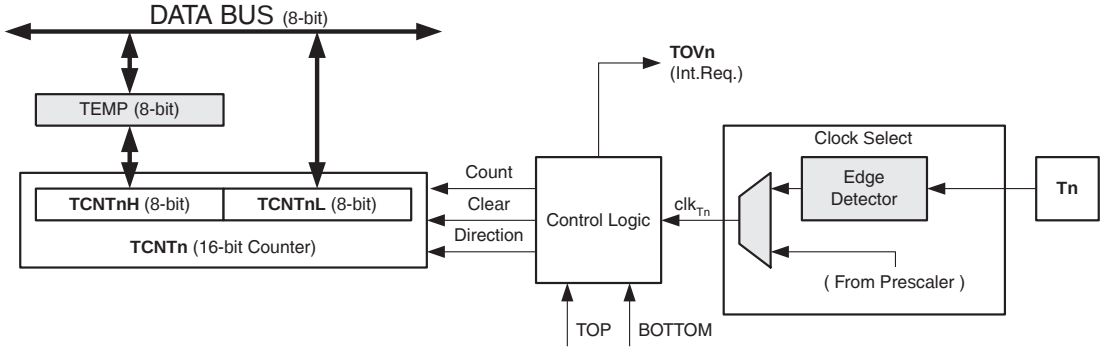
12.3 Timer/Counter Clock Sources

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 (CS12:0) bits located in the Timer/Counter control Register B (TCCR1B). For details on clock sources and prescaler, see “Timer/Counter Prescaler” on page 115.

12.4 Counter Unit

The main part of the 16-bit Timer/Counter is the programmable 16-bit bi-directional counter unit. Figure 12-2 on page 89 shows a block diagram of the counter and its surroundings.

Figure 12-2. Counter Unit Block Diagram



Signal description (internal signals):

Count	Increment or decrement TCNT1 by 1.
Direction	Select between increment and decrement.
Clear	Clear TCNT1 (set all bits to zero).
clk_{T1}	Timer/Counter clock.
TOP	Signalize that TCNT1 has reached maximum value.
BOTTOM	Signalize that TCNT1 has reached minimum value (zero).

The 16-bit counter is mapped into two 8-bit I/O memory locations: *Counter High* (TCNT1H) containing the upper eight bits of the counter, and Counter Low (TCNT1L) containing the lower eight bits. The TCNT1H Register can only be indirectly accessed by the CPU. When the CPU does an access to the TCNT1H I/O location, the CPU accesses the high byte temporary register (TEMP). The temporary register is updated with the TCNT1H value when the TCNT1L is read, and TCNT1H is updated with the temporary register value when TCNT1L is written. This allows the CPU to read or write the entire 16-bit counter value within one clock cycle via the 8-bit data bus. It is important to notice that there are special cases of writing to the TCNT1 Register when the counter is counting that will give unpredictable results. The special cases are described in the sections where they are of importance.

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clk_{T1}). The clk_{T1} can be generated from an external or internal clock source, selected by the Clock Select bits (CS12:0). When no clock source is selected (CS12:0 = 0) the timer is stopped. However, the TCNT1 value can be accessed by the CPU, independent of whether clk_{T1} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the Waveform Generation mode bits (WGM13:0) located in the Timer/Counter Control Registers A and B (TCCR1A and TCCR1B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare outputs OC1x. For more details about advanced counting sequences and waveform generation, see “Modes of Operation” on page 96.

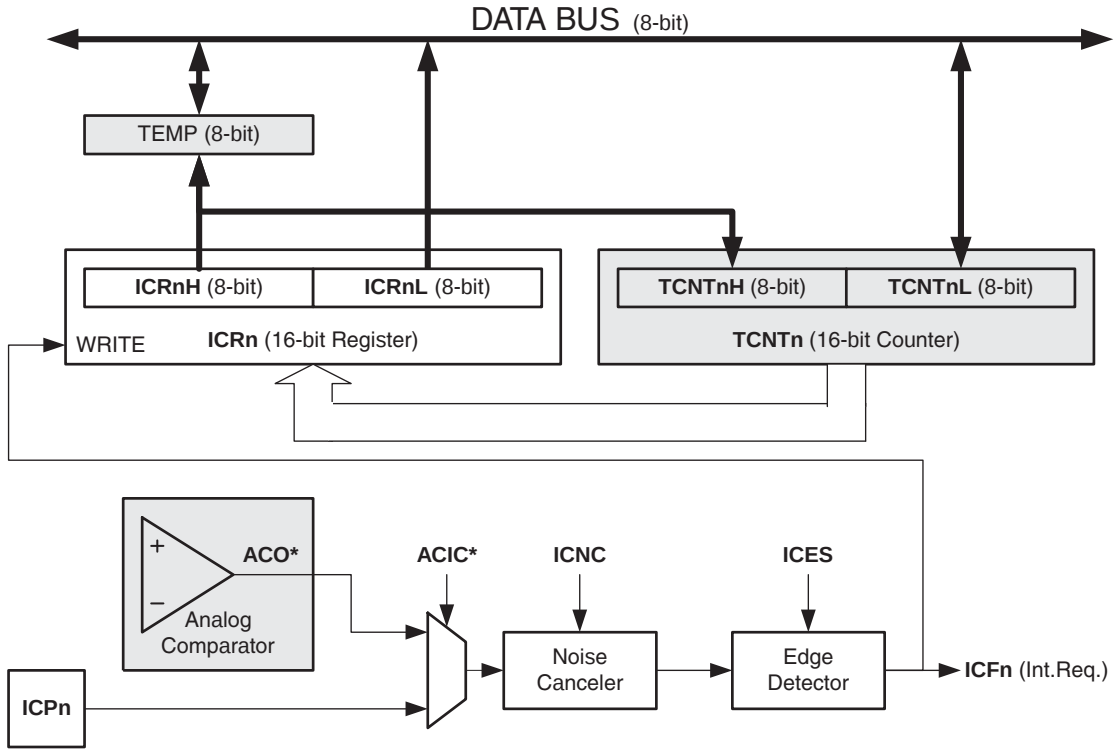
The Timer/Counter Overflow Flag (TOV1) is set according to the mode of operation selected by the WGM13:0 bits. TOV1 can be used for generating a CPU interrupt.

12.5 Input Capture Unit

The Timer/Counter incorporates an Input Capture unit that can capture external events and give them a time-stamp indicating time of occurrence. The external signal indicating an event, or multiple events, can be applied via the ICP1 pin or alternatively, via the analog-comparator unit. The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied. Alternatively the time-stamps can be used for creating a log of the events.

The Input Capture unit is illustrated by the block diagram shown in Figure 12-3 on page 91. The elements of the block diagram that are not directly a part of the Input Capture unit are gray shaded. The small “n” in register and bit names indicates the Timer/Counter number.

Figure 12-3. Input Capture Unit Block Diagram



When a change of the logic level (an event) occurs on the Input Capture pin (ICP1), alternatively on the Analog Comparator output (ACO), and this change confirms 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 "Accessing 16-bit Registers" on page 105.

12.5.1 Input Capture Trigger Source

The main trigger source for the Input Capture unit is the Input Capture pin (ICP1). Timer/Counter1 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 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) inputs are sampled using the same technique as for the T1 pin (Figure 13-1 on page 115). The edge detector is also identical. However, when the noise canceler is enabled, additional logic is inserted before the edge detector, which increases the delay by four system clock cycles. Note that the input of the noise canceler and edge detector is 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.

12.5.2 Noise Canceler

The noise canceler uses a simple digital filtering technique to improve noise immunity. Consecutive samples are monitored in a pipeline four units deep. The signal going to the edge detecter is allowed to change only when all four samples are equal.

The noise canceler is enabled by setting the Input Capture Noise Canceler (ICNC1) bit in Timer/Counter Control Register B (TCCR1B). When enabled, the noise canceler introduces an additional delay of four system clock cycles to a change applied to the input and before ICR1 is updated.

The noise canceler uses the system clock directly and is therefore not affected by the prescaler.

12.5.3 Using the Input Capture Unit

The main challenge when using the Input Capture unit is to assign enough processor capacity for handling the incoming events. The time between two events is critical. If the processor has not read the captured value in the ICR1 Register before the next event occurs, the ICR1 will be overwritten with a new value. In this case the result of the capture will be incorrect.

When using the Input Capture interrupt, the ICR1 Register should be read as early in the interrupt handler routine as possible. Even though the Input Capture interrupt has relatively high priority, the maximum interrupt response time is dependent on the maximum number of clock cycles it takes to handle any of the other interrupt requests.

Using the Input Capture unit in any mode of operation when the TOP value (resolution) is actively changed during operation, is not recommended.

Measurement of an external signal's duty cycle requires that the trigger edge is changed after each capture. Changing the edge sensing must be done as early as possible after the ICR1 Register has been read. After a change of the edge, the Input Capture Flag (ICF1) must be cleared by software (writing a logical one to the I/O bit location). For measuring frequency only, the clearing of the ICF1 flag is not required (if an interrupt handler is used).

12.6 Output Compare Units

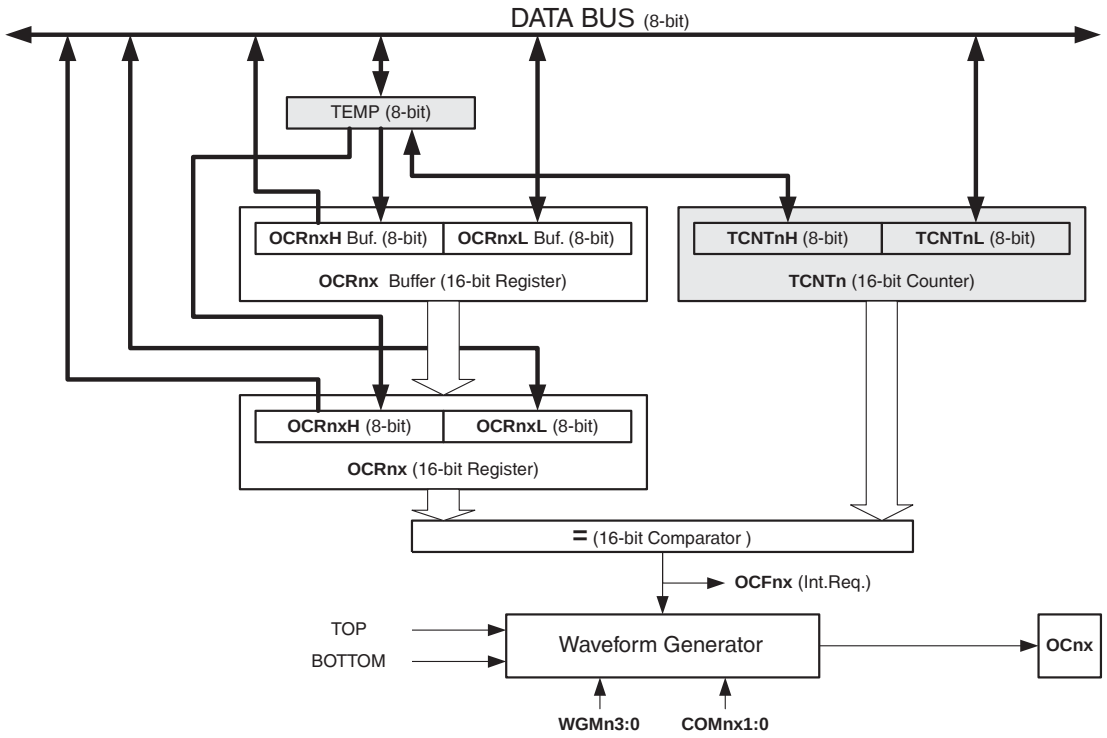
The 16-bit comparator continuously compares TCNT1 with the Output Compare Register (OCR1x). If TCNT equals OCR1x the comparator signals a match. A match will set the Output Compare Flag (OCF1x) at the next timer clock cycle. If enabled (OCIE1x = 1), the Output Compare Flag generates an Output Compare interrupt. The OCF1x flag is automatically cleared when the interrupt is executed. Alternatively the OCF1x flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the Waveform Generation mode

(WGM13:0) bits and Compare Output mode (COM1x1:0) bits. The TOP and BOTTOM signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (“Modes of Operation” on page 96).

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

Figure 12-4 on page 93 shows a block diagram of the Output Compare unit. The small “n” in the register and bit names indicates the device number (n = 1 for Timer/Counter 1), and the “x” indicates Output Compare unit (A/B). The elements of the block diagram that are not directly a part of the Output Compare unit are gray shaded.

Figure 12-4. 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 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 only changed by a write operation (the Timer/Counter does not update this register automatically as the TCNT1 and ICR1 Register). 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 Reg-

ister since 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 OCR1x Compare Register in the same system clock cycle.

For more information of how to access the 16-bit registers refer to “Accessing 16-bit Registers” on page 105.

12.6.1 Force Output Compare

In non-PWM Waveform Generation modes, the match output of the comparator can be forced by writing a 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 bits settings define whether the OC1x pin is set, cleared or toggled).

12.6.2 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.

12.6.3 Using the Output Compare Unit

Since writing TCNT1 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT1 when using any of the Output Compare channels, independent of whether the Timer/Counter is running or not. If the value written to TCNT1 equals the OCR1x value, the compare match will be missed, resulting in incorrect waveform generation. Do not write the TCNT1 equal to TOP in PWM modes with variable TOP values. The compare match for the TOP will be ignored and the counter will continue to 0xFFFF. Similarly, do not write the TCNT1 value equal to BOTTOM when the counter is downcounting.

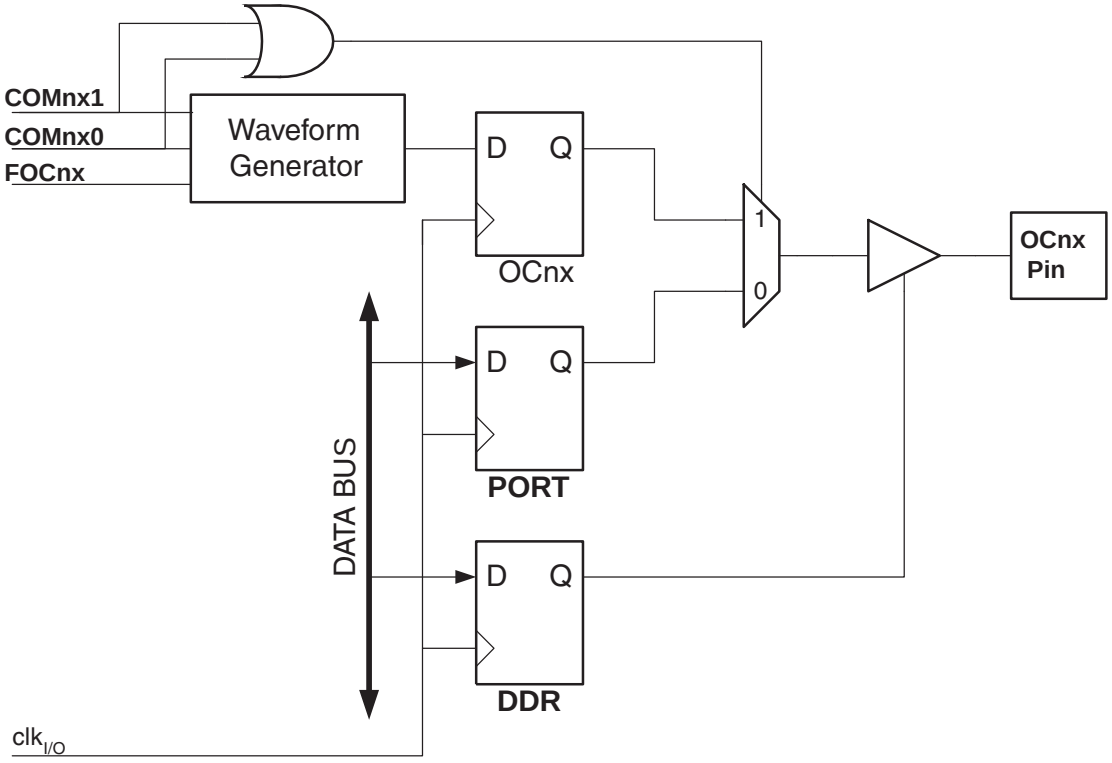
The setup of the OC1x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC1x value is to use the Force Output Compare (1x) strobe bits in Normal mode. The OC1x Register keeps its value even when changing between Waveform Generation modes.

Be aware that the COM1x1:0 bits are not double buffered together with the compare value. Changing the COM1x1:0 bits will take effect immediately.

12.7 Compare Match Output Unit

The Compare Output Mode (COM1x1:0) bits have two functions. The Waveform Generator uses the COM1x1:0 bits for defining the Output Compare (OC1x) state at the next compare match. Secondly the COM1x1:0 bits control the OC1x pin output source. Figure 12-5 on page 95 shows a simplified schematic of the logic affected by the COM1x1:0 bit setting. The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O port control registers (DDR and PORT) that are affected by the COM1x1:0 bits are shown. When referring to the OC1x state, the reference is for the internal OC1x Register, not the OC1x pin. If a system reset occur, the OC1x Register is reset to “0”.

Figure 12-5. Compare Match Output Unit, Schematic (non-PWM Mode)



The general I/O port function is overridden by the Output Compare (OC1x) from the Waveform Generator if either of the COM1x1:0 bits are set. However, the OC1x pin direction (input or output) is still controlled by the *Data Direction Register* (DDR) for the port pin. The Data Direction Register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the Waveform Generation mode, but there are some exceptions. See Table 12-2 on page 109, Table 12-3 on page 109 and Table 12-4 on page 109 for details.

The design of the Output Compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x1:0 bit settings are reserved for certain modes of operation. See "Register Description" on page 108

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

12.7.1 Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM1x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x1:0 = 0 tells the Waveform Generator that no action on the OC1x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 12-2 on page 109. For fast PWM mode refer to Table 12-3 on page 109, and for phase correct and phase and frequency correct PWM refer to Table 12-4 on page 109.

A change of the COM1x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the 1x strobe bits.

12.8 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 (WGM13:0) and Compare Output mode (COM1x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM1x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM1x1:0 bits control whether the output should be set, cleared or toggle at a compare match ("Compare Match Output Unit" on page 94)

For detailed timing information refer to "Timer/Counter Timing Diagrams" on page 103.

12.8.1 Normal Mode

The simplest mode of operation is the Normal mode (WGM13:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 16-bit value (MAX = 0xFFFF) and then restarts from the BOTTOM (0x0000). In normal operation the Timer/Counter Overflow Flag (TOV1) will be set in the same timer clock cycle as the TCNT1 becomes zero. The TOV1 flag in this case behaves like a 17th bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV1 flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

The Input Capture unit is easy to use in Normal mode. However, observe that the maximum interval between the external events must not exceed the resolution of the counter. If the interval between events are too long, the timer overflow interrupt or the prescaler must be used to extend the resolution for the capture unit.

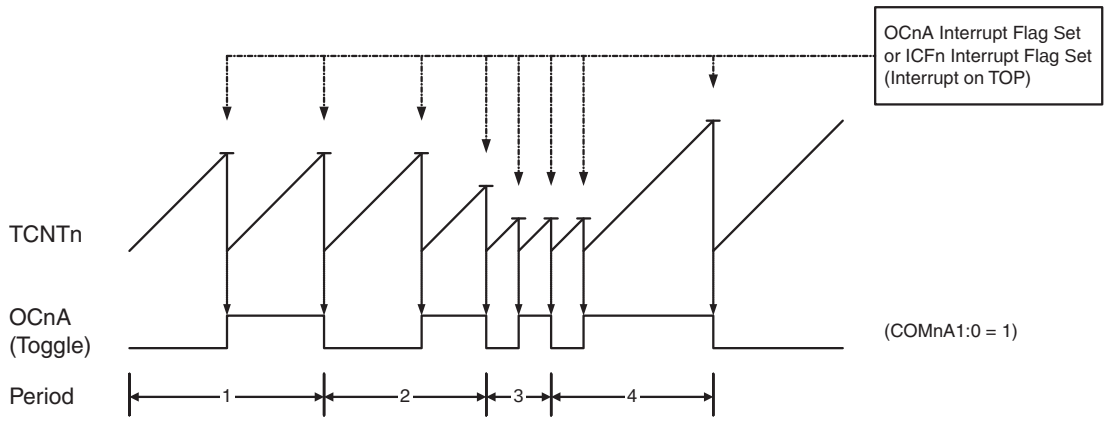
The Output Compare units can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

12.8.2 Clear Timer on Compare Match (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM13:0 = 4 or 12), the OCR1A or ICR1 Register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT1) matches either the OCR1A (WGM13:0 = 4) or the ICR1 (WGM13:0 = 12). The OCR1A or ICR1 define the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 12-6 on page 97. The counter value (TCNT1) increases until a compare match occurs with either OCR1A or ICR1, and then counter (TCNT1) is cleared.

Figure 12-6. CTC Mode, Timing Diagram



An interrupt can be generated at each time the counter value reaches the TOP value by either using the OCF1A or ICF1 flag according to the register used to define the TOP value. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing the TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR1A or ICR1 is lower than the current value of TCNT1, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. In many cases this feature is not desirable. An alternative will then be to use the fast PWM mode using OCR1A for defining TOP (WGM13:0 = 15) since the OCR1A then will be double buffered.

For generating a waveform output in CTC mode, the OC1A output can be set to toggle its logical level on each compare match by setting the Compare Output mode bits to toggle mode (COM1A1:0 = 1). The OC1A value will not be visible on the port pin unless the data direction for the pin is set to output (DDR_OC1A = 1). The waveform generated will have a maximum frequency of $_{1A} = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). The waveform frequency is defined by the following equation:

$$f_{OCnA} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

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

As for the Normal mode of operation, the TOV1 flag is set in the same timer clock cycle that the counter counts from MAX to 0x0000.

12.8.3 Fast PWM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM13:0 = 5, 6, 7, 14, or 15) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM options by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x, and set at BOTTOM. In inverting Compare Output mode output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct and phase and frequency correct PWM modes that use dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC

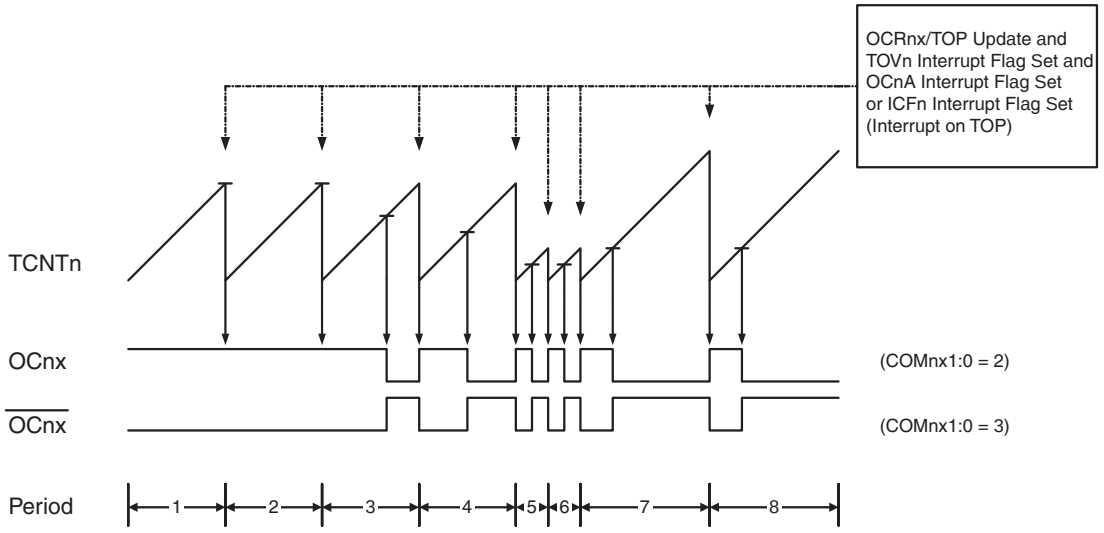
applications. High frequency allows physically small sized external components (coils, capacitors), hence reduces total system cost.

The PWM resolution for fast PWM can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{FPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

In fast PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 5, 6, or 7), the value in ICR1 (WGM13:0 = 14), or the value in OCR1A (WGM13:0 = 15). The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 12-7 on page 98. The figure shows fast PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 12-7. 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 at the same timer clock cycle as TOV1 is set when either OCR1A or ICR1 is used for defining the TOP value. If one of the interrupts are enabled, the interrupt handler routine can be used for updating the TOP and compare values.

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

The procedure for updating ICR1 differs from updating OCR1A when used for defining the TOP value. The ICR1 Register is not double buffered. This means that if ICR1 is changed to a low

value when the counter is running with none or a low prescaler value, there is a risk that the new ICR1 value written is lower than the current value of TCNT1. The result will then be that the counter will miss the compare match at the TOP value. The counter will then have to count to the MAX value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. The OCR1A Register however, is double buffered. This feature allows the OCR1A I/O location to be written anytime. When the OCR1A I/O location is written the value written will be put into the OCR1A Buffer Register. The OCR1A Compare Register will then be updated with the value in the Buffer Register at the next timer clock cycle the TCNT1 matches TOP. The update is done at the same timer clock cycle as the TCNT1 is cleared and the TOV1 flag is set.

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

In fast PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to three (see Table 12-3 on page 109). The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x Register at the compare match between OCR1x and TCNT1, and clearing (or setting) the OC1x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

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

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

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x Register represents 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 the 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 (with 50% duty cycle) waveform output 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.

12.8.4 Phase Correct PWM Mode

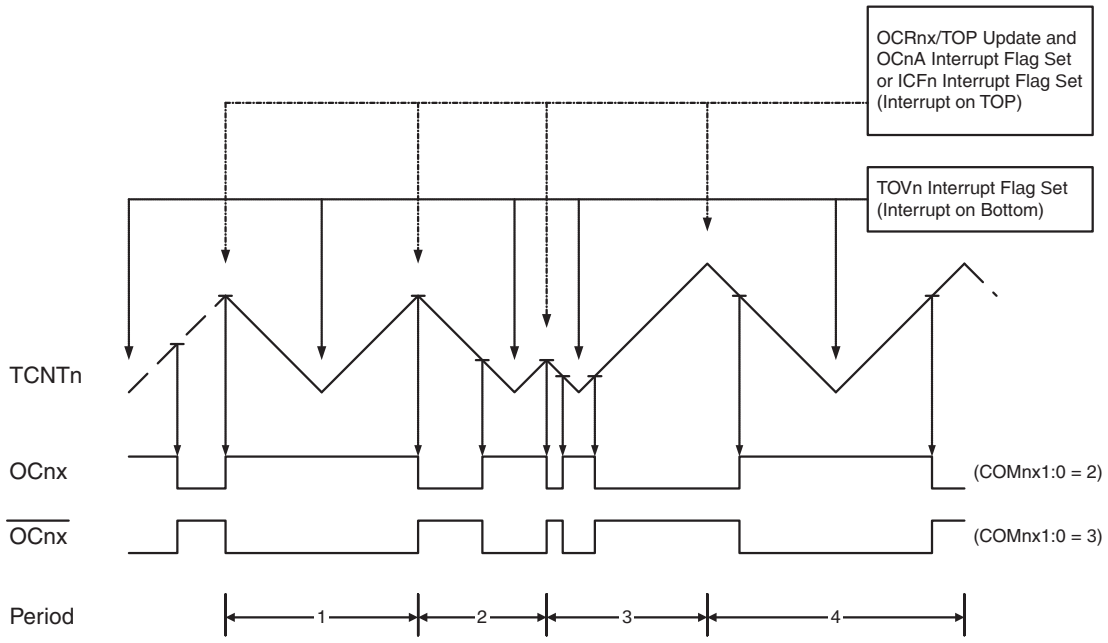
The phase correct Pulse Width Modulation or phase correct PWM mode (WGM13:0 = 1, 2, 3, 10, or 11) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is, like the phase and frequency correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. 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.

The PWM resolution for the phase correct PWM mode can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{PCPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

In phase correct PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 1, 2, or 3), the value in ICR1 (WGM13:0 = 10), or the value in OCR1A (WGM13:0 = 11). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 12-8 on page 100. The figure shows phase correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 12-8. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set each time the counter reaches BOTTOM. When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 flag is set accordingly at the same timer clock cycle as the OCR1x Registers are updated with the double buffer value (at TOP). The interrupt flags can be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values, the unused bits are masked to zero when any of the OCR1x Registers are written. As the third period shown in Figure 12-8 on page 100 illustrates,

changing the TOP actively while the Timer/Counter is running in the phase correct mode can result in an unsymmetrical output. The reason for this can be found in the time of update of the OCR1x Register. Since the OCR1x update occurs at TOP, the PWM period starts and ends at TOP. This implies that the length of the falling slope is determined by the previous TOP value, while the length of the rising slope is determined by the new TOP value. When these two values differ the two slopes of the period will differ in length. The difference in length gives the unsymmetrical result on the output.

It is recommended to use the phase and frequency correct mode instead of the phase correct mode when changing the TOP value while the Timer/Counter is running. When using a static TOP value there are practically no differences between the two modes of operation.

In phase correct 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 output can be generated by setting the COM1x1:0 to three (See Table 12-4 on page 109). 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 when the counter increments, and clearing (or setting) the OC1x Register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot TOP}$$

The N variable 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 phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

12.8.5 Phase and Frequency Correct PWM Mode

The phase and frequency correct Pulse Width Modulation, or phase and frequency correct PWM mode (WGM13:0 = 8 or 9) provides a high resolution phase and frequency correct PWM waveform generation option. The phase and frequency correct PWM mode is, like the phase correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting Compare Output mode, the operation is inverted. The dual-slope operation gives a lower maximum operation frequency compared to the single-slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

The main difference between the phase correct, and the phase and frequency correct PWM mode is the time the OCR1x Register is updated by the OCR1x Buffer Register, (see Figure 12-8 on page 100 and Figure 12-9 on page 102).

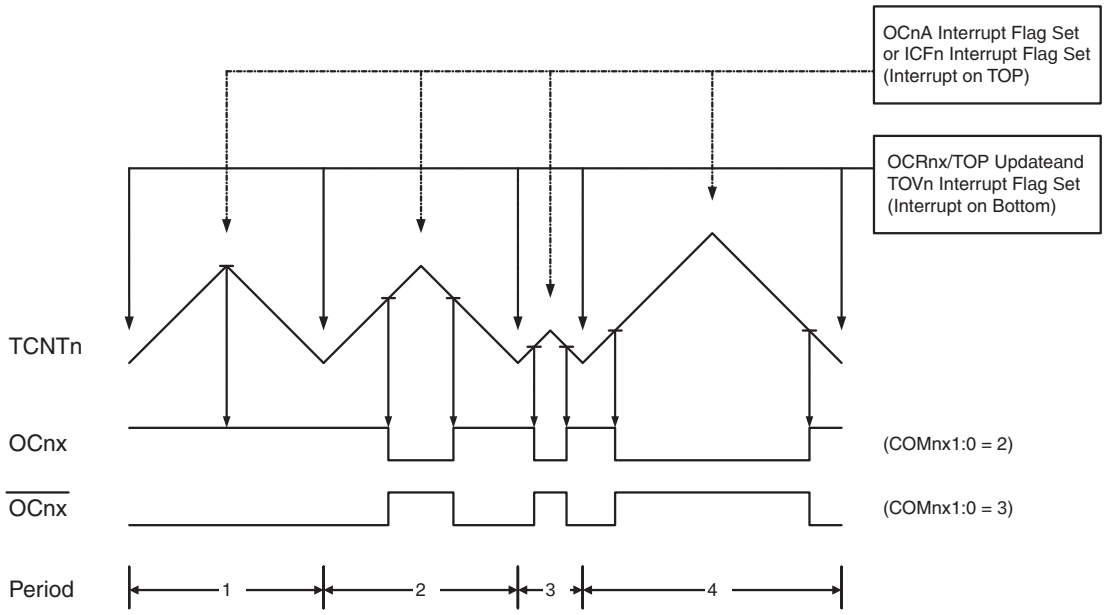
The PWM resolution for the phase and frequency correct PWM mode can be defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and

the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated using the following equation:

$$R_{PFCPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

In phase and frequency correct PWM mode the counter is incremented until the counter value matches either the value in ICR1 (WGM13:0 = 8), or the value in OCR1A (WGM13:0 = 9). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct and frequency correct PWM mode is shown on Figure 12-9 on page 102. The figure shows phase and frequency correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 12-9. Phase and Frequency Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set at the same timer clock cycle as the OCR1x Registers are updated with the double buffer value (at BOTTOM). When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 flag set when TCNT1 has reached TOP. The interrupt flags can then be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x.

As Figure 12-9 on page 102 shows the output generated is, in contrast to the phase correct mode, symmetrical in all periods. Since the OCR1x Registers are updated at BOTTOM, the length of the rising and the falling slopes will always be equal. This gives symmetrical output pulses and is therefore frequency correct.

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

In phase and frequency correct 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 output can be generated by setting the COM1x1:0 to three (See Table 12-4 on page 109). 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 when the counter increments, and clearing (or setting) the OC1x Register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase and frequency correct PWM can be calculated by the following equation:

$$f_{OCnxPFCPWM} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot TOP}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x Register represents special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be set to high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

12.9 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T1}) is therefore shown as a clock enable signal in the following figures. The figures include information on when interrupt flags are set, and when the OCR1x Register is updated with the OCR1x buffer value (only for modes utilizing double buffering). Figure 12-10 shows a timing diagram for the setting of OCF1x.

Figure 12-10. Timer/Counter Timing Diagram, Setting of OCF1x, no Prescaling

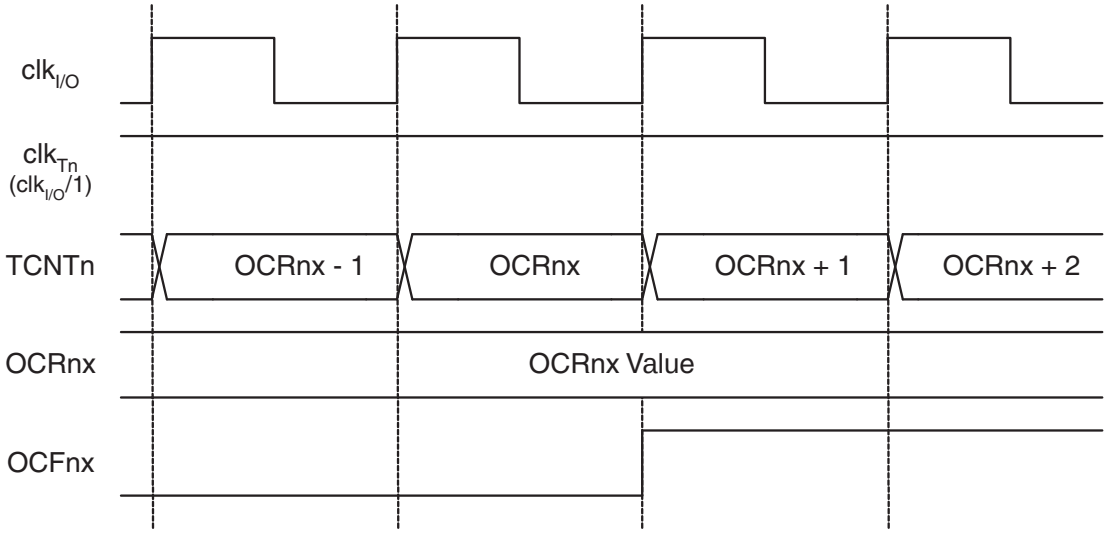


Figure 12-11 on page 104 shows the same timing data, but with the prescaler enabled.

Figure 12-11. Timer/Counter Timing Diagram, Setting of OCF1x, with Prescaler ($f_{clk\_I/O}/8$)

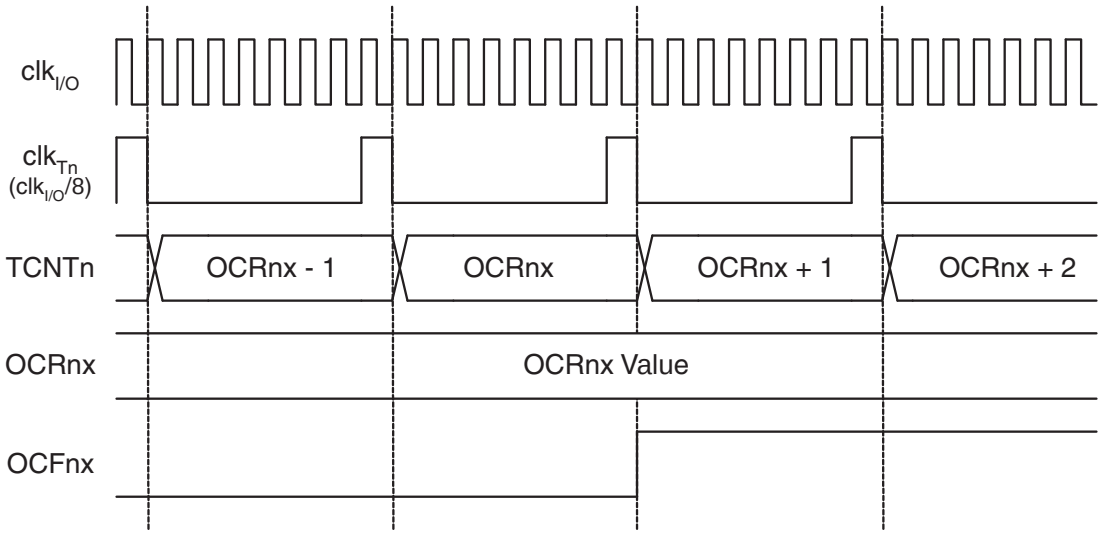


Figure 12-12 on page 104 shows the count sequence close to TOP in various modes. When using phase and frequency correct PWM mode the OCR1x Register is updated at BOTTOM. The timing diagrams will be the same, but TOP should be replaced by BOTTOM, TOP-1 by BOTTOM+1 and so on. The same renaming applies for modes that set the TOV1 flag at BOTTOM.

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

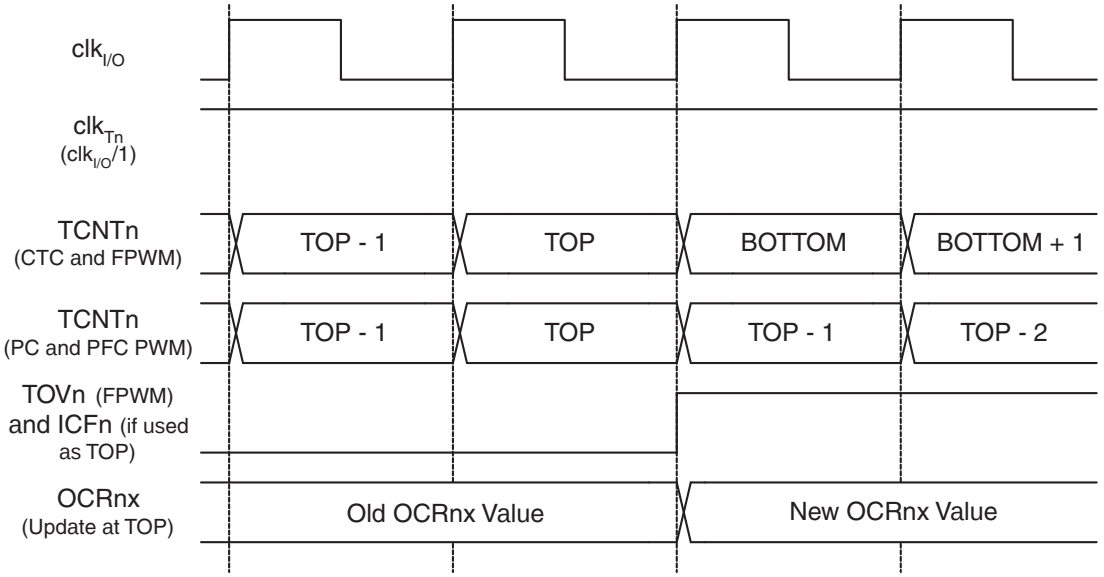
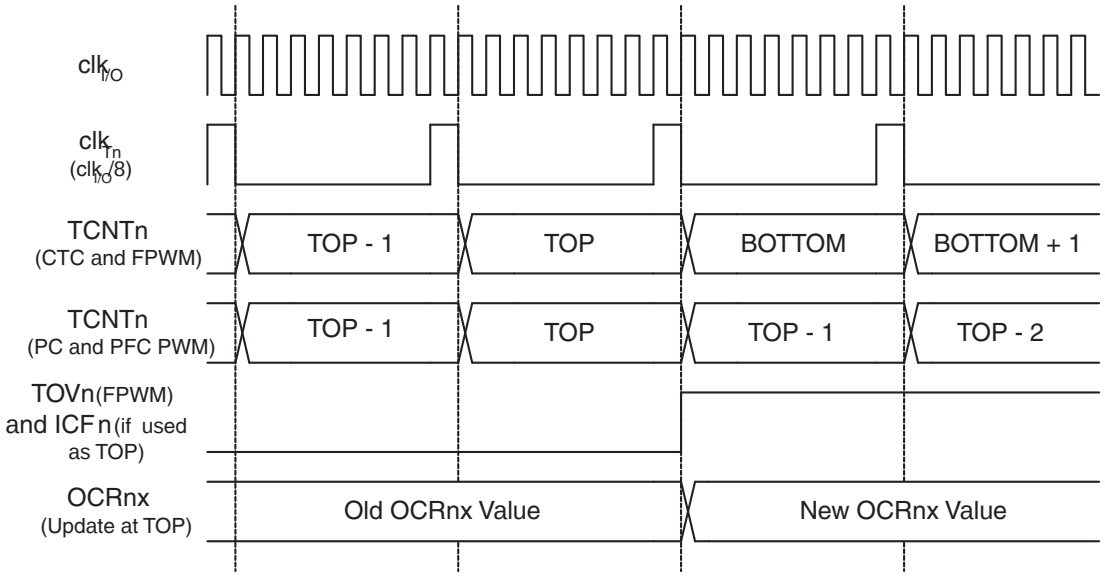


Figure 12-13 on page 105 shows the same timing data, but with the prescaler enabled.

Figure 12-13. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)



12.10 Accessing 16-bit Registers

The TCNT1, OCR1A/B, and ICR1 are 16-bit registers that can be accessed by the AVR CPU via the 8-bit data bus. The 16-bit register must be byte accessed using two read or write operations. Each 16-bit timer has a single 8-bit register for temporary storing of the high byte of the 16-bit access. The same temporary register is shared between all 16-bit registers within each 16-bit timer. Accessing the low byte triggers the 16-bit read or write operation. When the low byte of a 16-bit register is written by the CPU, the high byte stored in the temporary register, and the low byte written are both copied into the 16-bit register in the same clock cycle. When the low byte of a 16-bit register is read by the CPU, the high byte of the 16-bit register is copied into the temporary register in the same clock cycle as the low byte is read.

Not all 16-bit accesses uses the temporary register for the high byte. Reading the OCR1A/B 16-bit registers does not involve using the temporary register.

To do a 16-bit write, the high byte must be written before the low byte. For a 16-bit read, the low byte must be read before the high byte.

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

```
...
; 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

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

Note: See “Code Examples” on page 6.

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 other of the 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.

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

Assembly Code Example

```
TIM16_ReadTCNT1:
  ; Save global interrupt flag
  in  r18,SREG
  ; Disable interrupts
  cli
  ; Read TCNT1 into r17:r16
  in  r16,TCNT1L
  in  r17,TCNT1H
  ; Restore global interrupt flag
  out SREG,r18
  ret
```

C Code Example

```
unsigned int TIM16_ReadTCNT1( void )
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Read TCNT1 into i */
  i = TCNT1;
  /* Restore global interrupt flag */
  SREG = sreg;
  return i;
}
```

Note: See “Code Examples” on page 6.

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

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

```
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

```
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: See “Code Examples” on page 6.

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

12.10.1 Reusing the Temporary High Byte Register

If writing to more than one 16-bit register where the high byte is the same for all registers written, 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.

12.11 Register Description

12.11.1 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 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 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 the OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent of the WGM13:0 bits setting. Table 12-2 on page 109 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a Normal or a CTC mode (non-PWM).

Table 12-2. 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 12-3 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 12-3. 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. “Fast PWM Mode” on page 97 for more details.

Table 12-4 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

Table 12-4. Compare Output Mode, Phase Correct and Phase & Frequency Correct 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 when up-counting. Set OC1A/OC1B on Compare Match when downcounting.
1	1	Set OC1A/OC1B on Compare Match when up-counting. Clear OC1A/OC1B on Compare Match when downcounting.

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. “Phase Correct PWM Mode” on page 99 for more details.

- **Bit 1:0 – WGM11:0: Waveform Generation Mode**

Combined with the WGM13:2 bits found in the TCCR1B 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 12-5 on page 110. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare match (CTC) mode, and three types of Pulse Width Modulation (PWM) modes. (“Modes of Operation” on page 96).

Table 12-5. Waveform Generation Modes

Mode	WGM 13:10	Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0000	Normal	0xFFFF	Immediate	MAX
1	0001	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0010	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0011	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0100	CTC (Clear Timer on Compare)	OCR1A	Immediate	MAX
5	0101	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0110	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0111	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1000	PWM, Phase & Freq. Correct	ICR1	BOTTOM	BOTTOM
9	1001	PWM, Phase & Freq. Correct	OCR1A	BOTTOM	BOTTOM
10	1010	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1011	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1100	CTC (Clear Timer on Compare)	ICR1	Immediate	MAX
13	1101	(Reserved)	–	–	–
14	1110	Fast PWM	ICR1	TOP	TOP
15	1111	Fast PWM	OCR1A	TOP	TOP

12.11.2 TCCR1B – Timer/Counter1 Control Register B

Bit	7	6	5	4	3	2	1	0	
0x2E (0x4E)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ICNC1: Input Capture Noise Canceler**

Setting this bit (to one) activates the Input Capture Noise Canceler. When the noise canceler is activated, the input from the Input Capture pin (ICP1) is filtered. The filter function requires four successive equal valued samples of the ICP1 pin for changing its output. The Input Capture is therefore delayed by four Oscillator cycles when the noise canceler is enabled.

- **Bit 6 – ICES1: Input Capture Edge Select**

This bit selects which edge on the Input Capture pin (ICP1) that is used to trigger a capture event. When the ICES1 bit is written to zero, a falling (negative) edge is used as trigger, and when the ICES1 bit is written to one, a rising (positive) edge will trigger the capture.

When a capture is triggered according to the ICES1 setting, the counter value is copied into the Input Capture Register (ICR1). The event will also set the Input Capture Flag (ICF1), and this can be used to cause an Input Capture Interrupt, if this interrupt is enabled.

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

- **Bit 5 – Reserved Bit**

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

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

See TCCR1A Register description.

- **Bit 2:0 – CS12:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter, see Figure 12-10 and Figure 12-11.

Table 12-6. Clock Select Bit Description

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

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

12.11.3 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 specifies a non-PWM mode. However, for ensuring compatibility with future devices, these bits must be set to zero when TCCR1A is written when operating in a PWM mode. When writing a logical one to the FOC1A/FOC1B bit, an immediate compare match is forced on the Waveform Generation unit. The OC1A/OC1B output is changed according to its COM1x1:0 bits setting. Note that the

FOC1A/FOC1B bits are implemented as strobes. Therefore it is the value present in the COM1x1:0 bits that determine the effect of the forced compare.

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 zero when the register is written.

12.11.4 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, both for read and for write operations, to the Timer/Counter unit 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 “Accessing 16-bit Registers” on page 105.

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.

12.11.5 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	

12.11.6 OCR1BH and OCR1BL – Output Compare Register 1 B

Bit	7	6	5	4	3	2	1	0	
0x29 (0x49)	OCR1B[15:8]								OCR1BH
0x28 (0x48)	OCR1B[7:0]								OCR1BL
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 Output Compare Registers contain a 16-bit value that is continuously compared with the counter value (TCNT1). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC1x pin.

The Output Compare Registers are 16-bit in size. To ensure that both the high and low bytes are written simultaneously when the CPU writes to 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 “Accessing 16-bit Registers” on page 105.

12.11.7 ICR1H and ICR1L – Input Capture Register 1

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	ICR1[15:8]								ICR1H
0x24 (0x44)	ICR1[7:0]								ICR1L
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 Input Capture is updated with the counter (TCNT1) value each time an event occurs on the ICP1 pin (or optionally on the Analog Comparator output for Timer/Counter1). The Input Capture can be used for defining the counter TOP value.

The Input Capture Register is 16-bit in size. To ensure that both the high and low bytes are read 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. “Accessing 16-bit Registers” on page 105.

12.11.8 TIMSK1 – Timer/Counter Interrupt Mask Register 1

Bit	7	6	5	4	3	2	1	0	
0x0C (0x2C)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
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 zero when the register is written.

- **Bit 5 – ICIE1: Timer/Counter1, Input Capture Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Countern Input Capture interrupt is enabled. The corresponding Interrupt Vector (See “Interrupts” on page 66.) is executed when the ICF1 Flag, located in TIFR1, is set.

- **Bit 2– OCIE1B: Timer/Counter1, Output Compare B Match Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Output Compare B Match interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 48) is executed when the OCF1B flag, located in TIFR1, is set.

- **Bit 1– OCIE1A: Timer/Counter1, Output Compare A Match Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Output Compare A Match interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 48) is executed when the OCF1A flag, located in TIFR1, is set.

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

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Overflow interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 48) is executed when the TOV1 flag, located in TIFR1, is set.

12.11.9 TIFR1 – Timer/Counter Interrupt Flag Register 1

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	–	–	ICF1	–	–	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 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 logic 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 logic 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 logic one to its bit location.

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

The setting of this flag is dependent of the WGM13:0 bits setting. In Normal and CTC modes, the TOV1 flag is set when the timer overflows. See Table 12-5 on page 110 for the TOV1 flag behavior when using another WGM13:0 bit setting.

TOV1 is automatically cleared when the Timer/Counter1 Overflow Interrupt Vector is executed. Alternatively, TOV1 can be cleared by writing a logic one to its bit location.

13. Timer/Counter Prescaler

Timer/Counter0 and Timer/Counter1 share the same prescaler module, but the Timer/Counters can have different prescaler settings. The description below applies to both Timer/Counters. Tn is used as a general name, n = 0, 1.

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

13.1 Prescaler Reset

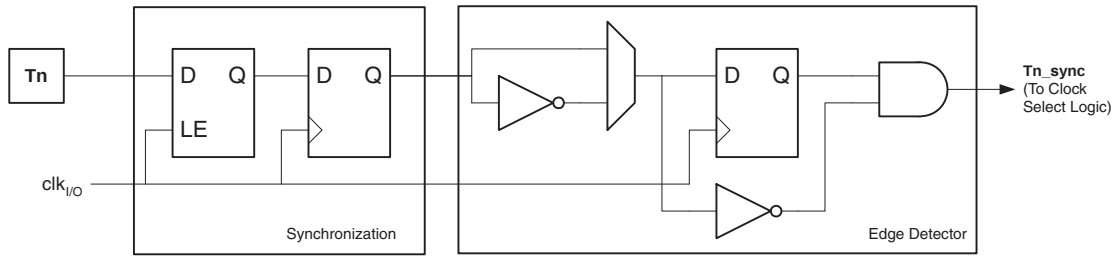
The prescaler is free running, i.e., operates independently of the Clock Select logic of the Timer/CounterCounter, and it is shared by the Timer/Counter Tn. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (CSn2:0 = 2, 3, 4, or 5). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

The Prescaler Reset can be used for synchronizing the Timer/Counter to program execution.

13.2 External Clock Source

An external clock source applied to the Tn pin can be used as Timer/Counter clock (clk_{Tn}). The Tn pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 13-1 shows a functional equivalent block diagram of the Tn synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

Figure 13-1. T0 Pin Sampling



The edge detector generates one clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects. The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the Tn pin to the counter is updated.

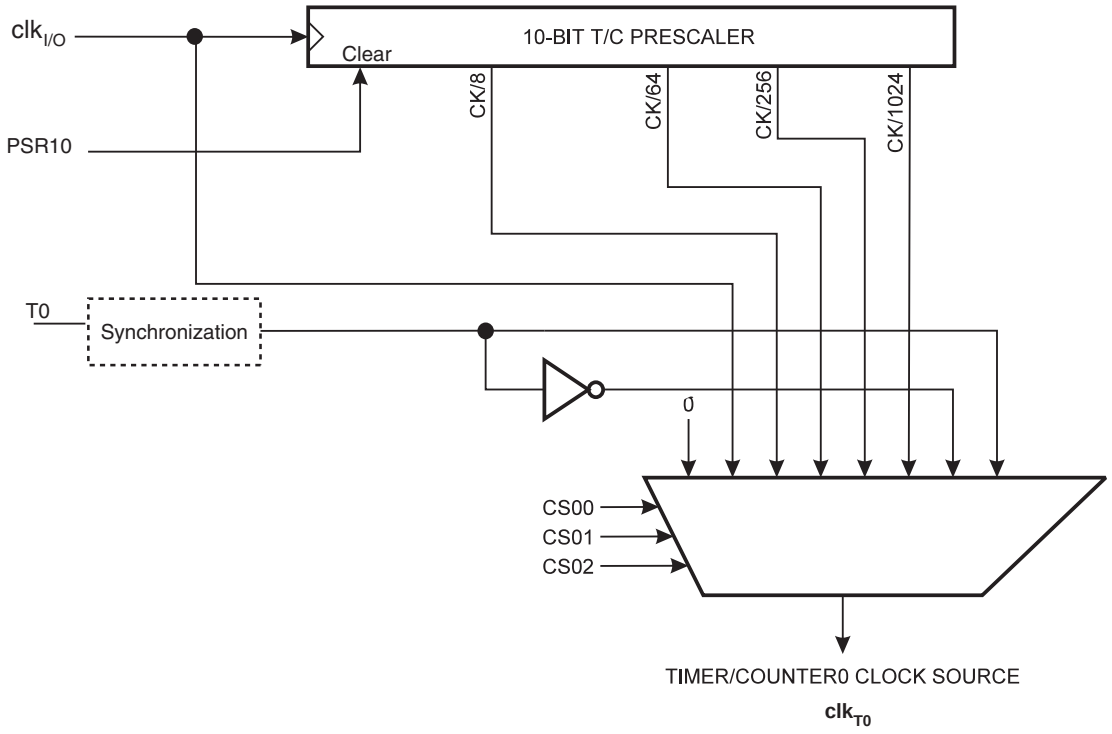
Enabling and disabling of the clock input must be done when Tn has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

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

quency (Nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by Oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk\_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 13-2. Prescaler for Timer/Counter0



Note: 1. The synchronization logic on the input pins (T0) is shown in Figure 13-1 on page 115.

13.3 Register Description

13.3.1 GTCCR – General Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x23 (0x43)	**TSM**	–	–	–	–	–	–	**PSR10**	**GTCCR**
Read/Write	R/W	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – TSM: Timer/Counter Synchronization Mode**

Writing the TSM bit to one activates the Timer/Counter Synchronization mode. In this mode, the value that is written to the PSR10 bit is kept, hence keeping the Prescaler Reset signal asserted. This ensures that the Timer/Counter is halted and can be configured without the risk of advancing during configuration. When the TSM bit is written to zero, the PSR10 bit is cleared by hardware, and the Timer/Counter start counting.

- **Bit 0 – PSR10: Prescaler 0 Reset Timer/Counter n**

When this bit is one, the Timer/Countern prescaler will be Reset. This bit is normally cleared immediately by hardware, except if the TSM bit is set.

14. USI – Universal Serial Interface

14.1 Features

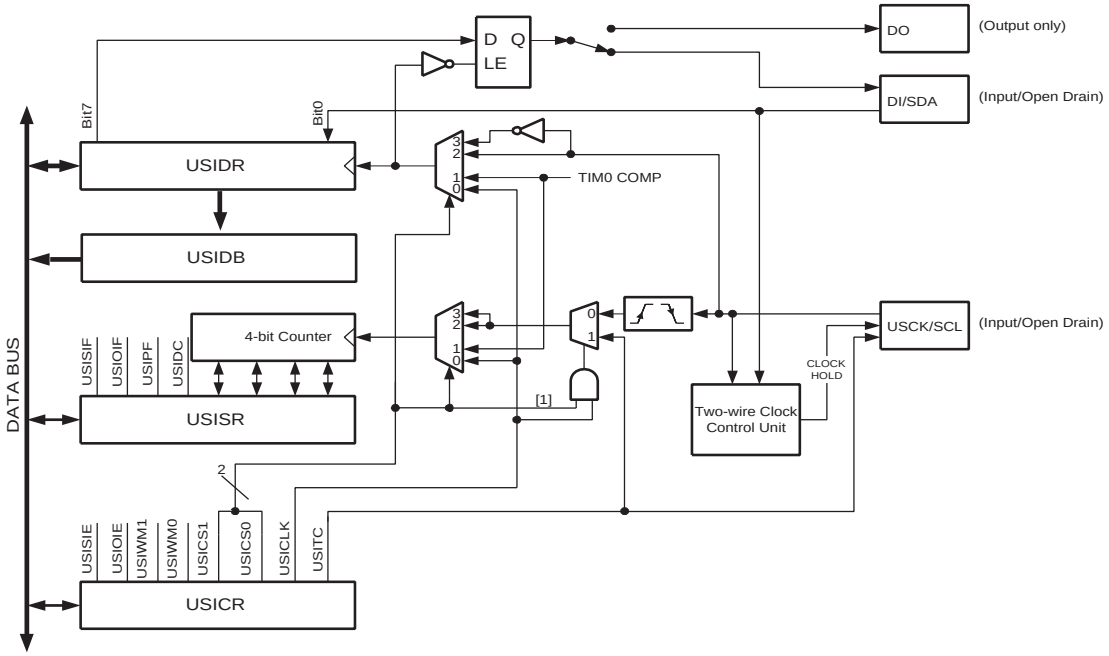
- **Two-wire Synchronous Data Transfer (Master or Slave)**
- **Three-wire Synchronous Data Transfer (Master or Slave)**
- **Data Received Interrupt**
- **Wakeup from Idle Mode**
- **In Two-wire Mode: Wake-up from All Sleep Modes, Including Power-down Mode**
- **Two-wire Start Condition Detector with Interrupt Capability**

14.2 Overview

The Universal Serial Interface (USI), provides the basic hardware resources needed for serial communication. Combined with a minimum of control software, the USI allows significantly higher transfer rates and uses less code space than solutions based on software only. Interrupts are included to minimize the processor load.

A simplified block diagram of the USI is shown in Figure 14-1 For actual placement of I/O pins refer to “Pinout ATtiny24/44/84” on page 2. Device-specific I/O Register and bit locations are listed in the “Register Descriptions” on page 124.

Figure 14-1. Universal Serial Interface, Block Diagram



The 8-bit USI Data Register (USIDR) contains the incoming and outgoing data. It is directly accessible via the data bus but a copy of the contents is also placed in the USI Buffer Register (USIBR) where it can be retrieved later. If reading the USI Data Register directly, the register must be read as quickly as possible to ensure that no data is lost.

The most significant bit of the USI Data Register is connected to one of two output pins (depending on the mode configuration, see “USICR – USI Control Register” on page 126). There is a transparent latch between the output of the USI Data Register and the output pin, which delays

the change of data output to the opposite clock edge of the data input sampling. The serial input is always sampled from the Data Input (DI) pin independent of the configuration.

The 4-bit counter can be both read and written via the data bus, and it can generate an overflow interrupt. Both the USI Data Register and the counter are clocked simultaneously by the same clock source. This allows the counter to count the number of bits received or transmitted and generate an interrupt when the transfer is complete. Note that when an external clock source is selected the counter counts both clock edges. This means the counter registers the number of clock edges and not the number of data bits. The clock can be selected from three different sources: The USCK pin, Timer/Counter0 Compare Match or from software.

The two-wire clock control unit can be configured to generate an interrupt when a start condition has been detected on the two-wire bus. It can also be set to generate wait states by holding the clock pin low after a start condition is detected, or after the counter overflows.

14.3 Functional Descriptions

14.3.1 Three-wire Mode

The USI Three-wire mode is compliant to the Serial Peripheral Interface (SPI) mode 0 and 1, but does not have the slave select (SS) pin functionality. However, this feature can be implemented in software if necessary. Pin names used by this mode are: DI, DO, and USCK.

Figure 14-2. Three-wire Mode Operation, Simplified Diagram

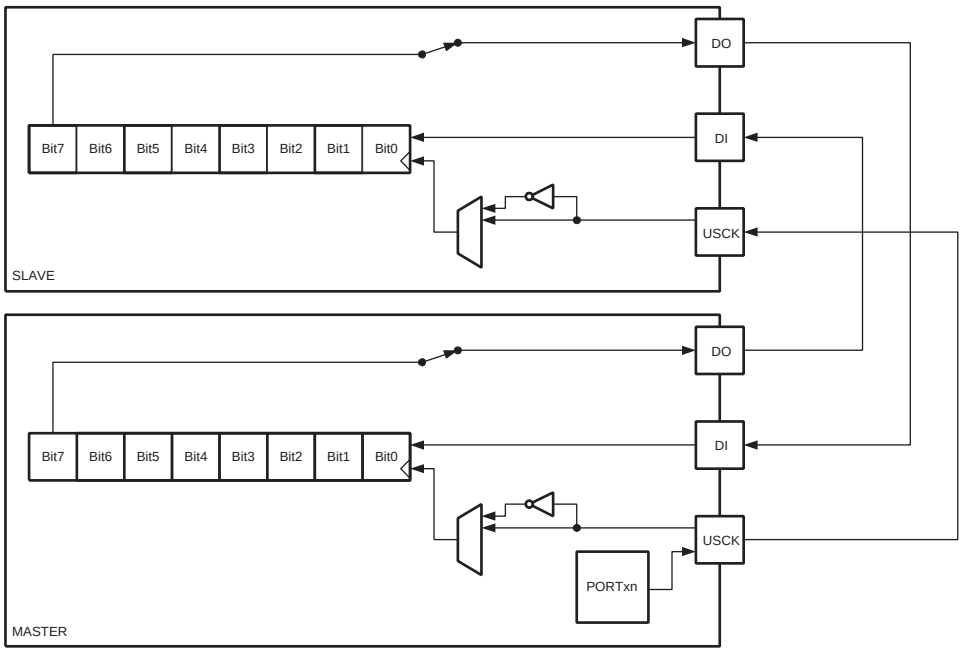
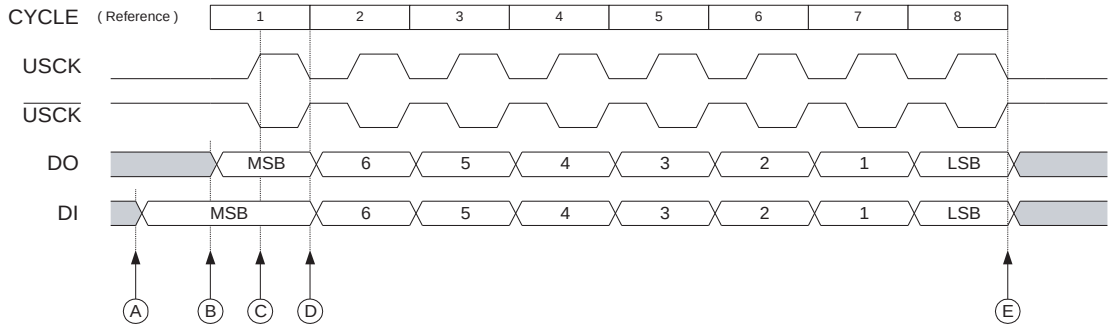


Figure 14-2 shows two USI units operating in three-wire mode, one as Master and one as Slave. The two USI Data Registers are interconnected in such way that after eight USCK clocks, the data in each register has been interchanged. The same clock also increments the USI's 4-bit counter. The Counter Overflow (interrupt) Flag, or USIOIF, can therefore be used to determine when a transfer is completed. The clock is generated by the Master device software by toggling the USCK pin via the PORTA register or by writing a one to bit USITC bit in USICR.

Figure 14-3. Three-wire Mode, Timing Diagram



The three-wire mode timing is shown in Figure 14-3 At the top of the figure is a USCK cycle reference. One bit is shifted into the USI Data Register (USIDR) for each of these cycles. The USCK timing is shown for both external clock modes. In external clock mode 0 (USICS0 = 0), DI is sampled at positive edges, and DO is changed (USI Data Register is shifted by one) at negative edges. In external clock mode 1 (USICS0 = 1) the opposite edges with respect to mode 0 are used. In other words, data is sampled at negative and output is changed at positive edges. The USI clock modes corresponds to the SPI data mode 0 and 1.

Referring to the timing diagram (Figure 14-3), a bus transfer involves the following steps:

1. The slave and master devices set up their data outputs and, depending on the protocol used, enable their output drivers (mark A and B). The output is set up by writing the data to be transmitted to the USI Data Register. The output is enabled by setting the corresponding bit in the Data Direction Register of Port A. Note that there is not a preferred order of points A and B in the figure, but both must be at least one half USCK cycle before point C, where the data is sampled. This is in order to ensure that the data setup requirement is satisfied. The 4-bit counter is reset to zero.
2. The master software generates a clock pulse by toggling the USCK line twice (C and D). The bit values on the data input (DI) pins are sampled by the USI on the first edge (C), and the data output is changed on the opposite edge (D). The 4-bit counter will count both edges.
3. Step 2. is repeated eight times for a complete register (byte) transfer.
4. After eight clock pulses (i.e., 16 clock edges) the counter will overflow and indicate that the transfer has been completed. If USI Buffer Registers are not used the data bytes that have been transferred must now be processed before a new transfer can be initiated. The overflow interrupt will wake up the processor if it is set to Idle mode. Depending on the protocol used the slave device can now set its output to high impedance.

14.3.2 SPI Master Operation Example

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

```
SPITransfer:
    out     USIDR,r16
    ldi     r16,(1<<USIOIF)
    out     USISR,r16
    ldi     r17,(1<<USIWM0)|(1<<USICS1)|(1<<USICLK)|(1<<USITC)
```

```
SPITransfer_loop:
    out    USICR,r17
    in     r16, USISR
    sbrs   r16, USIOIF
    rjmp   SPITransfer_loop
    in     r16,USIDR
    ret
```

The code is size optimized using only eight instructions (plus return). The code example assumes that the DO and USCK pins have been enabled as outputs in DDRA. The value stored in register r16 prior to the function is called is transferred to the slave device, and when the transfer is completed the data received from the slave is stored back into the register r16.

The second and third instructions clear the USI Counter Overflow Flag and the USI counter value. The fourth and fifth instructions set three-wire mode, positive edge clock, count at USITC strobe, and toggle USCK. The loop is repeated 16 times.

The following code demonstrates how to use the USI as an SPI master with maximum speed ($f_{SCK} = f_{CK}/2$):

```
SPITransfer_Fast:
    out    USIDR,r16
    ldi    r16,(1<<USIWM0)|(0<<USICS0)|(1<<USITC)
    ldi    r17,(1<<USIWM0)|(0<<USICS0)|(1<<USITC)|(1<<USICLK)

    out    USICR,r16 ; MSB
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16
    out    USICR,r17
    out    USICR,r16 ; LSB
    out    USICR,r17

    in     r16,USIDR
ret
```

14.3.3 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 (plus return). The code example assumes that the DO and USCK pins have been enabled as outputs in DDRA. The value stored in register r16 prior to the function is called is transferred to the master device, and when the transfer is completed the data received from the master is stored back into the register r16.

Note that the first two instructions are for initialization, only, and need only be executed once. These instructions set three-wire mode and positive edge clock. The loop is repeated until the USI Counter Overflow Flag is set.

14.3.4 Two-wire Mode

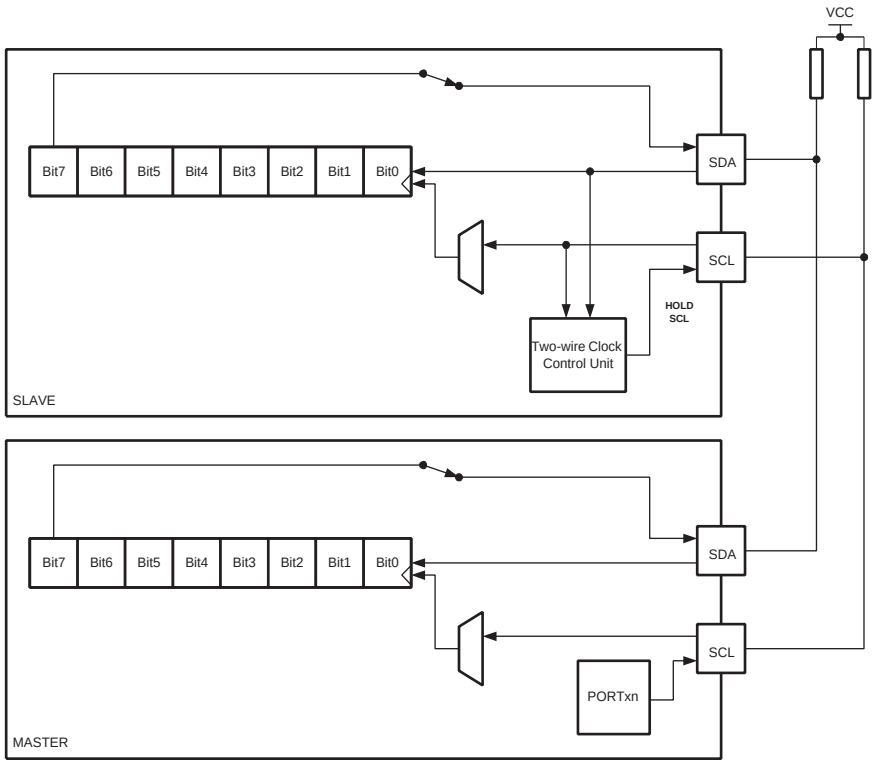
The USI two-wire mode is compliant to the Inter IC (TWI) bus protocol, but without slew rate limiting on outputs and without input noise filtering. Pin names used in this mode are SCL and SDA.

Figure 14-4 shows two USI units operating in two-wire mode, one as master and one as slave. It is only the physical layer that is shown since the system operation is highly dependent of the communication scheme used. The main differences between the master and slave operation at this level is the serial clock generation which is always done by the master. Only the slave uses the clock control unit.

Clock generation must be implemented in software, but the shift operation is done automatically in both devices. Note that clocking only on negative edges for shifting data is of practical use in this mode. The slave can insert wait states at start or end of 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.

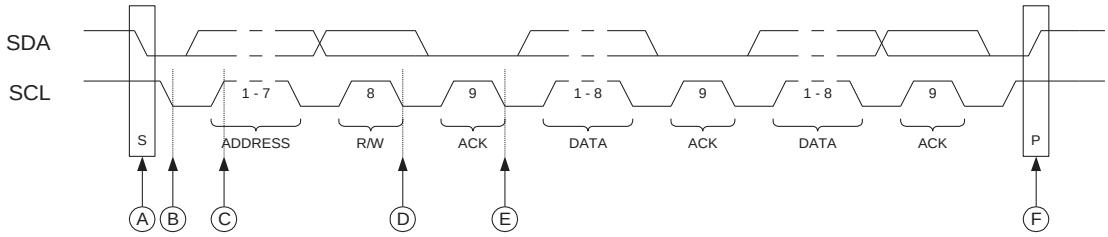
Since the clock also increments the counter, a counter overflow can be used to indicate that the transfer is completed. The clock is generated by the master by toggling the USCK pin via the PORTA register.

Figure 14-4. Two-wire Mode Operation, Simplified Diagram



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 14-5. Two-wire Mode, Typical Timing Diagram



Referring to the timing diagram (Figure 14-5), a bus transfer involves the following steps:

1. The start condition is generated by the master by forcing the SDA low line while keeping the SCL line high (A). SDA can be forced low either by writing a zero to bit 7 of the USI Data Register, or by setting the corresponding bit in the PORTA register to zero. Note that the Data Direction Register bit must be set to one for the output to be enabled. The start detector logic of the slave device (see Figure 14-6 on page 123) 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 other tasks before setting up the USI Data Register to receive the address. This is done by clearing the start condition flag and resetting the counter.

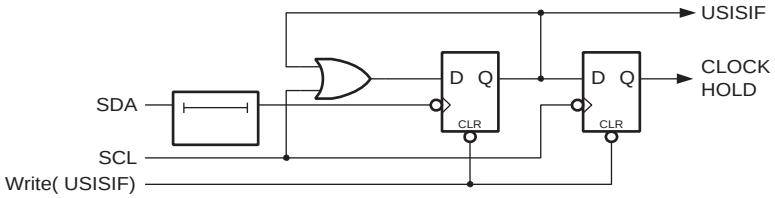
3. The master set the first bit to be transferred and releases the SCL line (C). The slave samples the data and shifts it into the USI Data Register at the positive edge of the SCL clock.
4. After eight bits containing slave address and data direction (read or write) have been transferred, the slave 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. When the slave is addressed, it holds the SDA line low during the acknowledgment cycle before holding the SCL line low again (i.e., the USI Counter Register must be set to 14 before releasing SCL at (D)). Depending on 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 acknowledge (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 transmitted.

14.3.5 Start Condition Detector

The start condition detector is shown in Figure 14-6. The SDA line is delayed (in the range of 50 to 300 ns) to ensure valid sampling of the SCL line. The start condition detector is only enabled in two-wire mode.

Figure 14-6. Start Condition Detector, Logic Diagram



The start condition detector works asynchronously and can therefore wake up the processor from power-down sleep mode. However, the protocol used might have restrictions on the SCL hold time. Therefore, when using this feature the oscillator start-up time (set by CKSEL fuses, see “Clock Sources” on page 25) must also be taken into consideration. Refer to the description of the USISIF bit on page 125 for further details.

14.3.6 Clock speed considerations

Maximum frequency for SCL and SCK is f_{CK} / 2. This is also the maximum data transmit and receive rate in both two- and three-wire mode. In two-wire slave mode the Two-wire Clock Control Unit will hold the SCL low until the slave is ready to receive more data. This may reduce the actual data rate in two-wire mode.

14.4 Alternative USI Usage

The flexible design of the USI allows it to be used for other tasks when serial communication is not needed. Below are some examples.

14.4.1 Half-Duplex Asynchronous Data Transfer

Using the USI Data Register in three-wire mode it is possible to implement a more compact and higher performance UART than by software, only.

14.4.2 4-Bit Counter

The 4-bit counter can be used as a stand-alone counter with overflow interrupt. Note that if the counter is clocked externally, both clock edges will increment the counter value.

14.4.3 12-Bit Timer/Counter

Combining the 4-bit USI counter with one of the 8-bit timer/counters creates a 12-bit counter.

14.4.4 Edge Triggered External Interrupt

By setting the counter to maximum value (F) it can function as an additional external interrupt. The Overflow Flag and Interrupt Enable bit are then used for the external interrupt. This feature is selected by the USICS1 bit.

14.4.5 Software Interrupt

The counter overflow interrupt can be used as a software interrupt triggered by a clock strobe.

14.5 Register Descriptions

14.5.1 USIDR – USI Data Register

Bit	7	6	5	4	3	2	1	0	
0x0F (0x2F)	MSB							LSB	USIDR
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 USI Data Register can be accessed directly but a copy of the data can also be found in the USI Buffer Register.

Depending on the USICS1:0 bits of the USI Control Register a (left) shift operation may be performed. The shift operation can be synchronised to an external clock edge, to a Timer/Counter0 Compare Match, or directly to software via the USICLK bit. If a serial clock occurs at the same cycle the register is written, the register will contain the value written and no shift is performed.

Note that even when no wire mode is selected (USIWM1:0 = 0) both the external data input (DI/SDA) and the external clock input (USCK/SCL) can still be used by the USI Data Register.

The output pin (DO or SDA, depending on the wire mode) is connected via the output latch to the most significant bit (bit 7) of the USI Data Register. The output latch ensures that data input is sampled and data output is changed on opposite clock edges. The latch is open (transparent) during the first half of a serial clock cycle when an external clock source is selected (USICS1 = 1) and constantly open when an internal clock source is used (USICS1 = 0). The output will be changed immediately when a new MSB is written as long as the latch is open.

Note that the Data Direction Register bit corresponding to the output pin must be set to one in order to enable data output from the USI Data Register.

14.5.2 USIBR – USI Data Buffer

Bit	7	6	5	4	3	2	1	0	
0x10 (0x30)	MSB							LSB	USIBR
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

Instead of reading data from the USI Data Register the USI Buffer Register can be used. This makes controlling the USI less time critical and gives the CPU more time to handle other program tasks. USI flags as set similarly as when reading the USIDR register.

The content of the USI Data Register is loaded to the USI Buffer Register when the transfer has been completed.

14.5.3 USISR – USI Status Register

Bit	7	6	5	4	3	2	1	0	
0x0D (0x2D)	USISIF	USIOIF	USIPF	USIDC	USICNT3	USICNT2	USICNT1	USICNT0	USISR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Status Register contains interrupt flags, line status flags and the counter value.

- **Bit 7 – USISIF: Start Condition Interrupt Flag**

When two-wire mode is selected, the USISIF Flag is set (to one) when a start condition has been detected. When three-wire mode or output disable mode has been selected any edge on the SCK pin will set the flag.

If USISIE bit in USICR and the Global Interrupt Enable Flag are set, an interrupt will be generated when this flag is set. The flag will only be cleared by writing a logical one to the USISIF bit. Clearing this bit will release the start detection hold of USCL in two-wire mode.

A start condition interrupt will wakeup the processor from all sleep modes.

- **Bit 6 – USIOIF: Counter Overflow Interrupt Flag**

This flag is set (one) when the 4-bit counter overflows (i.e., at the transition from 15 to 0). If the USIOIE bit in USICR and the Global Interrupt Enable Flag are set an interrupt will also be generated when the flag is set. The flag will only be cleared if a one is written to the USIOIF bit. Clearing this bit will release the counter overflow hold of SCL in two-wire mode.

A counter overflow interrupt will wakeup the processor from Idle sleep mode.

- **Bit 5 – USIPF: Stop Condition Flag**

When two-wire mode is selected, the USIPF Flag is set (one) when a stop condition has been detected. The flag is cleared by writing a one to this bit. Note that this is not an interrupt flag. This signal is useful when implementing two-wire bus master arbitration.

- **Bit 4 – USIDC: Data Output Collision**

This bit is logical one when bit 7 in the USI Data Register differs from the physical pin value. The flag is only valid when two-wire mode is used. This signal is useful when implementing Two-wire bus master arbitration.

- **Bits 3:0 – USICNT3:0: Counter Value**

These bits reflect the current 4-bit counter value. The 4-bit counter value can directly be read or written by the CPU.

The 4-bit counter increments by one for each clock generated either by the external clock edge detector, by a Timer/Counter0 Compare Match, or by software using USICLK or USITC strobe bits. The clock source depends on the setting of the USICS1:0 bits.

For external clock operation a special feature is added that allows the clock to be generated by writing to the USITC strobe bit. This feature is enabled by choosing an external clock source (USICS1 = 1) and writing a one to the USICLK bit.

Note that even when no wire mode is selected (USIWM1..0 = 0) the external clock input (USCK/SCL) can still be used by the counter.

14.5.4 USICR – USI Control Register

Bit	7	6	5	4	3	2	1	0	
0x0D (0x2D)	**USISIE**	**USIOIE**	**USIWM1**	**USIWM0**	**USICS1**	**USICS0**	**USICLK**	**USITC**	**USICR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W	
Initial Value	0	0	0	0	0	0	0	0	

The USI Control Register includes bits for interrupt enable, setting the wire mode, selecting the clock and clock strobe.

- **Bit 7 – USISIE: Start Condition Interrupt Enable**

Setting this bit to one enables the start condition detector interrupt. If there is a pending interrupt and USISIE and the Global Interrupt Enable Flag are set to one the interrupt will be executed immediately. Refer to the USISIF bit description on page 125 for further details.

- **Bit 6 – USIOIE: Counter Overflow Interrupt Enable**

Setting this bit to one enables the counter overflow interrupt. If there is a pending interrupt and USIOIE and the Global Interrupt Enable Flag are set to one the interrupt will be executed immediately. Refer to the USIOIF bit description on page 125 for further details.

- **Bit 5:4 – USIWM1:0: Wire Mode**

These bits set the type of wire mode to be used, as shown in Table 14-1 on page 127.

Basically, only the function of the outputs are affected by these bits. Data and clock inputs are not affected by the mode selected and will always have the same function. The counter and USI Data Register can therefore be clocked externally and data input sampled, even when outputs are disabled.

Table 14-1. Relationship between USIWM1:0 and USI Operation

USIWM1	USIWM0	Description
0	0	**Outputs, clock hold, and start detector disabled.** Port pins operate as normal.
0	1	**Three-wire mode. Uses DO, DI, and USCK pins.** The *Data Output* (DO) pin overrides the corresponding bit in the PORTA register. However, the corresponding DDRA bit still controls the data direction. When the port pin is set as input the pin pull-up is controlled by the PORTA bit. The *Data Input* (DI) and *Serial Clock* (USCK) pins do not affect the normal port operation. When operating as master, clock pulses are software generated by toggling the PORTA register, while the data direction is set to output. The USITC bit in the USICR Register can be used for this purpose.
1	0	**Two-wire mode. Uses SDA (DI) and SCL (USCK) pins(1).** The *Serial Data* (SDA) and the *Serial Clock* (SCL) pins are bi-directional and use open-collector output drives. The output drivers are enabled by setting the corresponding bit for SDA and SCL in the DDRA register. When the output driver is enabled for the SDA pin, the output driver will force the line SDA low if the output of the USI Data Register or the corresponding bit in the PORTA register is zero. Otherwise, the SDA line will not be driven (i.e., it is released). When the SCL pin output driver is enabled the SCL line will be forced low if the corresponding bit in the PORTA register is zero, or by the start detector. Otherwise the SCL line will not be driven. The SCL line is held low when a start detector detects a start condition and the output is enabled. Clearing the Start Condition Flag (USISIF) releases the line. The SDA and SCL pin inputs is not affected by enabling this mode. Pull-ups on the SDA and SCL port pin are disabled in Two-wire mode.
1	1	**Two-wire mode. Uses SDA and SCL pins.** Same operation as in two-wire mode above, except that the SCL line is also held low when a counter overflow occurs, and until the Counter Overflow Flag (USIOIF) is cleared.

Note: 1. The DI and USCK pins are renamed to *Serial Data* (SDA) and *Serial Clock* (SCL) respectively to avoid confusion between the modes of operation.

- **Bit 3:2 – USICS1:0: Clock Source Select**

These bits set the clock source for the USI Data Register and counter. The data output latch ensures that the output is changed at the opposite edge of the sampling of the data input (DI/SDA) when using external clock source (USCK/SCL). When software strobe or Timer/Counter0 Compare Match clock option is selected, the output latch is transparent and therefore the output is changed immediately.

Clearing the USICS1:0 bits enables software strobe option. When using this option, writing a one to the USICLK bit clocks both the USI Data Register and the counter. For external clock source (USICS1 = 1), the USICLK bit is no longer used as a strobe, but selects between external clocking and software clocking by the USITC strobe bit.

Table 14-2 shows the relationship between the USICS1:0 and USICLK setting and clock source used for the USI Data Register and the 4-bit counter.

Table 14-2. Relationship between the USICS1:0 and USICLK Setting

USICS1	USICS0	USICLK	Clock Source	4-bit Counter Clock Source
0	0	0	No Clock	No Clock
0	0	1	Software clock strobe (USICLK)	Software clock strobe (USICLK)
0	1	X	Timer/Counter0 Compare Match	Timer/Counter0 Compare Match
1	0	0	External, positive edge	External, both edges
1	1	0	External, negative edge	External, both edges
1	0	1	External, positive edge	Software clock strobe (USITC)
1	1	1	External, negative edge	Software clock strobe (USITC)

- **Bit 1 – USICLK: Clock Strobe**

Writing a one to this bit location strobes the USI Data Register to shift one step and the counter to increment by one, provided that the software clock strobe option has been selected by writing USICS1:0 bits to zero. The output will change immediately when the clock strobe is executed, i.e., during the same instruction cycle. The value shifted into the USI Data Register is sampled the previous instruction cycle.

When an external clock source is selected (USICS1 = 1), the USICLK function is changed from a clock strobe to a Clock Select Register. Setting the USICLK bit in this case will select the USITC strobe bit as clock source for the 4-bit counter (see Table 14-2).

The bit will be read as zero.

- **Bit 0 – USITC: Toggle Clock Port Pin**

Writing a one to this bit location toggles the USCK/SCL value either from 0 to 1, or from 1 to 0. The toggling is independent of the setting in the Data Direction Register, but if the PORT value is to be shown on the pin the corresponding DDR pin must be set as output (to one). This feature allows easy clock generation when implementing master devices.

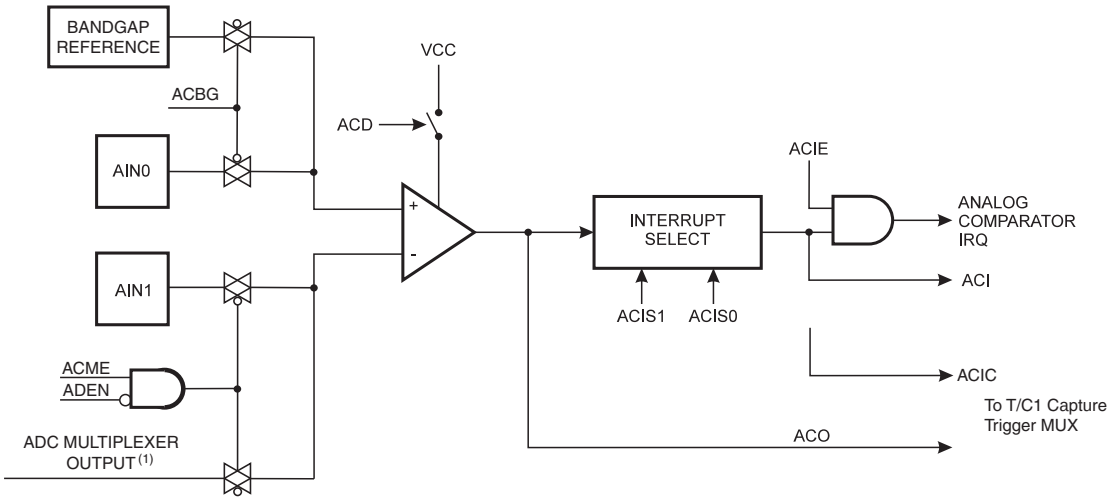
When an external clock source is selected (USICS1 = 1) and the USICLK bit is set to one, writing to the USITC strobe bit will directly clock the 4-bit counter. This allows an early detection of when the transfer is done when operating as a master device.

The bit will read as zero.

15. 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 15-1.

Figure 15-1. Analog Comparator Block Diagram



Notes: 1. See Table 15-1 on page 129.

See Figure 1-1 on page 2 and Table 10-9 on page 67 for Analog Comparator pin placement.

15.1 Analog Comparator Multiplexed Input

When the Analog to Digital Converter (ADC) is configurated as 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 15-1. If ACME is cleared or ADEN is set, AIN1 is applied to the negative input to the analog comparator.

Table 15-1. Analog Comparator Multiplexed Input

ACME	ADEN	MUX4..0	Analog Comparator Negative Input
0	X	XXXXX	AIN1
1	1	XXXXX	AIN1
1	0	00000	ADC0
1	0	00001	ADC1
1	0	00010	ADC2
1	0	00011	ADC3

Table 15-1. Analog Comparator Multiplexed Input (Continued)

ACME	ADEN	MUX4..0	Analog Comparator Negative Input
1	0	00100	ADC4
1	0	00101	ADC5
1	0	00110	ADC6
1	0	00111	ADC7

15.2 Register Description

15.2.1 ACSR – Analog Comparator Control and Status Register

Bit	7	6	5	4	3	2	1	0	
0x08 (0x28)	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	ACSR
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	N/A	0	0	0	0	0	

- **Bit 7 – ACD: Analog Comparator Disable**

When this bit is written logic one, the power to the Analog Comparator is switched off. This bit can be set at any time to turn off the Analog Comparator. This will reduce power consumption in Active and Idle mode. When changing the ACD bit, the Analog Comparator Interrupt must be disabled by clearing the ACIE bit in ACSR. Otherwise an interrupt can occur when the bit is changed.

- **Bit 6 – ACBG: Analog Comparator Bandgap Select**

When this bit is set, a fixed, internal bandgap reference voltage replaces the positive input to the Analog Comparator. When this bit is cleared, AIN0 is applied to the positive input of the Analog Comparator.

- **Bit 5 – ACO: Analog Comparator Output**

The output of the Analog Comparator is synchronized and then directly connected to ACO. The synchronization introduces a delay of 1 - 2 clock cycles.

- **Bit 4 – ACI: Analog Comparator Interrupt Flag**

This bit is set by hardware when a comparator output event triggers the interrupt mode defined by ACIS1 and ACIS0. The Analog Comparator interrupt routine is executed if the ACIE bit is set and the I-bit in SREG is set. ACI is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ACI is cleared by writing a logic one to the flag.

- **Bit 3 – ACIE: Analog Comparator Interrupt Enable**

When the ACIE bit is written logic one and the I-bit in the Status Register is set, the Analog Comparator interrupt is activated. When written logic zero, the interrupt is disabled.

- **Bit 2 – ACIC: Analog Comparator Input Capture Enable**

When written logic one, this bit enables the input capture function in Timer/Counter1 to be triggered by the Analog Comparator. The comparator output is in this case directly connected to the input capture front-end logic, making the comparator utilize the noise canceler and edge select features of the Timer/Counter1 Input Capture interrupt. When written logic zero, no connection

between the Analog Comparator and the input capture function exists. To make the comparator trigger the Timer/Counter1 Input Capture inter-rupt, the ICIE1 bit in the Timer Interrupt Mask Register (TIMSK1) must be set.

- **Bits 1, 0 – ACIS1, ACIS0: Analog Comparator Interrupt Mode Select**

These bits determine which comparator events that trigger the Analog Comparator interrupt. The different settings are shown in Table 15-2.

Table 15-2. ACIS1/ACIS0 Settings

ACIS1	ACIS0	Interrupt Mode
0	0	Comparator Interrupt on Output Toggle.
0	1	Reserved
1	0	Comparator Interrupt on Falling Output Edge.
1	1	Comparator Interrupt on Rising Output Edge.

When changing the ACIS1/ACIS0 bits, the Analog Comparator Interrupt must be disabled by clearing its Interrupt Enable bit in the ACSR Register. Otherwise an interrupt can occur when the bits are changed.

15.2.2 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	BIN	ACME	–	ADLAR	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R/W	R/W	R	R/W	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 6 – ACME: Analog Comparator Multiplexer Enable**

When this bit is written logic one and the ADC is switched off (ADEN in ADCSRA is zero), the ADC multiplexer selects the negative input to the Analog Comparator. When this bit is written logic zero, AIN1 is applied to the negative input of the Analog Comparator. For a detailed description of this bit, see “Analog Comparator Multiplexed Input” on page 129.

15.2.3 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 2, 1– ADC2D, ADC1D: ADC 2/1 Digital input buffer disable**

When this bit is written logic one, the digital input buffer on the AIN1/0 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 AIN1/0 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

16. Analog to Digital Converter

16.1 Features

- **10-bit Resolution**
- **1.0 LSB Integral Non-linearity**
- **± 2 LSB Absolute Accuracy**
- **13µs Conversion Time**
- **15 kSPS at Maximum Resolution**
- **Eight Multiplexed Single Ended Input Channels**
- **Twelve Differential Input Channels with Selectable Gain (1x, 20x)**
- **Temperature Sensor Input Channel**
- **Optional Left Adjustment for ADC Result Readout**
- **0 - V_{CC} ADC Input Voltage Range**
- **1.1V ADC Reference Voltage**
- **Free Running or Single Conversion Mode**
- **ADC Start Conversion by Auto Triggering on Interrupt Sources**
- **Interrupt on ADC Conversion Complete**
- **Sleep Mode Noise Canceler**
- **Unipolar / Bipolar Input Mode**
- **Input Polarity Reversal Mode**

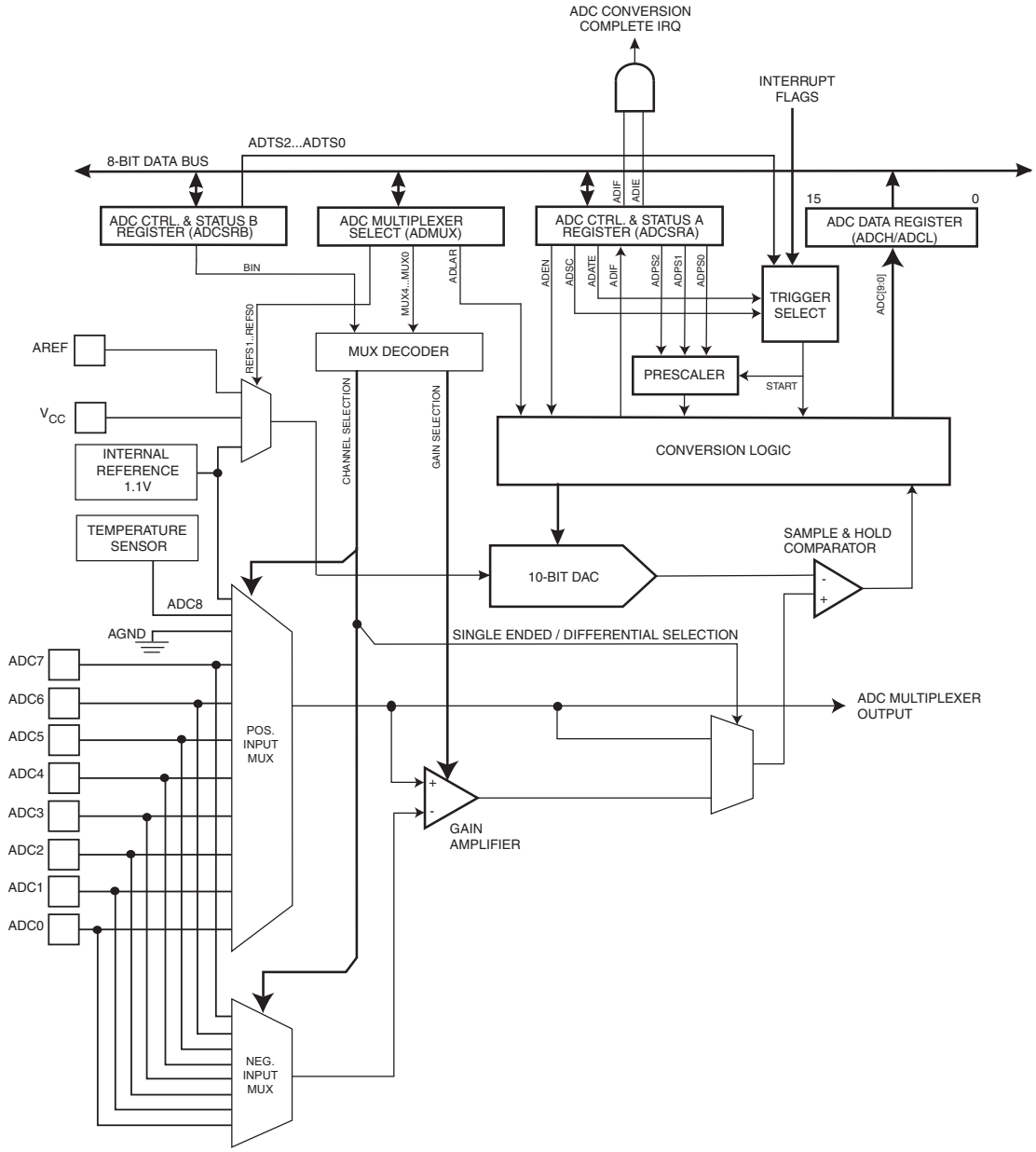
16.2 Overview

The ATtiny24/44/84 features a 10-bit successive approximation ADC. The ADC is connected to 8-pin port A for external sources. In addition to external sources internal temperature sensor can be measured by ADC. Analog Multiplexer allows eight single-ended channels or 12 differential channels from Port A. The programmable gain stage provides ampification steps 0 dB (1x) and 26 dB (20x) for 12 differential ADC channels.

The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 16-1 on page 133.

Internal reference voltage of nominally 1.1V is provided On-chip. Alternatively, V_{CC} can be used as reference voltage for single ended channels. There is also an option to use an external voltage reference and turn-off the internal voltage reference.

Figure 16-1. Analog to Digital Converter Block Schematic



16.3 Operation

The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the reference voltage.The voltage reference for the ADC may be selected by writing to the REFS1..0 bits in ADMUX. The VCC supply, the AREF pin or an internal 1.1V voltage reference may be selected as the ADC voltage reference.

The analog input channel and differential gain are selected by writing to the MUX5..0 bits in ADMUX. Any of the eight ADC input pins ADC7..0 can be selected as single ended inputs to the ADC. For differential measurements all analog inputs next to each other can be selected as a input pair. Every input is also possible to measure with ADC3. These pairs of differential inputs are measured by ADC trough the differential gain amplifier.

If differential channels are selected, the differential gain stage amplifies the voltage difference between the selected input pair by the selected gain factor, 1x or 20x, according to the setting of the MUX0 bit in ADMUX. This amplified value then becomes the analog input to the ADC. If single ended channels are used, the gain amplifier is bypassed altogether.

The offset of the differential channels can be measure by selecting the same input for both negative and positive input. Offset calibration can be done for ADC0, ADC3 and ADC7. When ADC0 or ADC3 or ADC7 is selected as both the positive and negative input to the differential gain amplifier , the remaining offset in the gain stage and conversion circuitry can be measured directly as the result of the conversion. This figure can be subtracted from subsequent conversions with the same gain setting to reduce offset error to below 1 LSB.

The on-chip temperature sensor is selected by writing the code “100010” to the MUX5..0 bits in ADMUX register.

The ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. Voltage reference and input channel selections will not go into effect until ADEN is set. The ADC does not consume power when ADEN is cleared, so it is recommended to switch off the ADC before entering power saving sleep modes.

The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted, but can optionally be presented left adjusted by setting the ADLAR bit in ADCSRB.

If the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH, to ensure that the content of the data registers belongs to the same conversion. Once ADCL is read, ADC access to data registers is blocked. This means that if ADCL has been read, and a conversion completes before ADCH is read, neither register is updated and the result from the conversion is lost. When ADCH is read, ADC access to the ADCH and ADCL Registers is re-enabled.

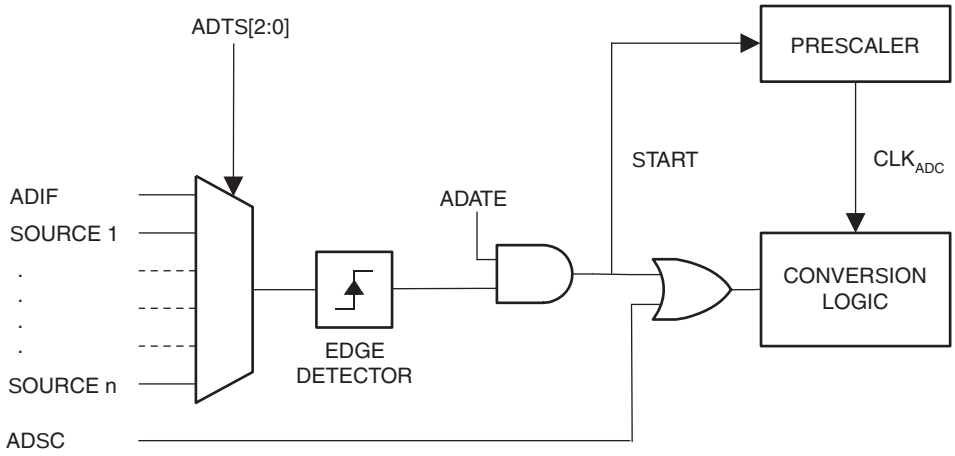
The ADC has its own interrupt which can be triggered when a conversion completes. When ADC access to the data registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result is lost.

16.4 Starting a Conversion

A single conversion is started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. If a different data channel is selected while a conversion is in progress, the ADC will finish the current conversion before performing the channel change.

Alternatively, a conversion can be triggered automatically by various sources. Auto Triggering is enabled by setting the ADC Auto Trigger Enable bit, ADATE in ADCSRA. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB (see description of the ADTS bits for a list of the trigger sources). When a positive edge occurs on the selected trigger signal, the ADC prescaler is reset and a conversion is started. This provides a method of starting conversions at fixed intervals. If the trigger signal still is set when the conversion completes, a new conversion will not be started. If another positive edge occurs on the trigger signal during conversion, the edge will be ignored. Note that an Interrupt Flag will be set even if the specific interrupt is disabled or the Global Interrupt Enable bit in SREG is cleared. A conversion can thus be triggered without causing an interrupt. However, the Interrupt Flag must be cleared in order to trigger a new conversion at the next interrupt event.

Figure 16-2. ADC Auto Trigger Logic



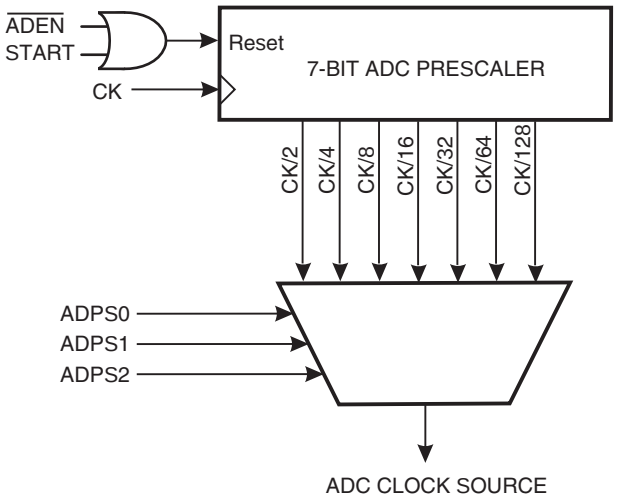
Using the ADC Interrupt Flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The ADC then operates in Free Running mode, constantly sampling and updating the ADC Data Register. The first conversion must be started by writing a logical one to the 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 one. ADSC can also be used to determine if a conversion is in progress. The ADSC bit will be read as one during a conversion, independently of how the conversion was started.

16.5 Prescaling and Conversion Timing

By default, the successive approximation circuitry requires an input clock frequency between 50 kHz and 200 kHz to get maximum resolution. If a lower resolution than 10 bits is needed, the input clock frequency to the ADC can be higher than 200 kHz to get a higher sample rate. It is not recommended to use a higher input clock frequency than 1 MHz.

Figure 16-3. ADC Prescaler

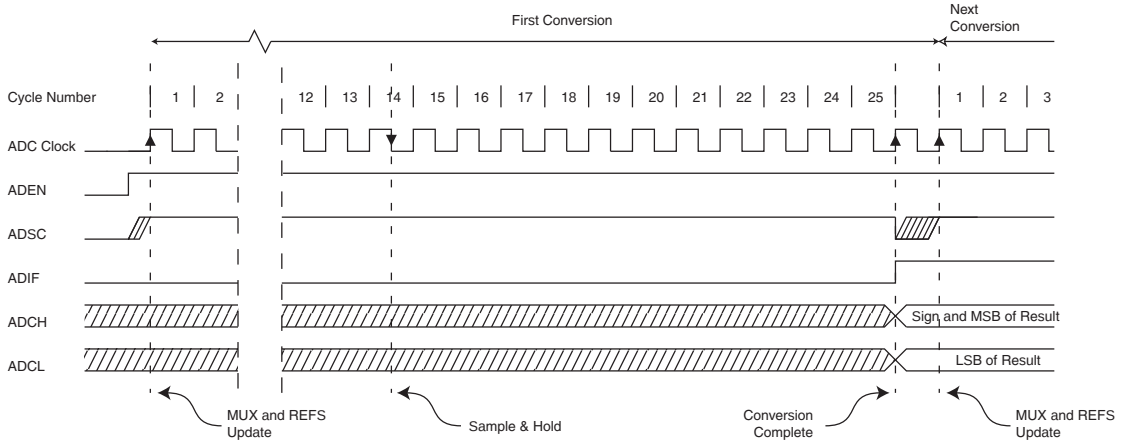


The ADC module contains a prescaler, as illustrated in Figure 16-3 on page 135, which generates an acceptable ADC clock frequency from any CPU frequency above 100 kHz. The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by setting the 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.

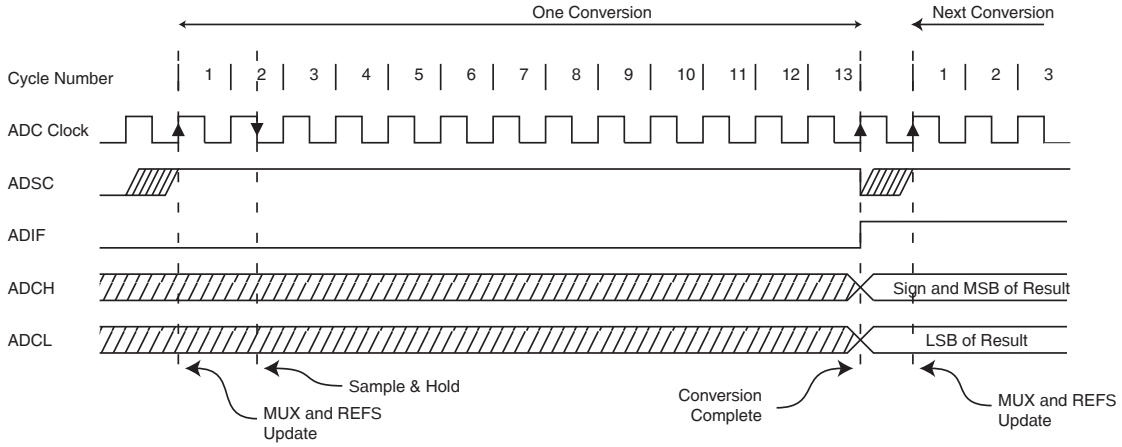
A normal conversion takes 13 ADC clock cycles. The first conversion after the ADC is switched on (ADEN in ADCSRA is set) takes 25 ADC clock cycles in order to initialize the analog circuitry, as shown in Figure 16-4 below.

Figure 16-4. ADC Timing Diagram, First Conversion (Single Conversion Mode)



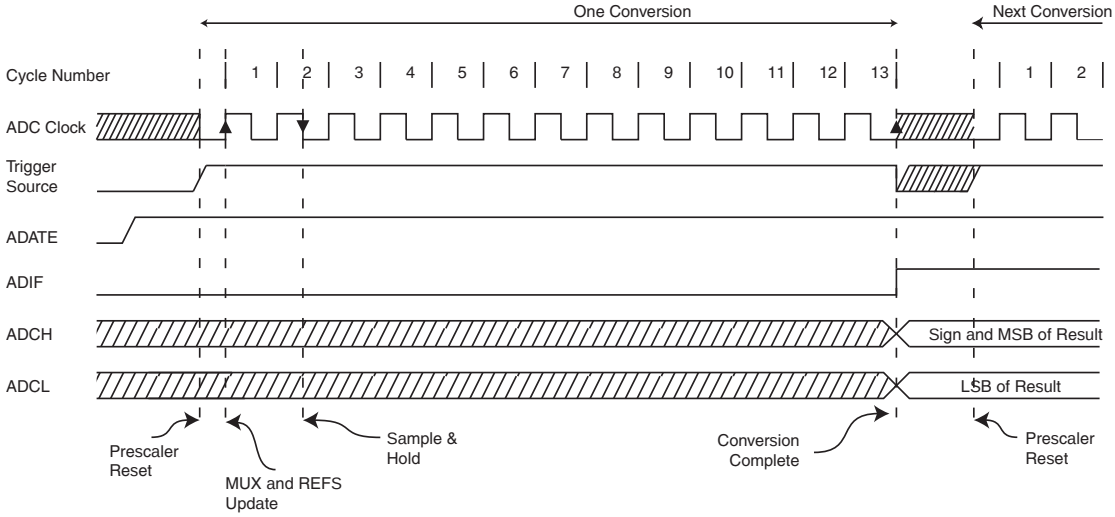
The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 13.5 ADC clock cycles after the start of an first conversion. When a conversion is complete, the result is written to the ADC Data Registers, and ADIF is set. In Single Conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

Figure 16-5. ADC Timing Diagram, Single Conversion



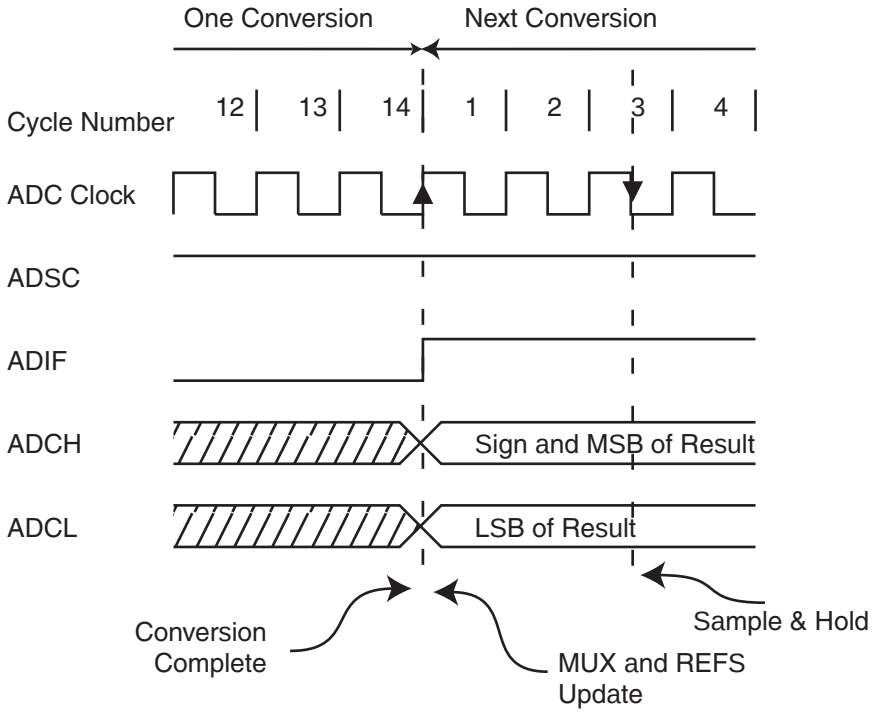
When Auto Triggering is used, the prescaler is reset when the trigger event occurs, as shown in Figure 16-6 below. This assures a fixed delay from the trigger event to the start of conversion. In this mode, the sample-and-hold takes place two ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic.

Figure 16-6. ADC Timing Diagram, Auto Triggered Conversion



In Free Running mode, a new conversion will be started immediately after the conversion completes, while ADSC remains high. See Figure 16-7.

Figure 16-7. ADC Timing Diagram, Free Running Conversion



For a summary of conversion times, see Table 16-1.

Table 16-1. ADC Conversion Time

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	13.5	25
Normal conversions	1.5	13
Auto Triggered conversions	2	13.5
Free Running conversion	2.5	14

16.6 Changing Channel or Reference Selection

The MUX5:0 and REFS1:0 bits in the ADMUX Register are single buffered through a temporary register to which the CPU has random access. This ensures that the channels and reference selection only takes place at a safe point during the conversion. The channel and reference selection is continuously updated until a conversion is started. Once the conversion starts, the channel and reference selection is locked to ensure a sufficient sampling time for the ADC. Continuous updating resumes in the last ADC clock cycle before the conversion completes (ADIF in ADCSRA is set). Note that the conversion starts on the following rising ADC clock edge after ADSC is written. The user is thus advised not to write new channel or reference selection values to ADMUX until one ADC clock cycle after ADSC is written.

If Auto Triggering is used, the exact time of the triggering event can be indeterministic. Special care must be taken when updating the ADMUX Register, in order to control which conversion will be affected by the new settings.

If both ADATE and ADEN is written to one, an interrupt event can occur at any time. If the ADMUX Register is changed in this period, the user cannot tell if the next conversion is based on the old or the new settings. ADMUX can be safely updated in the following ways:

- When ADATE or ADEN is cleared.
- During conversion, minimum one ADC clock cycle after the trigger event.
- After a conversion, before the Interrupt Flag used as trigger source is cleared.

When updating ADMUX in one of these conditions, the new settings will affect the next ADC conversion.

16.6.1 ADC Input Channels

When changing channel selections, the user should observe the following guidelines to ensure that the correct channel is selected:

In Single Conversion mode, always select the channel before starting the conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the conversion to complete before changing the channel selection.

In Free Running mode, always select the channel before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the first conversion to complete, and then change the channel

selection. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. Subsequent conversions will reflect the new channel selection.

16.6.2 ADC Voltage Reference

The reference voltage for the ADC (V_{REF}) indicates the conversion range for the ADC. Single ended channels that exceed V_{REF} will result in codes close to 0x3FF. V_{REF} can be selected as either V_{CC}, or internal 1.1V reference, or external AREF pin. The first ADC conversion result after switching reference voltage source may be inaccurate, and the user is advised to discard this result.

16.7 ADC Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode. This reduces noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC Noise Reduction and Idle mode. To make use of this feature, the following procedure should be used:

- Make sure that the ADC is enabled and is not busy converting. Single Conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
- Enter ADC Noise Reduction mode (or Idle mode). The ADC will start a conversion once the CPU has been halted.
- If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC Conversion Complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC Conversion Complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not automatically be turned off when entering other sleep modes than Idle mode and ADC Noise Reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption.

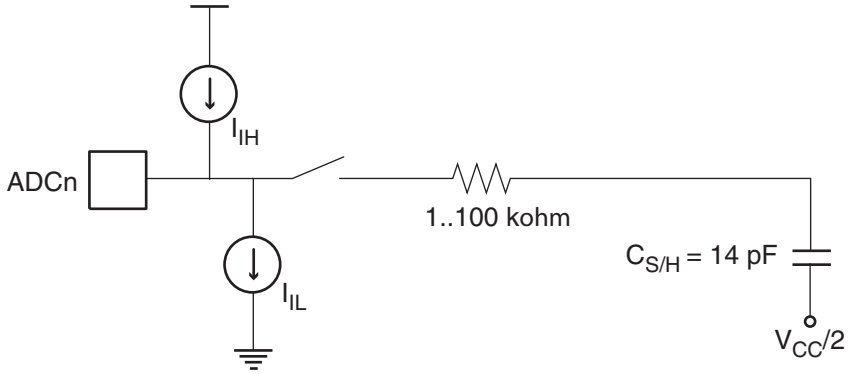
16.8 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 16-8 on page 140. 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. 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 time the source needs to charge the S/H capacitor, which can vary widely. With slowly varying signals the user is recommended to use sources with low impedance, only, 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 16-8. Analog Input Circuitry



Note: The capacitor in the figure depicts the total capacitance, including the sample/hold capacitor and any stray or parasitic capacitance inside the device. The value given is worst case.

16.9 Noise Canceling Techniques

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

- Keep analog signal paths as short as possible.
- Make sure analog tracks run over the analog ground plane.
- Keep analog tracks well away from high-speed switching digital tracks.
- If any port pin is used as a digital output, it mustn't switch while a conversion is in progress.
- Place bypass capacitors as close to V_{CC} and GND pins as possible.

Where high ADC accuracy is required it is recommended to use ADC Noise Reduction Mode, as described in Section 16.7 on page 139. This is especially the case when system clock frequency is above 1 MHz, or when the ADC is used for reading the internal temperature sensor, as described in Section 16.12 on page 144. A good system design with properly placed, external bypass capacitors does reduce the need for using ADC Noise Reduction Mode

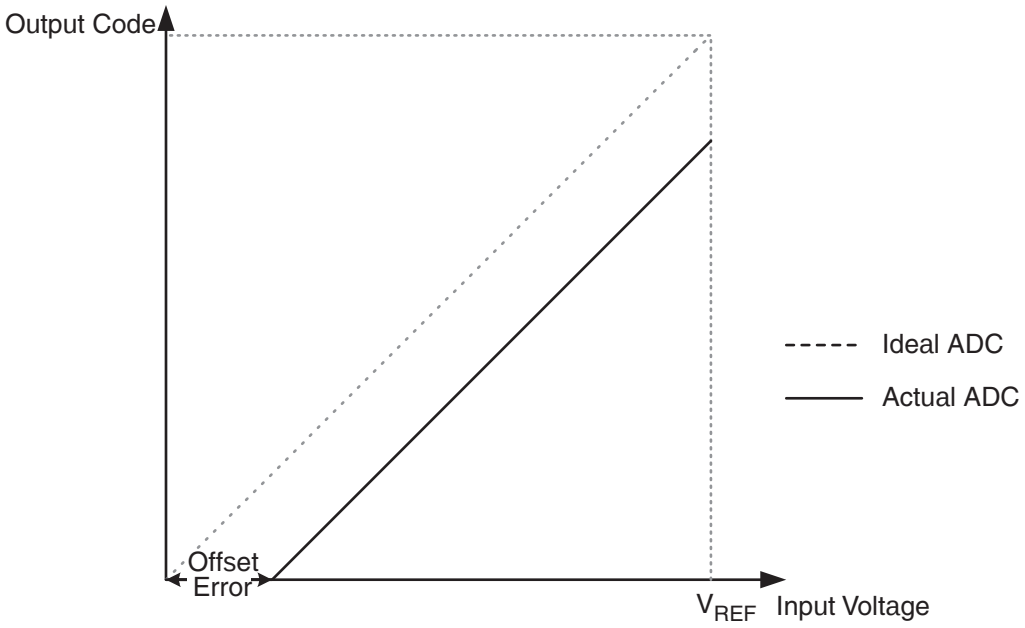
16.10 ADC Accuracy Definitions

An n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSBs). The lowest code is read as 0, and the highest code is read as 2^n-1.

Several parameters describe the deviation from the ideal behavior, as follows:

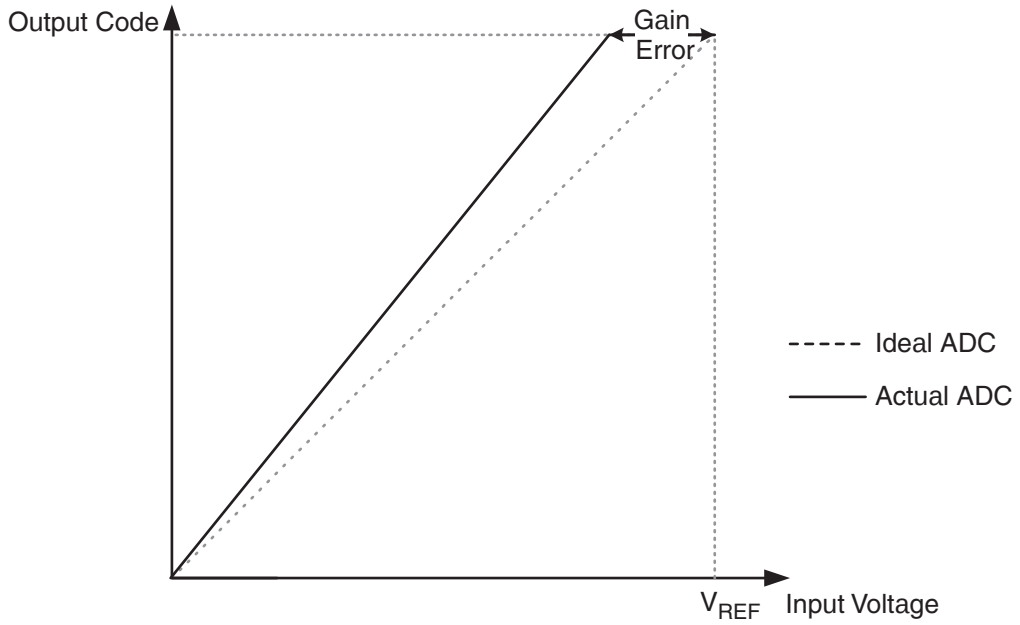
- Offset: The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSB). Ideal value: 0 LSB.

Figure 16-9. Offset Error



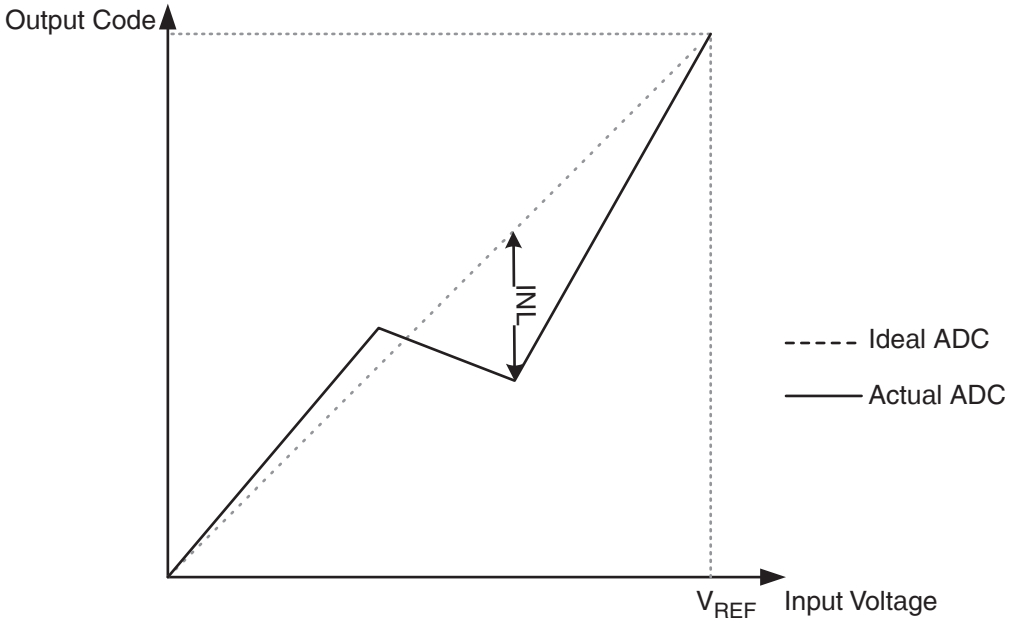
- Gain Error: After adjusting for offset, the Gain Error is found as the deviation of the last transition (0x3FE to 0x3FF) compared to the ideal transition (at 1.5 LSB below maximum). Ideal value: 0 LSB

Figure 16-10. Gain Error



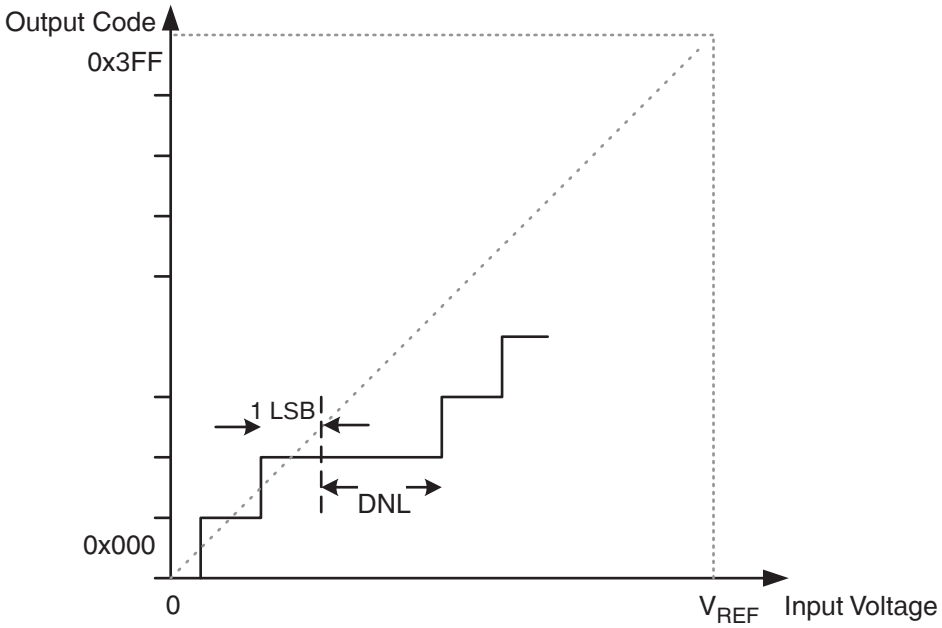
- Integral Non-linearity (INL): After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSB.

Figure 16-11. Integral Non-linearity (INL)



- Differential Non-linearity (DNL): The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSB). Ideal value: 0 LSB.

Figure 16-12. Differential Non-linearity (DNL)



- Quantization Error: Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSB wide) will code to the same value. Always ± 0.5 LSB.

- Absolute Accuracy: The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of offset, gain error, differential error, non-linearity, and quantization error. Ideal value: ± 0.5 LSB.

16.11 ADC Conversion Result

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC Result Registers (ADCL, ADCH). The form of the conversion result depends on the type of the conversio as there are three types of conversions: single ended conversion, unipolar differential conversion and bipolar differential conversion.

16.11.1 Single Ended Conversion

For single ended conversion, the result is

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference (see Table 16-3 on page 145 and Table 16-4 on page 146). 0x000 represents analog ground, and 0x3FF represents the selected reference voltage minus one LSB. The result is presented in one-sided form, from 0x3FF to 0x000.

16.11.2 Unipolar Differential Conversion

If differential channels and an unipolar input mode are used, the result is

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot 1024}{V_{REF}} \cdot GAIN$$

where V_{POS} is the voltage on the positive input pin, V_{NEG} the voltage on the negative input pin, and V_{REF} the selected voltage reference. The voltage of the positive pin must always be larger than the voltage of the negative pin or otherwise the voltage difference is saturated to zero. The result is presented in one-sided form, from 0x000 (0d) through 0x3FF (+1023d). The GAIN is either 1x or 20x.

16.11.3 Bipolar Differential Conversion

If differential channels and a bipolar input mode are used, the result is

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot 512}{V_{REF}} \cdot GAIN$$

where V_{POS} is the voltage on the positive input pin, V_{NEG} the voltage on the negative input pin, and V_{REF} the selected voltage reference. The result is presented in two's complement form, from 0x200 (-512d) through 0x1FF (+511d). The GAIN is either 1x or 20x. Note that if the user wants

to perform a quick polarity check of the result, it is sufficient to read the MSB of the result (ADC9 in ADCH). If the bit is one, the result is negative, and if this bit is zero, the result is positive.

As default the ADC converter operates in the unipolar input mode, but the bipolar input mode can be selected by writting the BIN bit in the ADCSRB to one. In the bipolar input mode two-sided voltage differences are allowed and thus the voltage on the negative input pin can also be larger than the voltage on the positive input pin.

16.12 Temperature Measurement

The temperature measurement is based on an on-chip temperature sensor that is coupled to a single ended ADC8 channel. Selecting the ADC8 channel by writing the MUX5:0 bits in ADMUX register to “100010” enables the temperature sensor. The internal 1.1V reference must also be selected for the ADC reference source in the temperature sensor measurement. When the temperature sensor is enabled, the ADC converter can be used in single conversion mode to measure the voltage over the temperature sensor.

The measured voltage has a linear relationship to the temperature as described in Table 16-2 The sensitivity is approximately 1 LSB / °C and the accuracy depends on the method of user calibration. Typically, the measurement accuracy after a single temperature calibration is ±10°C, assuming calibration at room temperature. Better accuracies are achieved by using two temperature points for calibration.

Table 16-2. Temperature vs. Sensor Output Voltage (Typical Case)

Temperature	**-40°C**	**+25°C**	**+85°C**
ADC	230 LSB	300 LSB	370 LSB

The values described in Table 16-2 are typical values. However, due to 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 sofware calibration can be done using the formula:

$$T = k * [(ADCH << 8) \mid ADCL] + T_{OS}$$

where ADCH and ADCL are the ADC data registers, k is the fixed slope coefficient and T_{OS} is the temperature sensor offset. Typically, k is very close to 1.0 and in single-point calibration the coefficient may be omitted. Where higher accuracy is required the slope coefficient should be evaluated based on measurements at two temperatures.

16.13 Register Description

16.13.1 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 16-3.

Table 16-3. 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 reference turned off
1	0	Internal 1.1V voltage reference
1	1	Reserved

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). Also note, that when these bits are changed, the next conversion will take 25 ADC clock cycles.

Special care should be taken when changing differential channels. Once a differential channel has been selected the input stage may take a while to stabilize. It is therefore recommended to force the ADC to perform a long conversion when changing multiplexer or voltage reference settings. This can be done by first turning off the ADC, then changing reference settings and then turn on the ADC. Alternatively, the first conversion results after changing reference settings should be discarded.

It is not recommended to use an external AREF higher than (V_{CC} - 1V) for channels with differential gain, as this will affect ADC accuracy.

Internal voltage reference options may not be used if an external voltage is being applied to the AREF pin.

- **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 case of differential input, gain selection is also made with these bits. Selections on Table 16-4 on page 146 show values for single endid channels and where the differential channels as well as the offset calibration selections are located. Selecting the single-ended channel ADC8 enables the temperature measurement. See Table 16-4 on page 146 for 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).

Special care should be taken when changing differential channels. Once a differential channel has been selected the input stage may take a while to stabilize. It is therefore recommended to force the ADC to perform a long conversion when changing multiplexer settings. This can be

done by first turning off the ADC, then changing multiplexer settings and then turn on the ADC. Alternatively, the first conversion results after changing multiplexer settings should be discarded.

Table 16-4. Single-Ended Input channel Selections.

Single Ended Input	MUX5:0
ADC0 (PA0)	000000
ADC1 (PA1)	000001
ADC2 (PA2)	000010
ADC3 (PA3)	000011
ADC4 (PA4)	000100
ADC5 (PA5)	000101
ADC6 (PA6)	000110
ADC7 (PA7)	000111
Reserved for differential channels(1)	001000 - 011111
0V (AGND)	100000
1.1V (I Ref)	100001
ADC8(2)	100010
Reserved for offset calibration(3)	100011 - 100111
Reserved for reversal differential channels(1)	101000 - 111111

Notes: 1. See Table 16-5 for details.
2. See “Temperature Measurement” on page 144.
3. For offset calibration, only. See Table 16-5 on page 146 and “Operation” on page 133.

See Table 16-5 for details of selections of differential input channel selections as well as selections of offset calibration channels. MUX0 bit works as a gain selection bit for differential channels. When MUX0 is cleared (‘0’) 1x gain is selected and when it is set (‘1’) 20x gain is selected. For normal differential channel pairs MUX5 bit work as a polarity reversal bit. Toggling of the MUX5 bit exhanges the positive and negative channel other way a round.

Table 16-5. Differential Input channel Selections.

Positive Differential Input	Negative Differential Input	MUX5:0	
		Gain 1x	Gain 20x
ADC0 (PA0)	ADC0 (PA0) (1)	N/A	100011
	ADC1 (PA1)	001000	001001
	ADC3 (PA3)	001010	001011
ADC1 (PA1)	ADC0 (PA0)	101000	101001
	ADC2 (PA2)	001100	001101
	ADC3 (PA3)	001110	001111
ADC2 (PA2)	ADC1 (PA1)	101100	101101
	ADC3 (PA3)	010000	010001

Table 16-5. Differential Input channel Selections. (Continued)

Positive Differential Input	Negative Differential Input	MUX5:0	
		Gain 1x	Gain 20x
ADC3 (PA3)	ADC0 (PA0)	101010	101011
	ADC1 (PA1)	101110	101111
	ADC2 (PA2)	110000	110001
	ADC3 (PA3)(1)	100100	100101
	ADC4 (PA4	010010	010011
	ADC5 (PA5)	010100	010101
	ADC6 (PA6)	010110	010111
	ADC7 (PA7)	011000	011001
ADC4 (PA4)	ADC3 (PA3)	110010	110011
	ADC5 (PA5)	011010	011011
ADC5 (PA5)	ADC3 (PA3)	110100	110101
	ADC4 (PA4)	111010	111011
	ADC6 (PA6)	011100	011101
ADC6 (PA6)	ADC3 (PA3)	110110	110111
	ADC5 (PA5)	111100	111101
	ADC7 (PA7)	011110	011111
ADC7 (PA7)	ADC3 (PA3)	111000	111001
	ADC6 (PA6)	111110	111111
	ADC7 (PA7)(1)	100110	100111

1. For offset calibration, only. See “Operation” on page 133.

For offset calibration purpose the offset of the certain differential channels can be measure by selecting the same input for both negative and positive input. This calibration can be done for ADC0, ADC3 and ADC7. “Operation” on page 133 describes offset calibration in a more detailed level.

16.13.2 ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
0x06 (0x26)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
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 – ADEN: ADC Enable**

Writing this bit to one enables the ADC. By writing it to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion.

- **Bit 6 – ADSC: ADC Start Conversion**

In Single Conversion mode, write this bit to one to start each conversion. In Free Running mode, write this bit to one to start the first conversion. The first conversion after ADSC has been written

after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, will take 25 ADC clock cycles instead of the normal 13. This first conversion performs initialization of the ADC.

ADSC will read as one as long as a conversion is in progress. When the conversion is complete, it returns to zero. Writing zero to this bit has no effect.

- **Bit 5 – ADATE: ADC Auto Trigger Enable**

When this bit is written to one, Auto Triggering of the ADC is enabled. The ADC will start a conversion on a positive edge of the selected trigger signal. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB.

- **Bit 4 – ADIF: ADC Interrupt Flag**

This bit is set when an ADC conversion completes and the data registers are updated. The ADC Conversion Complete Interrupt is executed if the ADIE bit and the I-bit in SREG are set. ADIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ADIF is cleared by writing a logical one to the flag. Beware that if doing a Read-Modify-Write on ADCSRA, a pending interrupt can be disabled. This also applies if the SBI instruction is used.

- **Bit 3 – ADIE: ADC Interrupt Enable**

When this bit is written to one and the I-bit in SREG is set, the ADC Conversion Complete Interrupt is activated.

- **Bits 2:0 – ADPS2:0: ADC Prescaler Select Bits**

These bits determine the division factor between the system clock frequency and the input clock to the ADC.

Table 16-6. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

16.13.3 ADCL and ADCH – ADC Data Register

16.13.3.1 ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
0x05 (0x25)	–	–	–	–	–	–	ADC9	ADC8	ADCH
0x04 (0x24)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

16.13.3.2 ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
0x05 (0x25)	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
0x04 (0x24)	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers.

When ADCL is read, the ADC Data Register is not updated until ADCH is read. Consequently, if the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH.

The ADLAR bit in ADCSRB, and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set, the result is left adjusted. If ADLAR is cleared (default), the result is right adjusted.

- **ADC9:0: ADC Conversion Result**

These bits represent the result from the conversion, as detailed in “ADC Conversion Result” on page 143.

16.13.4 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	BIN	ACME	–	ADLAR	–	ADTS2	ADTS1	ADTS0	ADCSRB
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 – BIN: Bipolar Input Mode**

The gain stage is working in the unipolar mode as default, but the bipolar mode can be selected by writing the BIN bit in the ADCSRB register. In the unipolar mode only one-sided conversions are supported and the voltage on the positive input must always be larger than the voltage on the negative input. Otherwise the result is saturated to the voltage reference. In the bipolar mode two-sided conversions are supported and the result is represented in the two’s complement form. In the unipolar mode the resolution is 10 bits and the bipolar mode the resolution is 9 bits + 1 sign bit.

- **Bit 6 – ACME: Analog Comparator Multiplexer Enable**

See “ADCSRB – ADC Control and Status Register B” on page 131.

- **Bit 5 – Res: Reserved Bit**

This is a reserved bit in ATtiny24/44/84. For compatibility with future devices always write this bit to zero.

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

The ADLAR bit affects the presentation of the ADC conversion result in the ADC Data Register. Write 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 comple the description of this bit, see “ADCL and ADCH – ADC Data Register” on page 149.

- **Bit 3 – Res: Reserved Bit**

This bit is reserved bit in the ATtiny24/44/84 and will always read as what was wrote there.

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

If ADATE in ADCSRA is written to 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 16-7. 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

16.13.5 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 a bit is written logic 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 logic one to reduce power consumption in the digital input buffer.

17. debugWIRE On-chip Debug System

17.1 Features

- **Complete Program Flow Control**
- **Emulates All On-chip Functions, Both Digital and Analog , except RESET Pin**
- **Real-time Operation**
- **Symbolic Debugging Support (Both at C and Assembler Source Level, or for Other HLLs)**
- **Unlimited Number of Program Break Points (Using Software Break Points)**
- **Non-intrusive Operation**
- **Electrical Characteristics Identical to Real Device**
- **Automatic Configuration System**
- **High-Speed Operation**
- **Programming of Non-volatile Memories**

17.2 Overview

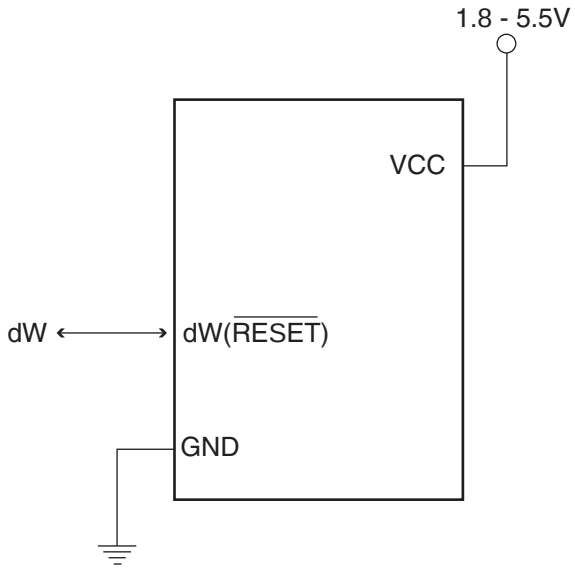
The debugWIRE On-chip debug system uses a One-wire, bi-directional interface to control the program flow, execute AVR instructions in the CPU and to program the different non-volatile memories.

17.3 Physical Interface

When the debugWIRE Enable (DWEN) Fuse is programmed and Lock bits are unprogrammed, the debugWIRE system within the target device is activated. The RESET port pin is configured as a wire-AND (open-drain) bi-directional I/O pin with pull-up enabled and becomes the communication gateway between target and emulator.

Figure 17-1 shows the schematic of a target MCU, with debugWIRE enabled, and the emulator connector. The system clock is not affected by debugWIRE and will always be the clock source selected by the CKSEL Fuses.

Figure 17-1. The debugWIRE Setup



When designing a system where debugWIRE will be used, the following must be observed:

- Pull-Up resistor on the dW/(RESET) line must be in the range of 10k to 20 kΩ. However, the pull-up resistor is optional.
- Connecting the RESET pin directly to V_{CC} will not work.
- Capacitors inserted on the RESET pin must be disconnected when using debugWire.
- All external reset sources must be disconnected.

17.4 Software Break Points

debugWIRE supports Program memory Break Points by the AVR Break instruction. Setting a Break Point in AVR Studio® will insert a BREAK instruction in the Program memory. The instruction replaced by the BREAK instruction will be stored. When program execution is continued, the stored instruction will be executed before continuing from the Program memory. A break can be inserted manually by putting the BREAK instruction in the program.

The Flash must be re-programmed each time a Break Point is changed. This is automatically handled by AVR Studio through the debugWIRE interface. The use of Break Points will therefore reduce the Falsh Data retention. Devices used for debugging purposes should not be shipped to end customers.

17.5 Limitations of debugWIRE

The debugWIRE communication pin (dW) is physically located on the same pin as External Reset (RESET). An External Reset source is therefore not supported when the debugWIRE is enabled.

The debugWIRE system accurately emulates all I/O functions when running at full speed, i.e., when the program in the CPU is running. When the CPU is stopped, care must be taken while accessing some of the I/O Registers via the debugger (AVR Studio). See the debugWIRE documentation for detailed description of the limitations.

The debugWIRE interface is asynchronous, which means that the debugger needs to synchronize to the system clock. If the system clock is changed by software (e.g. by writing CLKPS bits) communication via debugWIRE may fail. Also, clock frequencies below 100 kHz may cause communication problems.

A programmed DWEN Fuse enables some parts of the clock system to be running in all sleep modes. This will increase the power consumption while in sleep. Thus, the DWEN Fuse should be disabled when debugWire is not used.

17.6 Register Description

The following section describes the registers used with the debugWire.

17.6.1 DWDR – debugWire Data Register

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	DWDR[7:0]								DWDR
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 DWDR Register provides a communication channel from the running program in the MCU to the debugger. This register is only accessible by the debugWIRE and can therefore not be used as a general purpose register in the normal operations.

18. Self-Programming the Flash

The device provides a Self-Programming mechanism 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 the Program memory. The SPM instruction is disabled by default but it can be enabled by programming the SELFPRGEN fuse (to “0”).

The 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 temporary page buffer
- Perform a Page Erase
- Perform a Page Write

Alternative 2, fill the buffer after Page Erase

- Perform a Page Erase
- Fill 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, 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 since 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.

18.1 Performing Page Erase by SPM

To execute 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 is 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.

Note: The CPU is halted during the Page Erase operation.

18.2 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.

18.3 Performing a Page Write

To execute 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 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

Note: The CPU is halted during the Page Write operation.

18.4 Addressing the Flash During Self-Programming

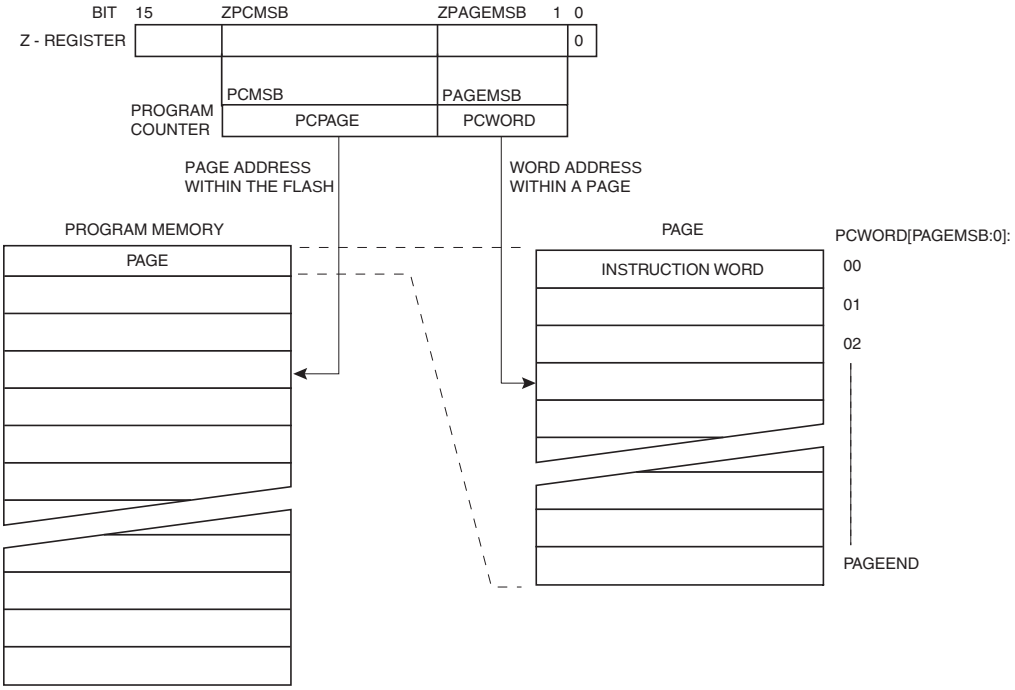
The Z-pointer is used to address the SPM commands.

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

Since the Flash is organized in pages (see Table 19-8 on page 162), 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 19-1 on page 163. 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. Since this instruction addresses the Flash byte-by-byte, also the LSB (bit Z0) of the Z-pointer is used.

Figure 18-1. Addressing the Flash During SPM



Note: The variables used in Figure 18-1 are listed in Table 19-8 on page 162.

18.5 EEPROM Write Prevents Writing to SPMCSR

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

18.6 Reading Lock, Fuse and Signature Data from Software

It is possible for firmware to read device fuse and lock bits. In addition, firmware can also read data from the device signature imprint table (see page 161).

Note: Fuse and Lock bits that are programmed, will be read as zero. Fuse and Lock bits that are unprogrammed, will be read as one.

18.6.1 Reading Lock Bits from Firmware

Lock bit values are returned in the destination register after an LPM instruction has been issued within three CPU cycles after RFLB and SELFPRGEN bits have been set in SPMCSR. The RFLB and SELFPRGEN bits automatically clear upon completion of reading the lock bits, or if no LPM instruction is executed within three CPU cycles, or if no SPM instruction is executed within four CPU cycles. When RFLB and SELFPRGEN are cleared LPM functions normally.

To read the lock bits, follow the below procedure:

1. Load the Z-pointer with 0x0001.
2. Set RFLB and SELFPRGEN bits in SPMCSR.
3. Issue an LPM instruction within three clock cycles.
4. Read the lock bits from the LPM destination register.

If successful, the contents of the destination register are as follows.

Bit	7	6	5	4	3	2	1	0
Rd	–	–	–	–	–	–	LB2	LB1

See section “Program And Data Memory Lock Bits” on page 159 for more information.

18.6.2 Reading Fuse Bits from Firmware

The algorithm for reading fuse bytes is similar to the one described above for reading lock bits, only the addresses are different. To read the Fuse Low Byte (FLB), follow the below procedure:

1. Load the Z-pointer with 0x0000.
2. Set RFLB and SELFPRGEN bits in SPMCSR.
3. Issue an LPM instruction within three clock cycles.
4. Read the FLB from the LPM destination register.

If successful, the contents of the destination register are as follows.

Bit	7	6	5	4	3	2	1	0
Rd	FLB7	FLB6	FLB5	FLB4	FLB3	FLB2	FLB1	FLB0

Refer to Table 19-5 on page 161 for a detailed description and mapping of the Fuse Low Byte.

To read the Fuse High Byte (FHB), simply replace the address in the Z-pointer with 0x0003 and repeat the procedure above. If successful, the contents of the destination register are as follows.

Bit	7	6	5	4	3	2	1	0
Rd	FHB7	FHB6	FHB5	FHB4	FHB3	FHB2	FHB1	FHB0

Refer to Table 19-4 on page 160 for detailed description and mapping of the Fuse High Byte.

To read the Fuse Extended Byte (FEB), replace the address in the Z-pointer with 0x0002 and repeat the previous procedure. If successful, the contents of the destination register are as follows.

Bit	7	6	5	4	3	2	1	0
Rd	FEB7	FEB6	FEB5	FEB4	FEB3	FEB2	FEB1	FEB0

Refer to Table 19-3 on page 160 for detailed description and mapping of the Fuse Extended Byte.

18.6.3 Reading Device Signature Imprint Table from Firmware

To read the contents of the device signature imprint table, follow the below procedure:

1. Load the Z-pointer with the table index.
2. Set RSIG and SPMEN bits in SPMCSR.
3. Issue an LPM instruction within three clock cycles.
4. Wait three clock cycles for SPMEN bits to be cleared.
5. Read table data from the LPM destination register.

The RSIG and SPMEN bits will auto-clear after three CPU cycles. When RSIG and SPMEN are cleared, LPM will work as described in the “AVR Instruction Set” description.

See program example below.

Assembly Code Example

```
DSIT_read:
  ; Uses Z-pointer as table index
  ldi  ZH, 0
  ldi  ZL, 1
  ; Preload SPMCSR bits into R16, then write to SPMCSR
  ldi  r16, (1<<RSIG)|(1<<SPMEN)
  out  SPMCSR, r16
  ; Issue LPM. Table data will be returned into r17
  lpm  r17, Z
  ret
```

Note: See “Code Examples” on page 6.

If successful, the contents of the destination register are as described in section “Device Signature Imprint Table” on page 161.

18.7 Preventing Flash Corruption

During periods of low V_{CC}, the Flash program can be corrupted because the supply voltage is too low for the CPU and the Flash to operate properly. These issues are the same as for board level systems using the Flash, and the same design solutions should be applied.

A Flash program corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the Flash requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage for executing instructions is too low.

Flash corruption can easily be avoided by following these design recommendations (one is sufficient):

1. 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 operating voltage matches the detection level. If not, 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 that the power supply voltage is sufficient.
2. Keep the AVR core in Power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR Register and thus the Flash from unintentional writes.

18.8 Programming Time for Flash when Using SPM

The calibrated RC Oscillator is used to time Flash accesses. Table 18-1 shows the typical programming time for Flash accesses from the CPU.

Table 18-1. SPM Programming Time(1)

Symbol	Min Programming Time	Max Programming Time
Flash write (Page Erase, Page Write, and write Lock bits by SPM)	3.7 ms	4.5 ms

Note: 1. The min and max programming times is per individual operation.

18.9 Register Description

18.9.1 SPMCSR – Store Program Memory Control and Status Register

The Store Program Memory Control and Status Register contains the control bits needed to control the Program memory operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	–	–	RSIG	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..6 – Res: Reserved Bits**

These bits are reserved and always read as zero.

- **Bit 5 – RSIG: Read Device Signature Imprint Table**

Issuing an LPM instruction within three cycles after RSIG and SPMEN bits have been set in SPMCSR will return the selected data (depending on Z-pointer value) from the device signature imprint table into the destination register. See “Device Signature Imprint Table” on page 161 for details.

- **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 the SPMCSR Register, will read either the Lock bits or the Fuse bits (depending on Z0 in the Z-pointer) into the destination register. See “EEPROM Write Prevents Writing to SPMCSR” on page 155 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 one at the same time as SPMEN, the next SPM instruction within four clock cycles executes 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 Write operation.

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

This bit enables the SPM instruction for the next four clock cycles. If set to one together with RSIG, CTPB, RFLB, PGWRT or PGERS, the following LPM/SPM instruction will have a special meaning, as described elsewhere.

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.

19. Memory Programming

This section describes the different methods for programming ATtiny24/44/84 memories.

19.1 Program And Data Memory Lock Bits

The ATtiny24/44/84 provides two lock bits which can be left unprogrammed (“1”) or can be programmed (“0”) to obtain the additional security listed in Table 19-2. The lock bits can only be erased to “1” with the Chip Erase command.

The device has no separate boot loader section. The SPM instruction is enabled for the whole Flash, if the SELFPROGEN fuse is programmed (“0”), otherwise it is disabled.

Program memory can be read out via the debugWIRE interface when the DWEN fuse is programmed, even if lock bits are set. Thus, when lock bit security is required, debugWIRE should always be disabled by clearing the DWEN fuse.

Table 19-1. Lock Bit Byte

Lock Bit	Bit No	Description	Default Value
	7	–	1 (unprogrammed)
	6	–	1 (unprogrammed)
	5	–	1 (unprogrammed)
	4	–	1 (unprogrammed)
	3	–	1 (unprogrammed)
	2	–	1 (unprogrammed)
LB2	1	Lock bit	1 (unprogrammed)
LB1	0	Lock bit	1 (unprogrammed)

Note: “1” means unprogrammed, “0” means programmed.

Table 19-2. Lock Bit Protection Modes.

Memory Lock Bits (1) (2)			Protection Type
LB Mode	LB2	LB1	
1	1	1	No memory lock features enabled.
2	1	0	Further programming of the Flash and EEPROM is disabled in High-voltage and Serial Programming mode. The Fuse bits are locked in both Serial and High-voltage Programming mode.(1) debugWire is disabled.
3	0	0	Further programming and verification of the Flash and EEPROM is disabled in High-voltage and Serial Programming mode. The Fuse bits are locked in both Serial and High-voltage Programming mode.(1) debugWire is disabled.

Notes: 1. Program fuse bits before programming LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed

Lock bits can also be read by device firmware. See section “Reading Lock, Fuse and Signature Data from Software” on page 155.

19.2 Fuse Bytes

The ATtiny24/44/84 have three fuse bytes. Table 19-3, Table 19-4 and Table 19-5 briefly describe the functionality of all the fuses and how they are mapped into the fuse bytes. Note that the fuses are read as logical zero, “0”, if they are programmed..

Table 19-3. Fuse Extended Byte

Fuse Extended Byte	Bit No	Description	Default Value
	7	-	1 (unprogrammed)
	6	-	1 (unprogrammed)
	5	-	1 (unprogrammed)
	4	-	1 (unprogrammed)
	3	-	1 (unprogrammed)
	2	-	1 (unprogrammed)
	1	-	1 (unprogrammed)
SELFPRGEN (1)	0	Self-Programming Enable	1 (unprogrammed)

Notes: 1. Enables SPM instruction. See “Self-Programming the Flash” on page 153.

Table 19-4. Fuse High Byte

Fuse High Byte	Bit No	Description	Default Value
RSTDISBL (1)	7	External Reset disable	1 (unprogrammed)
DWEN (2)	6	DebugWIRE Enable	1 (unprogrammed)
SPIEN (3)	6	Enable Serial Program and Data Downloading	0 (programmed, SPI prog. enabled)
WDTON (4)	4	Watchdog Timer always on	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed, EEPROM not preserved)
BODLEVEL2 (5)	2	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL1 (5)	1	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL0 (5)	0	Brown-out Detector trigger level	1 (unprogrammed)

Notes:
1. See “Alternate Functions of Port B” on page 65 for description of RSTDISBL and DWEN Fuses. After programming the RSTDISBL fuse, high-voltage serial programming must be used to change fuses and allow further programming.
2. DWEN must be unprogrammed when Lock Bit security is required. See “Program And Data Memory Lock Bits” on page 159.
3. The SPIEN Fuse is not accessible in SPI programming mode.
4. Programming this fues will disable the Watchdog Timer Interrupt. See “WDT Configuration as a Function of the Fuse Settings of WDTON” on page 43 for details.
5. See Table 20-7 on page 179 for BODLEVEL Fuse decoding.

Table 19-5. Fuse Low Byte

Fuse Low Byte	Bit No	Description	Default Value
CKDIV8 (1)	7	Divide clock by 8	0 (programmed)
CKOUT (2)	6	Clock Output Enable	1 (unprogrammed)
SUT1 (3)	5	Select start-up time	1 (unprogrammed)
SUT0 (3)	4	Select start-up time	0 (programmed)
CKSEL3 (4)	3	Select Clock source	0 (programmed)
CKSEL2 (4)	2	Select Clock source	0 (programmed)
CKSEL1 (4)	1	Select Clock source	1 (unprogrammed)
CKSEL0 (4)	0	Select Clock source	0 (programmed)

Notes:
1. See "System Clock Prescaler" on page 30 for details.
2. Allows system clock to be output on pin. See "Clock Output Buffer" on page 30 for details.
3. The default value results in maximum start-up time for the default clock source. See Table 6-5 on page 27 for details.
4. The default setting results in internal RC Oscillator @ 8.0 MHz. See Table 6-4 on page 27 for details.

Note that fuse bits are locked if Lock Bit 1 (LB1) is programmed. Fuse bits should be programmed before lock bits. The status of fuse bits is not affected by chip erase.

Fuse bits can also be read by device firmware. See section "Reading Lock, Fuse and Signature Data from Software" on page 155.

19.2.1 Latching of Fuses

Fuse values are latched when the device enters programming mode and changes to fuse values have no effect until the part leaves programming mode. This does not apply to the EESAVE Fuse which will take effect once it is programmed. Fuses are also latched on power-up.

19.3 Device Signature Imprint Table

The device signature imprint table is a dedicated memory area used for storing miscellaneous device information, such as the device signature and oscillator calibaration data. Most of this memory segment is reserved for internal use, as outlined in Table 19-6.

Table 19-6. Contents of Device Signature Imprint Table.

Address	High Byte
0x00	Signature byte 0 (1)
0x01	Calibration data for internal oscillator (2)
0x02	Signature byte 1 (1)
0x03	Reserved for internal use
0x04	Signature byte 2 (1)
0x05 ... 0x2A	Reserved for internal use

Notes:
1. See section "Signature Bytes" for more information.
2. See section "Calibration Byte" for more information.

19.3.1 Signature Bytes

All Atmel microcontrollers have a three-byte signature code which identifies the device. This code can be read in both serial and High-voltage Programming mode, also when the device is locked.

Signature bytes can also be read by the device firmware. See section “Reading Lock, Fuse and Signature Data from Software” on page 155.

The three signature bytes reside in a separate address space called the device signature imprint table. The signature data for ATtiny24/44/84 is given in Table 19-7.

Table 19-7. Device Signature Bytes

Part	Signature Byte 0	Signature Byte 1	Signature Byte 0
ATtiny24	0x1E	0x91	0x0B
ATtiny44	0x1E	0x92	0x07
ATtiny84	0x1E	0x93	0x0C

19.3.2 Calibration Byte

The device signature imprint table of ATtiny24/44/84 contains one byte of calibration data for the internal oscillator, as shown in Table 19-6 on page 161. During reset, this byte is automatically written into the OSCCAL register to ensure correct frequency of the calibrated oscillator.

Calibration bytes can also be read by the device firmware. See section “Reading Lock, Fuse and Signature Data from Software” on page 155.

19.4 Page Size

Table 19-8. 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	1K words (2K bytes)	16 words	PC[3:0]	64	PC[9:4]	9
ATtiny44	2K words (4K bytes)	32 words	PC[4:0]	64	PC[10:5]	10
ATtiny84	4K words (8K bytes)	32 words	PC[4:0]	128	PC[11:5]	11

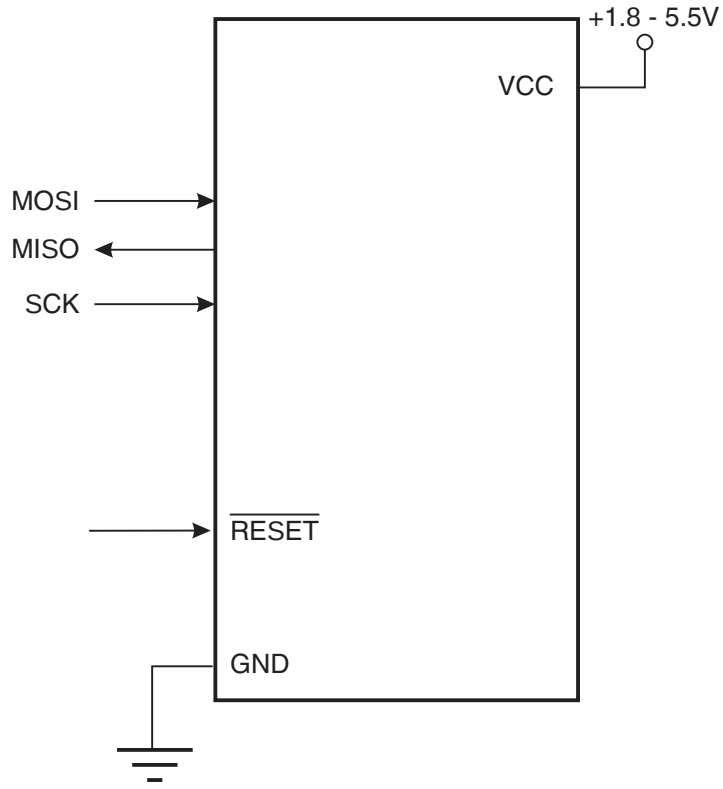
Table 19-9. No. of Words in a Page and No. of Pages in the EEPROM

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

19.5 Serial Programming

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). See Figure 19-1 below.

Figure 19-1. Serial Programming and Verify



Note: If clocked by internal oscillator there is no need to connect a clock source to the CLKI pin.

After $\overline{\text{RESET}}$ is set low, the Programming Enable instruction needs to be executed first before program/erase operations can be executed.

Table 19-10. 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

Note: In Table 19-10 above, the pin mapping for SPI programming is listed. Not all devices use the SPI pins dedicated for the internal SPI interface.

When programming the EEPROM, an auto-erase cycle is built into the self-timed programming operation (in the Serial mode ONLY) and there is no need to first execute the Chip Erase

instruction. The Chip Erase operation turns the content of every memory location in both the Program and EEPROM arrays into 0xFF.

Depending on CKSEL Fuses, a valid clock must be present. The minimum low and high periods for the serial clock (SCK) input are defined as follows:

- Low:> 2 CPU clock cycles for f_{ck} < 12 MHz, 3 CPU clock cycles for f_{ck} >= 12 MHz
- High:> 2 CPU clock cycles for f_{ck} < 12 MHz, 3 CPU clock cycles for f_{ck} >= 12 MHz

19.5.1 Serial Programming Algorithm

When writing serial data to the ATtiny24/44/84, data is clocked on the rising edge of SCK. When reading, data is clocked on the falling edge of SCK. See Figure 20-4 and Figure 20-5 for timing details.

To program and verify the ATtiny24/44/84 in the Serial Programming mode, the following sequence is recommended (see four byte instruction formats in Table 19-12):

1. Power-up sequence:
   Apply power between V_{CC} and GND while $\overline{\text{RESET}}$ and SCK are set to “0”. In some systems, the programmer can not guarantee that SCK is held low during power-up. In this case, $\overline{\text{RESET}}$ must be given a positive pulse after SCK has been set to '0'. The duration of the pulse must be at least t_{RST} (the minimum pulse width on $\overline{\text{RESET}}$ pin, see Table 20-4 on page 177) plus two CPU clock cycles.
2. Wait for at least 20 ms and enable serial programming by sending the Programming Enable serial instruction to pin MOSI.
3. The serial programming instructions will not work if the communication is out of synchronization. When in sync. the second byte (0x53), will echo back when issuing the third byte of the Programming Enable instruction. Whether the echo is correct or not, all four bytes of the instruction must be transmitted. If the 0x53 did not echo back, give $\overline{\text{RESET}}$ a positive pulse and issue a new Programming Enable command.
4. The Flash is programmed one page at a time. The memory page is loaded one byte at a time by supplying the 5 LSB of the address and data together with the Load Program memory Page instruction. To ensure correct loading of the page, the data low byte must be loaded before data high byte is applied for a given address. The Program memory Page is stored by loading the Write Program memory Page instruction with the 3 MSB of the address. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_FLASH} before issuing the next page. (See Table 19-11 on page 165.) Accessing the serial programming interface before the Flash write operation completes can result in incorrect programming.
5. **A:** The EEPROM array is programmed one byte at a time by supplying the address and data together with the appropriate Write instruction. An EEPROM memory location is first automatically erased before new data is written. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_EEPROM} before issuing the next byte. (See Table 19-11 on page 165.) In a chip erased device, no 0xFFs in the data file(s) need to be programmed.
   B: The EEPROM array is programmed one page at a time. The Memory page is loaded one byte at a time by supplying the 2 LSB of the address and data together with the Load EEPROM Memory Page instruction. The EEPROM Memory Page is stored by loading the Write EEPROM Memory Page Instruction with the 4 MSB of the address. When using EEPROM page access only byte locations loaded with the Load EEPROM Memory Page instruction is altered. The remaining locations remain unchanged. If polling (RDY/$\overline{\text{BSY}}$) is not used, the used must wait at least t_{WD_EEPROM} before issuing the

next page (See Table 19-11 on page 165). In a chip erased device, no 0xFF in the data file(s) need to be programmed.

6. Any memory location can be verified by using the Read instruction which returns the content at the selected address at serial output MISO.
7. At the end of the programming session, $\overline{\text{RESET}}$ can be set high to commence normal operation.
8. Power-off sequence (if needed):
   Set $\overline{\text{RESET}}$ to “1”.
   Turn V_{CC} power off.

Table 19-11. Minimum Wait Delay Before Writing the Next Flash or EEPROM Location

Symbol	Minimum Wait Delay
t_{WD_FLASH}	4.5 ms
t_{WD_EEPROM}	4.0 ms
t_{WD_ERASE}	4.0 ms
t_{WD_FUSE}	4.5 ms

19.5.2 Serial Programming Instruction set

The instruction set is described in Table 19-12 and Figure 19-2 on page 166.

Table 19-12. Serial Programming Instruction Set

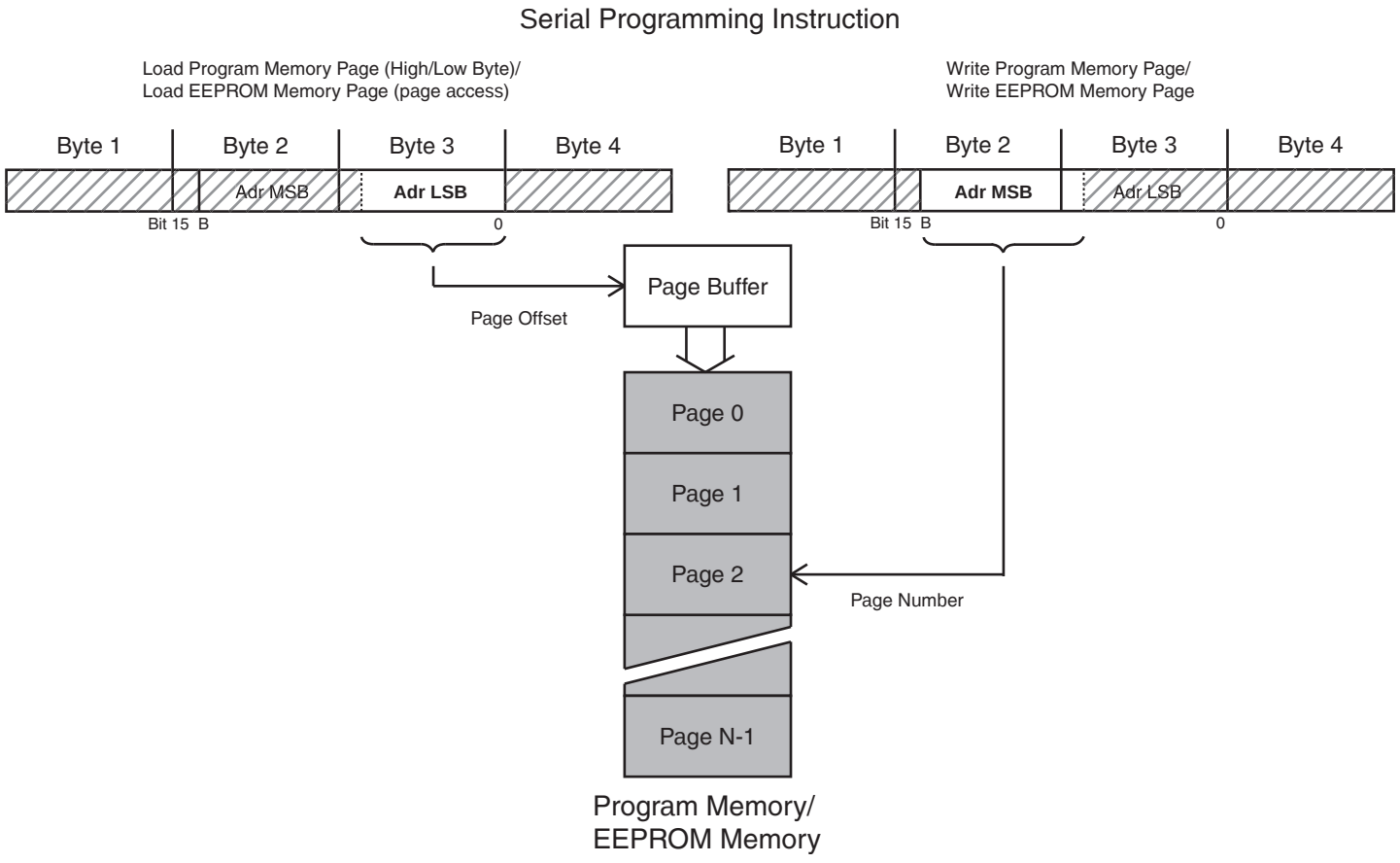
Instruction/Operation[1]	Instruction Format			
	Byte 1	Byte 2	Byte 3	Byte4
Programming Enable	$AC	$53	$00	$00
Chip Erase (Program Memory/EEPROM)	$AC	$80	$00	$00
Poll RDY/$\overline{\text{BSY}}$	$F0	$00	$00	data byte out
Load Instructions				
Load Extended Address byte	$4D	$00	Extended adr	$00
Load Program Memory Page, High byte	$48	adr MSB	adr LSB	high data byte in
Load Program Memory Page, Low byte	$40	adr MSB	adr LSB	low data byte in
Load EEPROM Memory Page (page access)	$C1	$00	adr LSB	data byte in
Read Instructions				
Read Program Memory, High byte	$28	adr MSB	adr LSB	high data byte out
Read Program Memory, Low byte	$20	adr MSB	adr LSB	low data byte out
Read EEPROM Memory	$A0	$00	adr LSB	data byte out
Read Lock bits	$58	$00	$00	data byte out
Read Signature Byte	$30	$00	adr LSB	data byte out
Read Fuse bits	$50	$00	$00	data byte out
Read Fuse High bits	$58	$08	$00	data byte out
Read Extended Fuse Bits	$50	$08	$00	data byte out
Read Calibration Byte	$38	$00	$00	data byte out

Table 19-12. Serial Programming Instruction Set (Continued)

Instruction/Operation[1]	Instruction Format			
	Byte 1	**Byte 2**	**Byte 3**	**Byte4**
Write Instructions[6]				
Write Program Memory Page	$4C	adr MSB	adr LSB	$00
Write EEPROM Memory	$C0	$00	adr LSB	data byte in
Write EEPROM Memory Page (page access)	$C2	$00	adr LSB	$00
Write Lock bits	$AC	$E0	$00	data byte in
Write Fuse bits	$AC	$A0	$00	data byte in
Write Fuse High bits	$AC	$A8	$00	data byte in
Write Extended Fuse Bits	$AC	$A4	$00	data byte in

Notes:
1. Not all instructions are applicable for all parts.
2. a = address
3. Bits are programmed ‘0’, unprogrammed ‘1’.
4. To ensure future compatibility, unused Fuses and Lock bits should be unprogrammed (‘1’) .
5. Refer to the correspondig section for Fuse and Lock bits, Calibration and Signature bytes and Page size.
6. Instructions accessing program memory use a word address. This address may be random within the page range.
7. See http://www.atmel.com/avr for Application Notes regarding programming and programmers.

Figure 19-2. Serial Programming Instruction example



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

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

After data is loaded to the page buffer, program the EEPROM page, see Figure 19-2 on page 166.

19.6 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 ATtiny24/44/84.

Figure 19-3. High-voltage Serial Programming

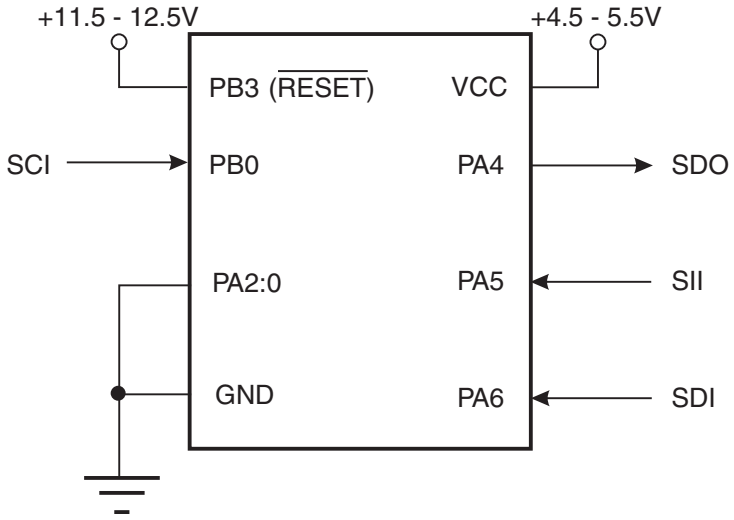


Table 19-13. Pin Name Mapping

Signal Name in High-voltage Serial Programming Mode	Pin Name	I/O	Function
SDI	PA6	I	Serial Data Input
SII	PA5	I	Serial Instruction Input
SDO	PA4	O	Serial Data Output
SCI	PB0	I	Serial Clock Input (min. 220ns period)

The minimum period for the Serial Clock Input (SCI) during High-voltage Serial Programming is 220 ns.

Table 19-14. Pin Values Used to Enter Programming Mode

Pin	Symbol	Value
PA0	Prog_enable[0]	0
PA1	Prog_enable[1]	0
PA2	Prog_enable[2]	0

19.7 High-Voltage Serial Programming Algorithm

To program and verify the ATtiny24/44/84 in the High-voltage Serial Programming mode, the following sequence is recommended (See instruction formats in Table 19-16 on page 171):

19.7.1 Enter High-voltage Serial Programming Mode

The following algorithm puts the device in High-voltage Serial Programming mode:

1. Set Prog_enable pins listed in Table 19-14 on page 167 to “000”, RESET pin and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND. Ensure that V_{CC} reaches at least 1.8V within the next 20 µs.
3. Wait 20 - 60 µs, and apply 11.5 - 12.5V to RESET.
4. Keep the Prog_enable pins unchanged for at least 10 µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
5. Release the Prog_enable[2] pin after t_{HVRST} has elapsed.
6. Wait at least 300 µs before giving any serial instructions on SDI/SII.
7. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

If the rise time of the V_{CC} is unable to fulfill the requirements listed above, the following alternative algorithm can be used:

1. Set Prog_enable pins listed in Table 19-14 on page 167 to “000”, RESET pin and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND.
3. Monitor V_{CC}, and as soon as V_{CC} reaches 0.9 - 1.1V, apply 11.5 - 12.5V to RESET.
4. Keep the Prog_enable pins unchanged for at least 10 µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
5. Release the Prog_enable[2] pin to avoid drive contention on the Prog_enable[2]/SDO pin.
6. Wait until V_{CC} actually reaches 4.5 - 5.5V before giving any serial instructions on SDI/SII.
7. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

Table 19-15. High-voltage Reset Characteristics

Supply Voltage	RESET Pin High-voltage Threshold	Minimum High-voltage Period for Latching Prog_enable
V_{CC}	V_{HVRST}	t_{HVRST}
4.5V	11.5V	100 ns
5.5V	11.5V	100 ns

19.7.2 Considerations for Efficient Programming

The loaded command and address are retained in the device during programming. For efficient programming, the following should be considered.

- The command needs only be loaded once when writing or reading multiple memory locations.
- Skip writing the data value 0xFF that is the contents of the entire EEPROM (unless the EESAVE Fuse is programmed) and Flash after a Chip Erase.
- Address High byte needs only be loaded before programming or reading a new 256 word window in Flash or 256 byte EEPROM. This consideration also applies to Signature bytes reading.

19.7.3 Chip Erase

The Chip Erase will erase the Flash and EEPROM[1] memories plus 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 re-programmed.

1. Load command “Chip Erase” (see Table 19-16 on page 171).
2. Wait after Instr. 3 until SDO goes high for the “Chip Erase” cycle to finish.
3. Load Command “No Operation”.

Note: 1. The EEPROM memory is preserved during Chip Erase if the EESAVE Fuse is programmed.

19.7.4 Programming the Flash

The Flash is organized in pages, see “Page Size” on page 162. When programming the Flash, the program data is 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 Command “Write Flash” (see Table 19-16 on page 171).
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 through 3 until the entire Flash is programmed or until all data has been programmed.
5. End Page Programming by Loading Command “No Operation”.

When writing or reading serial data to the ATtiny24/44/84, data is clocked on the rising edge of the serial clock, see Figure 20-6 on page 184, Figure 19-3 on page 167 and Table 20-13 on page 184 for details.

Figure 19-4. Addressing the Flash which is Organized in Pages

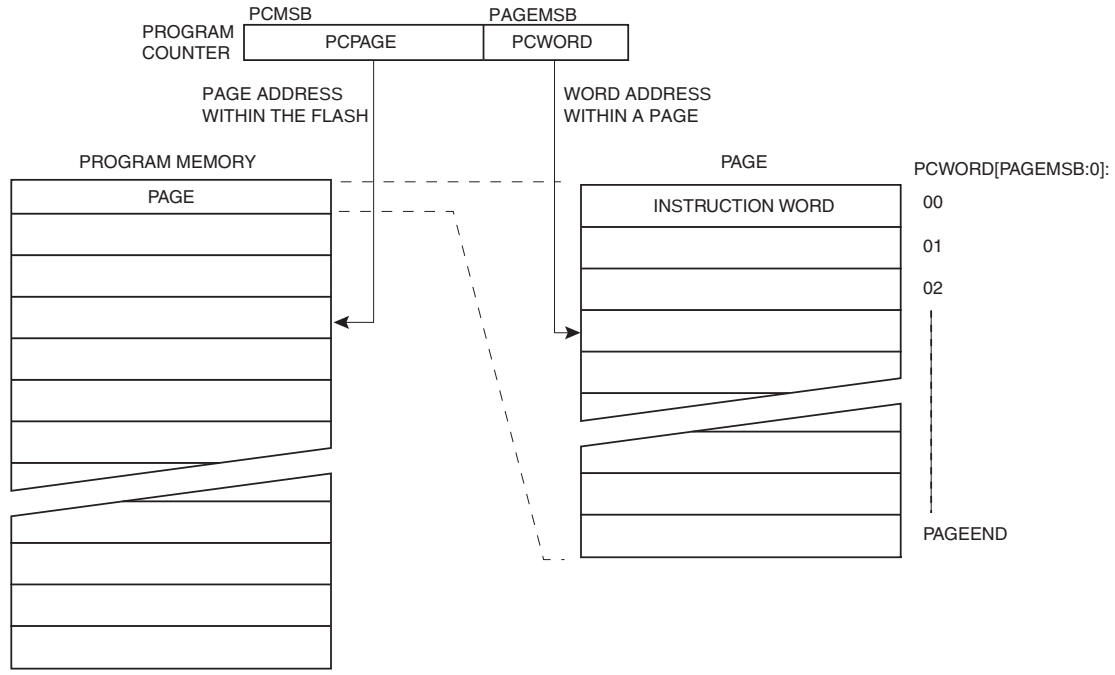
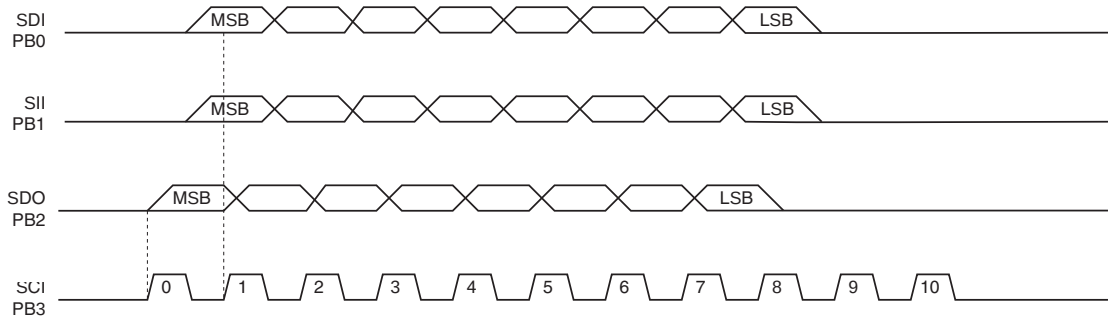


Figure 19-5. High-voltage Serial Programming Waveforms



19.7.5 Programming the EEPROM

The EEPROM is organized in pages, see Table 20-12 on page 183. When programming the EEPROM, the data is latched into a page buffer. This allows one page of data to be programmed simultaneously. The programming algorithm for the EEPROM Data memory is as follows (refer to Table 19-16 on page 171):

1. Load Command “Write EEPROM”.
2. Load EEPROM Page Buffer.
3. Program EEPROM Page. Wait after Instr. 2 until SDO goes high for the “Page Programming” cycle to finish.
4. Repeat 2 through 3 until the entire EEPROM is programmed or until all data has been programmed.
5. End Page Programming by Loading Command “No Operation”.

19.7.6 Reading the Flash

The algorithm for reading the Flash memory is as follows (refer to Table 19-16 on page 171):

1. Load Command "Read Flash".
2. Read Flash Low and High Bytes. The contents at the selected address are available at serial output SDO.

19.7.7 Reading the EEPROM

The algorithm for reading the EEPROM memory is as follows (refer to Table 19-16 on page 171):

1. Load Command “Read EEPROM”.
2. Read EEPROM Byte. The contents at the selected address are available at serial output SDO.

19.7.8 Programming and Reading the Fuse and Lock Bits

The algorithms for programming and reading the Fuse Low/High bits and Lock bits are shown in Table 19-16 on page 171.

19.7.9 Reading the Signature Bytes and Calibration Byte

The algorithms for reading the Signature bytes and Calibration byte are shown in Table 19-16 on page 171.

19.7.10 Power-off sequence

Set SCI to “0”. Set RESET to “1”. Turn V_{CC} power off.

Table 19-16. High-voltage Serial Programming Instruction Set for ATtiny24/44/84

Instruction		Instruction Format				Operation Remarks
		Instr.1/5	Instr.2/6	Instr.3/7	Instr.4	
Chip Erase	SDI SII SDO	0_1000_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx		Wait after Instr.3 until SDO goes high for the Chip Erase cycle to finish.
Load “Write Flash” Command	SDI SII SDO	0_0001_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx				Enter Flash Programming code.
Load Flash Page Buffer	SDI SII SDO	0_ **bbbb_bbbb** _00 0_0000_1100_00 x_xxxx_xxxx_xx	0_**eeee_eeee**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1101_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	Repeat after Instr. 1 - 7until the entire page buffer is filled or until all data within the page is filled. See Note 1.
	SDI SII SDO	0_**dddd_dddd**_00 0_0011_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1101_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 x_xxxx_xxxx_xx		Instr 5-7.
Load Flash High Address and Program Page	SDI SII SDO	0_0000_000**a**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx		Wait after Instr 3 until SDO goes high. Repeat Instr. 2 - 3 for each loaded Flash Page until the entire Flash or all data is programmed. Repeat Instr. 1 for a new 256 byte page. See Note 1.

Table 19-16. High-voltage Serial Programming Instruction Set for ATtiny24/44/84 (Continued)

Instruction		Instruction Format				Operation Remarks
		Instr.1/5	Instr.2/6	Instr.3/7	Instr.4	
Load “Read Flash” Command	SDI SII SDO	0_0000_0010_00 0_0100_1100_00 x_xxxx_xxxx_xx				Enter Flash Read mode.
Read Flash Low and High Bytes	SDI SII SDO	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_0000_000**a**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **q_qqqq_qqq**x_xx	Repeat Instr. 1, 3 - 6 for each new address. Repeat Instr. 2 for a new 256 byte page.
	SDI SII SDO	0_0000_0000_00 0_0111_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 **p_pppp_ppp**x_xx			Instr 5 - 6.
Load “Write EEPROM” Command	SDI SII SDO	0_0001_0001_00 0_0100_1100_00 x_xxxx_xxxx_xx				Enter EEPROM Programming mode.
Load EEPROM Page Buffer	SDI SII SDO	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_**aaaa_aaaa**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_**eeee_eeee**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1101_00 x_xxxx_xxxx_xx	Repeat Instr. 1 - 5 until the entire page buffer is filled or until all data within the page is filled. See Note 2.
	SDI SII SDO	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx				
Program EEPROM Page	SDI SII SDO	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx			Wait after Instr. 2 until SDO goes high. Repeat Instr. 1 - 2 for each loaded EEPROM page until the entire EEPROM or all data is programmed.
Write EEPROM Byte	SDI SII SDO	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_**aaaa_aaaa**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_**eeee_eeee**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1101_00 x_xxxx_xxxx_xx	Repeat Instr. 1 - 6 for each new address. Wait after Instr. 6 until SDO goes high. See Note 3.
	SDI SII SDO	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx			Instr. 5-6
Load “Read EEPROM” Command	SDI SII SDO	0_0000_0011_00 0_0100_1100_00 x_xxxx_xxxx_xx				Enter EEPROM Read mode.
Read EEPROM Byte	SDI SII SDO	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_**aaaa_aaaa**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **q_qqqq_qqq**0_00	Repeat Instr. 1, 3 - 4 for each new address. Repeat Instr. 2 for a new 256 byte page.
Write Fuse Low Bits	SDI SII SDO	0_0100_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_**A987_6543**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	Wait after Instr. 4 until SDO goes high. Write **A - 3** = “0” to program the Fuse bit.
Write Fuse High Bits	SDI SII SDO	0_0100_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_**IHGF_EDCB**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 x_xxxx_xxxx_xx	Wait after Instr. 4 until SDO goes high. Write **F - B** = “0” to program the Fuse bit.
Write Fuse Extended Bits	SDI SII SDO	0_0100_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_000**J**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0110_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1110_00 x_xxxx_xxxx_xx	Wait after Instr. 4 until SDO goes high. Write **J** = “0” to program the Fuse bit.

Table 19-16. High-voltage Serial Programming Instruction Set for ATtiny24/44/84 (Continued)

Instruction		Instruction Format				Operation Remarks
		Instr.1/5	Instr.2/6	Instr.3/7	Instr.4	
Write Lock Bits	SDI SII SDO	0_0010_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_00**21**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	Wait after Instr. 4 until SDO goes high. Write **2 - 1** = “0” to program the Lock Bit.
Read Fuse Low Bits	SDI SII SDO	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **A_9876_543**x_xx		Reading **A - 3** = “0” means the Fuse bit is programmed.
Read Fuse High Bits	SDI SII SDO	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1010_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 **I_HGFE_DCB**x_xx		Reading **F - B** = “0” means the Fuse bit is programmed.
Read Fuse Extended Bits	SDI SII SDO	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1010_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1110_00 x_xxxx_xxJx_xx		Reading **J** = “0” means the Fuse bit is programmed.
Read Lock Bits	SDI SII SDO	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_x**21**x_xx		Reading **2, 1** = “0” means the Lock bit is programmed.
Read Signature Bytes	SDI SII SDO	0_0000_1000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_00**bb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **q_qqqq_qqq**x_xx	Repeats Instr 2 4 for each signature byte address.
Read Calibration Byte	SDI SII SDO	0_0000_1000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 **p_pppp_ppp**x_xx	
Load “No Operation” Command	SDI SII SDO	0_0000_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx				

Note: **a** = address high bits, **b** = address low bits, **d** = data in high bits, **e** = data in low bits, **p** = data out high bits, **q** = data out low bits, x = don’t care, ***1*** = Lock Bit1, ***2*** = Lock Bit2, **3** = CKSEL0 Fuse, **4** = CKSEL1 Fuse, **5** = CKSEL2 Fuse, **6** = CKSEL3 Fuse, **7** = SUT0 Fuse, **8** = SUT1 Fuse, **9** = CKDIV8 Fuse, **A** = CKOUT Fuse, **B** = BODLEVEL0 Fuse, **C** = BODLEVEL1 Fuse, **D**= BODLEVEL2 Fuse, **E** = EESAVE Fuse, **F** = WDTON Fuse, **G** = SPIEN Fuse, **H** = DWEN Fuse, **I** = RSTDISBL Fuse

Notes:
1. For page sizes less than 256 words, parts of the address (bbbb_bbbb) will be parts of the page address.
2. For page sizes less than 256 bytes, parts of the address (bbbb_bbbb) will be parts of the page address.
3. The EEPROM is written page-wise. But only the bytes that are loaded into the page are actually written to the EEPROM. Page-wise EEPROM access is more efficient when multiple bytes are to be written to the same page. Note that auto-erase of EEPROM is not available in High-voltage Serial Programming, only in SPI Programming.

20. Electrical Characteristics

20.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

20.2 DC Characteristics

Table 20-1. DC Characteristics. T_A = -40°C to +85°C

Symbol	Parameter	Condition	Min (1)	Typ (1)	Max (1)	Units
V_{IL}	Input Low Voltage	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5		$0.2V_{CC}$ $0.3V_{CC}$	V
V_{IH}	Input High-voltage Except $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$(2) $0.6V_{CC}$(2)		V_{CC} +0.5(3)	V
V_{IH1}	Input High-voltage $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V to 5.5V	$0.9V_{CC}$(2)		V_{CC} +0.5(3)	V
V_{OL}	Output Low Voltage(4) Except $\overline{\text{RESET}}$ pin(6)	I_{OL} = 10 mA, V_{CC} = 5V I_{OL} = 5 mA, V_{CC} = 3V			0.6 0.5	V V
V_{OH}	Output High-voltage(5) Except $\overline{\text{RESET}}$ pin(6)	I_{OH} = -10 mA, V_{CC} = 5V I_{OH} = -5 mA, V_{CC} = 3V	4.3 2.5			V V
I_{LIL}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin low (absolute value)		<0.05	1	µA
I_{LIH}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin high (absolute value)		<0.05	1	µA
R_{RST}	Reset Pull-up Resistor	Vcc = 5.5V, input low	30		60	kΩ
R_{PU}	I/O Pin Pull-up Resistor	Vcc = 5.5V, input low	20		50	kΩ

Table 20-1. DC Characteristics. T_A = -40°C to +85°C (Continued)

Symbol	Parameter	Condition	Min (1)	Typ (1)	Max (1)	Units
I_{CC}	Power Supply Current	Active 1MHz, V_{CC} = 2V		0.33	0.8	mA
		Active 4MHz, V_{CC} = 3V		1.6	2.5	mA
		Active 8MHz, V_{CC} = 5V		5	9	mA
		Idle 1MHz, V_{CC} = 2V		0.11	0.4	mA
		Idle 4MHz, V_{CC} = 3V		0.4	1.0	mA
		Idle 8MHz, V_{CC} = 5V		1.5	3.5	mA
	Power-down mode	WDT enabled, V_{CC} = 3V		4.5	10	µA
		WDT disabled, V_{CC} = 3V		0.15	2	µA

Notes:
1. All DC Characteristics contained in this data sheet are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are preliminary values representing design targets, and will be updated after characterization of actual silicon.
2. “Min” means the lowest value where the pin is guaranteed to be read as high.
3. “Max” means the highest value where the pin is guaranteed to be read as low.
4. Although each I/O port can sink more than the test conditions (10 mA at V_{CC} = 5V, 5 mA at V_{CC} = 3V) under steady state conditions (non-transient), the sum of all I_{OL} (for all ports) should not exceed 60 mA. If I_{OL} exceeds the test conditions, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
5. Although each I/O port can source more than the test conditions (10 mA at V_{CC} = 5V, 5 mA at V_{CC} = 3V) under steady state conditions (non-transient), the sum of all I_{OH} (for all ports) should not exceed 60 mA. If I_{OH} exceeds the test condition, V_{OH} may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.
6. The $\overline{\text{RESET}}$ pin must tolerate high voltages when entering and operating in programming modes and, as a consequence, has a weak drive strength as compared to regular I/O pins. See Figure 21-24, Figure 21-25, Figure 21-26, and Figure 21-27 (starting on page 198).

20.3 Speed Grades

The maximum operating frequency of the device depends on $V_{CC.}$ As shown in Figure 20-1 and Figure 20-2, the maximum frequency vs. V_{CC} relationship is linear between 1.8V < V_{CC} < 2.7V and between 2.7V < V_{CC} < 4.5V.

Figure 20-1. Maximum Frequency vs. V_{CC} (ATtiny24V/44V/84V)

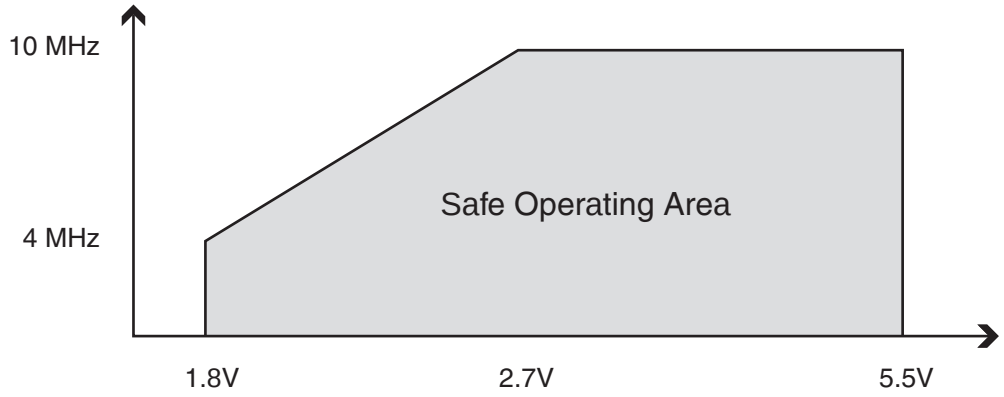
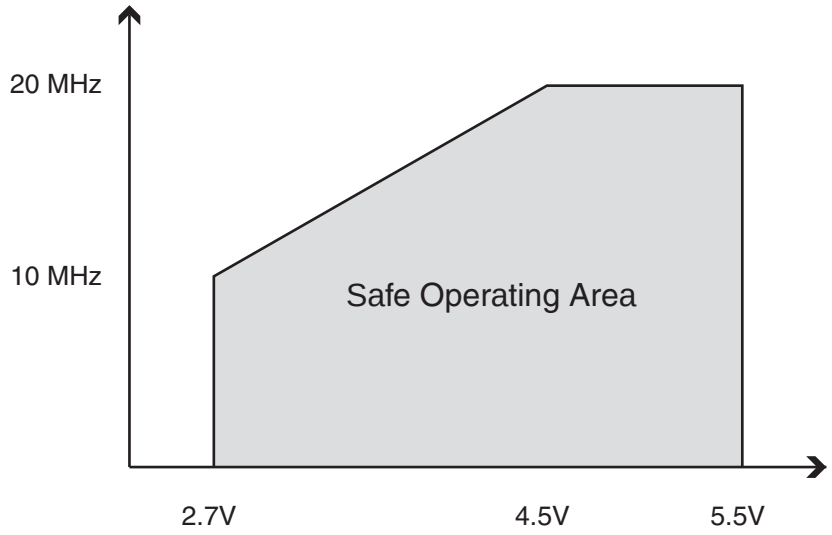


Figure 20-2. Maximum Frequency vs. V_{CC} (ATtiny24/44/84)



20.4 Clock Characteristics

20.4.1 Accuracy of Calibrated Internal RC Oscillator

It is possible to manually calibrate the internal oscillator to be more accurate than default factory calibration. Note that the oscillator frequency depends on temperature and voltage. Voltage and temperature characteristics can be found in Figure 21-40 on page 206 and Figure 21-41 on page 206.

Table 20-2. Calibration Accuracy of Internal RC Oscillator

Calibration Method	Target Frequency	V_{CC}	Temperature	Accuracy at given voltage & temperature[1]
Factory Calibration	8.0 MHz	3V	25°C	±10%
User Calibration	Fixed frequency within: 7.3 – 8.1 MHz	Fixed voltage within: 1.8V – 5.5V[2] 2.7V – 5.5V[3]	Fixed temperature within: -40°C – 85°C	±1%

Notes:
1. Accuracy of oscillator frequency at calibration point (fixed temperature and fixed voltage).
2. Voltage range for ATtiny24V/44V/84V.
3. Voltage range for ATtiny24/44/84.

20.4.2 External Clock Drive

Figure 20-3. External Clock Drive Waveform

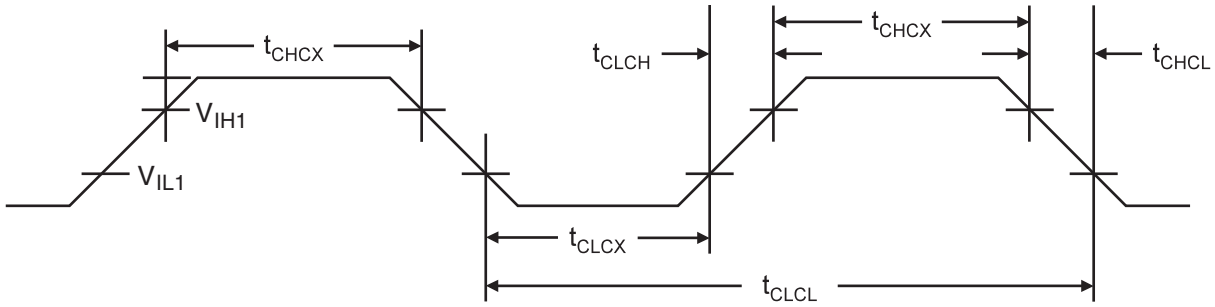


Table 20-3. External Clock Drive Characteristics

Symbol	Parameter	V_{CC} = 1.8 - 5.5V		V_{CC} = 2.7 - 5.5V		V_{CC} = 4.5 - 5.5V		Units
		Min.	Max.	Min.	Max.	Min.	Max.	
$1/t_{CLCL}$	Clock Frequency	0	4	0	10	0	20	MHz
t_{CLCL}	Clock Period	250		100		50		ns
t_{CHCX}	High Time	100		40		20		ns
t_{CLCX}	Low Time	100		40		20		ns
t_{CLCH}	Rise Time		2.0		1.6		0.5	µs
t_{CHCL}	Fall Time		2.0		1.6		0.5	µs
Δt_{CLCL}	Change in period from one clock cycle to the next		2		2		2	%

20.5 System and Reset Characteristics

Table 20-4. Reset, Brown-out, and Internal Voltage Characteristics

Symbol	Parameter	Condition	Min[1]	Typ[1]	Max[1]	Units
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		0.2 V_{CC}		0.9V_{CC}	V
t_{RST}	Minimum pulse width on $\overline{\text{RESET}}$ Pin	V_{CC} = 1.8V V_{CC} = 3V V_{CC} = 5V		2000 700 400		ns
V_{HYST}	Brown-out Detector Hysteresis			50		mV
t_{BOD}	Min Pulse Width on Brown-out Reset			2		µs
V_{BG}	Internal bandgap reference voltage	V_{CC} = 5V T_A = 25°C	1.0	1.1	1.2	V
t_{BG}	Internal bandgap reference start-up time	V_{CC} = 5V T_A = 25°C		40	70	µs
I_{BG}	Internal bandgap reference current consumption	V_{CC} = 5V T_A = 25°C		15		µA

Note: 1. Values are guidelines, only

Two versions of power-on reset have been implemented, as follows.

20.5.1 Standard Power-On Reset

This implementation of power-on reset existed in early versions of ATtiny24/44/84. The table below describes the characteristics of this power-on reset and it is valid for the following devices, only:

- ATtiny24, revision D, and older
- ATtiny44, revision C, and older
- ATtiny84, revision A

Note: Revisions are marked on the package (packages 14P3 and 14S1: bottom, package 20M1: top)

Table 20-5. Characteristics of Standard Power-On Reset. T_A = -40 - 85°C

Symbol	Parameter	Min(1)	Typ(1)	Max(1)	Units
V_{POR}	Release threshold of power-on reset (2)	0.7	1.0	1.4	V
V_{POA}	Activation threshold of power-on reset (3)	0.05	0.9	1.3	V
SR_{ON}	Power-on slope rate	0.01		4.5	V/ms

Note:
1. Values are guidelines, only
2. Threshold where device is released from reset when voltage is rising
3. The power-on reset will not work unless the supply voltage has been below V_{POA}

20.5.2 Enhanced Power-On Reset

This implementation of power-on reset exists in newer versions of ATtiny24/44/84. The table below describes the characteristics of this power-on reset and it is valid for the following devices, only:

- ATtiny24, revision E, and newer
- ATtiny44, revision D, and newer
- ATtiny84, revision B, and newer

Table 20-6. Characteristics of Enhanced Power-On Reset. T_A = -40 - 85°C

Symbol	Parameter	Min(1)	Typ(1)	Max(1)	Units
V_{POR}	Release threshold of power-on reset (2)	1.1	1.4	1.6	V
V_{POA}	Activation threshold of power-on reset (3)	0.6	1.3	1.6	V
SR_{ON}	Power-On Slope Rate	0.01			V/ms

Note:
1. Values are guidelines, only
2. Threshold where device is released from reset when voltage is rising
3. The Power-on Reset will not work unless the supply voltage has been below V_{POA}

20.5.3 Brown-Out Detection

Table 20-7. V_{BOT} vs. BODLEVEL Fuse Coding

BODLEVEL [2..0] Fuses	Min(1)	Typ(1)	Max(1)	Units
111	BOD Disabled			
110	1.7	1.8	2.0	V
101	2.5	2.7	2.9	
100	4.1	4.3	4.5	
0XX	Reserved			

Note: 1. V_{BOT} may be below nominal minimum operating voltage for some devices. For devices where this is the case, the device is tested down to $V_{CC} = V_{BOT}$ during the production test. This guarantees that a Brown-out Reset will occur before V_{CC} drops to a voltage where correct operation of the microcontroller is no longer guaranteed.

20.6 Analog Comparator Characteristics

Table 20-8. Analog Comparator Characteristics, T_A = -40°C - 85°C

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{AIO}	Input Offset Voltage	V_{CC} = 5V, V_{IN} = V_{CC} / 2		< 10	40	mV
I_{LAC}	Input Leakage Current	V_{CC} = 5V, V_{IN} = V_{CC} / 2	-50		50	nA
t_{APD}	Analog Propagation Delay (from saturation to slight overdrive)	V_{CC} = 2.7V		750		ns
		V_{CC} = 4.0V		500		
	Analog Propagation Delay (large step change)	V_{CC} = 2.7V		100		
		V_{CC} = 4.0V		75		
t_{DPD}	Digital Propagation Delay	V_{CC} = 1.8V - 5.5		1	2	CLK

Note: All parameters are based on simulation results and they are not tested in production

20.7 ADC Characteristics – Preliminary Data

Table 20-9. ADC Characteristics, Single Ended Channels. T = -40°C - 85°C

Symbol	Parameter	Condition	Min[1]	Typ[1]	Max[1]	Units
	Resolution				10	Bits
	Absolute accuracy (Including INL, DNL, and Quantization, Gain and Offset Errors)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2.0	TBD	LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz		2.5	TBD	LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz Noise Reduction Mode		1.5	TBD	LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz Noise Reduction Mode		2.0	TBD	LSB
	Integral Non-Linearity (INL) (Accuracy after Offset and Gain Calibration)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		1.0	TBD	LSB
	Differential Non-linearity (DNL)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		0.5	TBD	LSB
	Gain Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2.0	TBD	LSB
	Offset Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		1.5	TBD	LSB
	Conversion Time	Free Running Conversion	14		280	µs
	Clock Frequency		50		1000	kHz
V_{IN}	Input Voltage		GND		V_{REF}	V
	Input Bandwidth			38.4		kHz
A_{REF}	External Voltage Reference		2.0		V_{CC}	V
V_{INT}	Internal Voltage Reference		1.0	1.1	1.2	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance			100		MΩ
	ADC Conversion Output		0		1023	LSB

Note: 1. Values are preliminary.

Table 20-10. ADC Characteristics, Differential Channels (Unipolar Mode), T_A = -40°C to 85°C

Symbol	Parameter	Condition	Min[1]	Typ[1]	Max[1]	Units
	Resolution	Gain = 1x			10	Bits
		Gain = 20x			10	Bits
	Absolute accuracy (Including INL, DNL, and Quantization, Gain and Offset Errors)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		10.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		20.0	TBD	LSB
	Integral Non-Linearity (INL) (Accuracy after Offset and Gain Calibration)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		4.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		10.0	TBD	LSB
	Gain Error	Gain = 1x		10.0	TBD	LSB
		Gain = 20x		15.0	TBD	LSB
	Offset Error	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		3.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		4.0	TBD	LSB
	Conversion Time	Free Running Conversion	70		280	µs
	Clock Frequency		50		200	kHz
V_{IN}	Input Voltage		GND		V_{CC}	V
V_{DIFF}	Input Differential Voltage				V_{REF}/Gain	V
	Input Bandwidth			4		kHz
A_{REF}	External Reference Voltage		2.0		V_{CC} - 1.0	V
V_{INT}	Internal Voltage Reference		1.0	1.1	1.2	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance			100		MΩ
	ADC Conversion Output		0		1023	LSB

Note: 1. Values are preliminary.

Table 20-11. ADC Characteristics, Differential Channels (Bipolar Mode), T_A = -40°C to 85°C

Symbol	Parameter	Condition	Min[1]	Typ[1]	Max[1]	Units
	Resolution	Gain = 1x			10	Bits
		Gain = 20x			10	Bits
	Absolute accuracy (Including INL, DNL, and Quantization, Gain and Offset Errors)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		8.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		8.0	TBD	LSB
	Integral Non-Linearity (INL) (Accuracy after Offset and Gain Calibration)	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		4.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		5.0	TBD	LSB
	Gain Error	Gain = 1x		4.0	TBD	LSB
		Gain = 20x		5.0	TBD	LSB
	Offset Error	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		3.0	TBD	LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC clock = 50 - 200 kHz		4.0	TBD	LSB
	Conversion Time	Free Running Conversion	70		280	µs
	Clock Frequency		50		200	kHz
V_{IN}	Input Voltage		GND		V_{CC}	V
V_{DIFF}	Input Differential Voltage				V_{REF}/Gain	V
	Input Bandwidth			4		kHz
A_{REF}	External Reference Voltage		2.0		V_{CC} - 1.0	V
V_{INT}	Internal Voltage Reference		1.0	1.1	1.2	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance			100		MΩ
	ADC Conversion Output		-512		511	LSB

Note: 1. Values are preliminary.

20.8 Serial Programming Characteristics

Figure 20-4. Serial Programming Timing

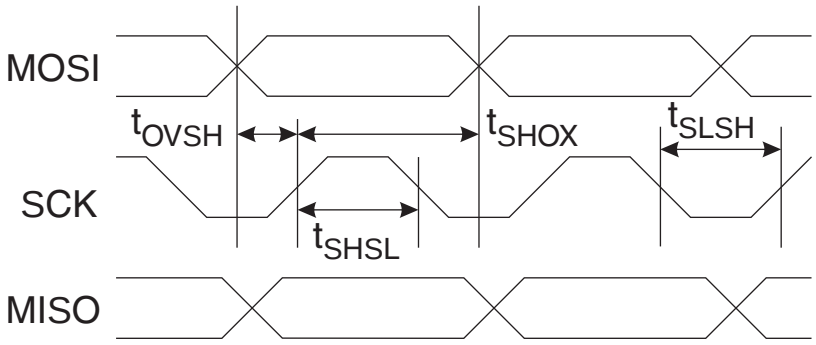


Figure 20-5. Serial Programming Waveform

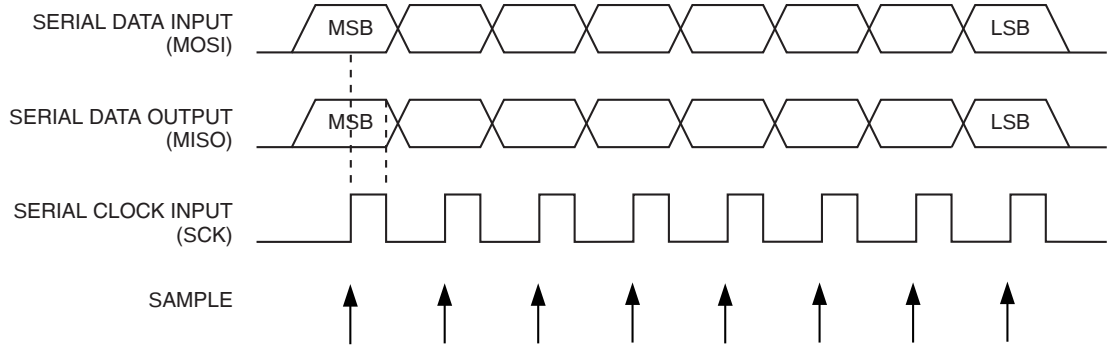


Table 20-12. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 1.8 - 5.5V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
$1/t_{CLCL}$	Oscillator Frequency (ATtiny24/44/84V)	0		4	MHz
t_{CLCL}	Oscillator Period (ATtiny24/44/84V)	250			ns
$1/t_{CLCL}$	Oscillator Freq. (ATtiny24/44/84, V_{CC} = 4.5V - 5.5V)	0		20	MHz
t_{CLCL}	Oscillator Period (ATtiny24/44/84, V_{CC} = 4.5V - 5.5V)	50			ns
t_{SHSL}	SCK Pulse Width High	2 t_{CLCL} (1)			ns
t_{SLSH}	SCK Pulse Width Low	2 t_{CLCL} (1)			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2 t_{CLCL}			ns

Note: 1. 2 t_{CLCL} for f_{ck} < 12 MHz, 3 t_{CLCL} for f_{ck} >= 12 MHz

20.9 High-Voltage Serial Programming Characteristics

Figure 20-6. High-voltage Serial Programming Timing

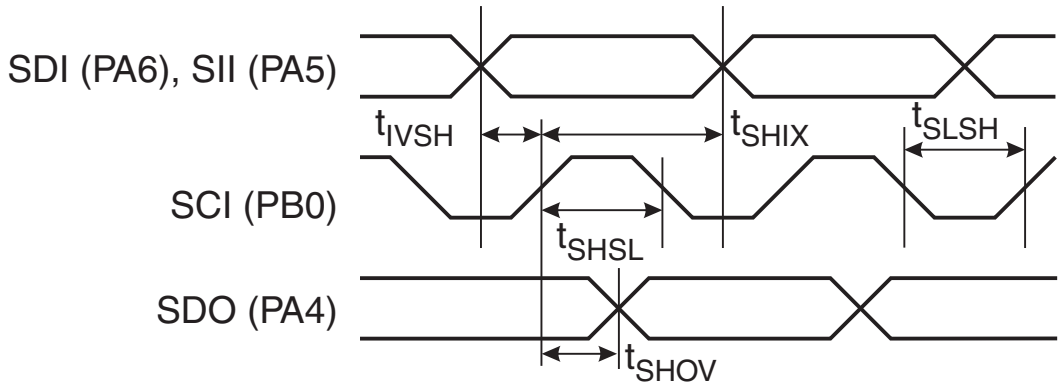


Table 20-13. High-voltage Serial Programming Characteristics
T_A = 25°C, V_{CC} = 5V (Unless otherwise noted)

Symbol	Parameter	Min	Typ	Max	Units
t_{SHSL}	SCI (PB0) Pulse Width High	125			ns
t_{SLSH}	SCI (PB0) Pulse Width Low	125			ns
t_{IVSH}	SDI (PA6), SII (PB1) Valid to SCI (PB0) High	50			ns
t_{SHIX}	SDI (PA6), SII (PB1) Hold after SCI (PB0) High	50			ns
t_{SHOV}	SCI (PB0) High to SDO (PA4) Valid		16		ns
t_{WLWH_PFB}	Wait after Instr. 3 for Write Fuse Bits		2.5		ms

21. Typical Characteristics

The data contained in this section is largely based on simulations and characterization of similar devices in the same process and design methods. Thus, the data should be treated as indications of how the part will behave.

The following charts show typical behavior. These figures are not tested during manufacturing. During characterisation devices are operated at frequencies higher than test limits but they are not guaranteed to function properly at frequencies higher than the ordering code indicates.

All current consumption measurements are performed with all I/O pins configured as inputs and with internal pull-ups enabled. Current consumption is a function of several factors such as operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature. The dominating factors are operating voltage and frequency.

A sine wave generator with rail-to-rail output is used as clock source but current consumption in Power-Down mode is independent of clock selection. The difference between current consumption in Power-Down mode with Watchdog Timer enabled and Power-Down mode with Watchdog Timer disabled represents the differential current drawn by the Watchdog Timer.

The current drawn from pins with a capacitive load may be estimated (for one pin) as follows:

$$I_{CP} \approx V_{CC} \times C_L \times f_{SW}$$

where V_{CC} = operating voltage, C_L = load capacitance and f_{SW} = average switching frequency of I/O pin.

21.1 Supply Current of I/O Modules

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in Active and Idle mode. The enabling or disabling of the I/O modules is controlled by the Power Reduction Register. See “Power Reduction Register” on page 35 for details.

Table 21-1. Additional Current Consumption for the different I/O modules (absolute values)

	Typical numbers		
PRR bit	**V_{CC} = 2V, f = 1MHz**	**V_{CC} = 3V, f = 4MHz**	**V_{CC} = 5V, f = 8MHz**
PRTIM1	5.1 uA	31.0 uA	118.2 uA
PRTIM0	6.6 uA	40.0 uA	153.0 uA
PRUSI	3.7 uA	23.1 uA	92.2 uA
PRADC	29.6 uA	88.3 uA	333.3 uA

Table 21-2 below can be used for calculating typical current consumption for other supply voltages and frequencies than those mentioned in the Table 21-1 above.

Table 21-2. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Current consumption additional to active mode with external clock (see Figure 21-1 and Figure 21-2)	Current consumption additional to idle mode with external clock (see Figure 21-6 and Figure 21-7)
PRTIM1	1.8 %	8.0 %
PRTIM0	2.3 %	10.4 %
PRUSI	1.4 %	6.1 %
PRADC	6.7 %	28.8 %

21.1.1 Example

Calculate the expected current consumption in idle mode with USI, TIMER0, and ADC enabled at V_{CC} = 2.0V and f = 1MHz. From Table 21-2 on page 186, third column, we see that we need to add 6.1% for the USI, 10.4% for TIMER0, and 28.8% for the ADC. Reading from Figure 21-6 on page 189, we find that current consumption in idle mode at 2V and 1MHz is about 0.04mA. The total current consumption in idle mode with USI, TIMER0, and ADC enabled is therefore:

$$I_{CCTOT} \approx 0.04mA \times (1 + 0.061 + 0.104 + 0.288) \approx 0.06mA$$

21.2 Active Supply Current

Figure 21-1. Active Supply Current vs. Low Frequency (0.1 - 1.0 MHz)

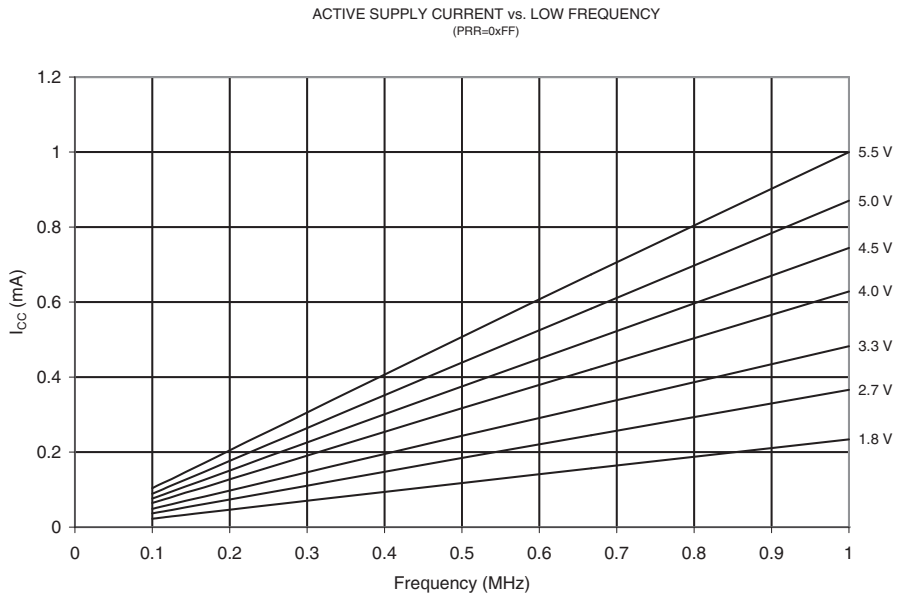


Figure 21-2. Active Supply Current vs. frequency (1 - 20 MHz)

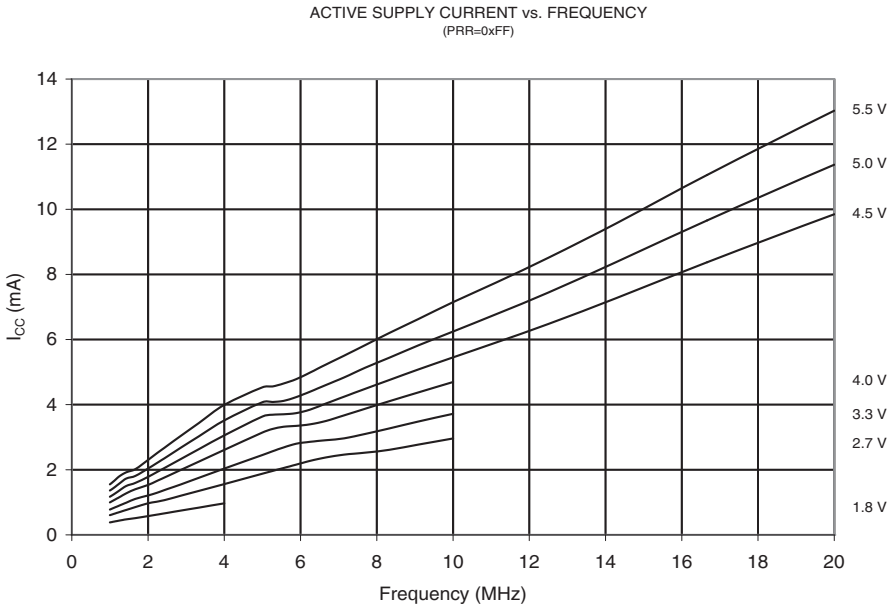


Figure 21-3. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz)

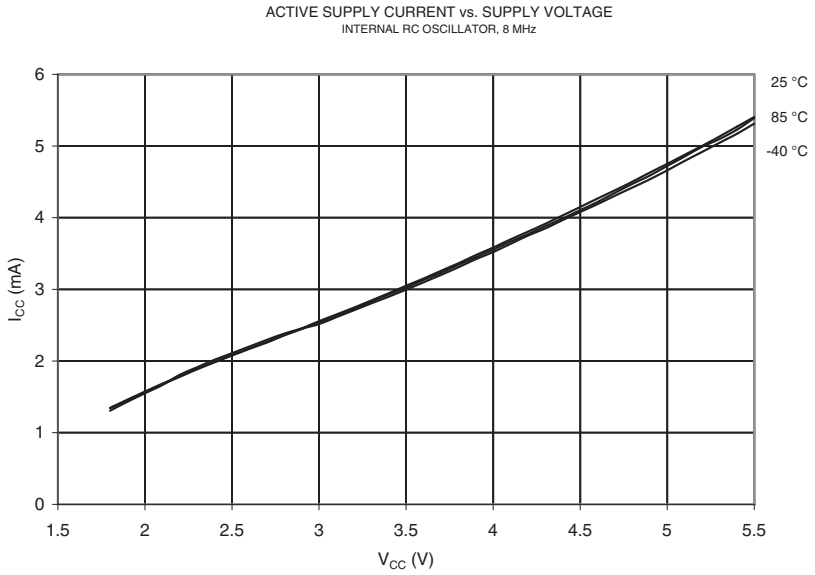


Figure 21-4. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz)

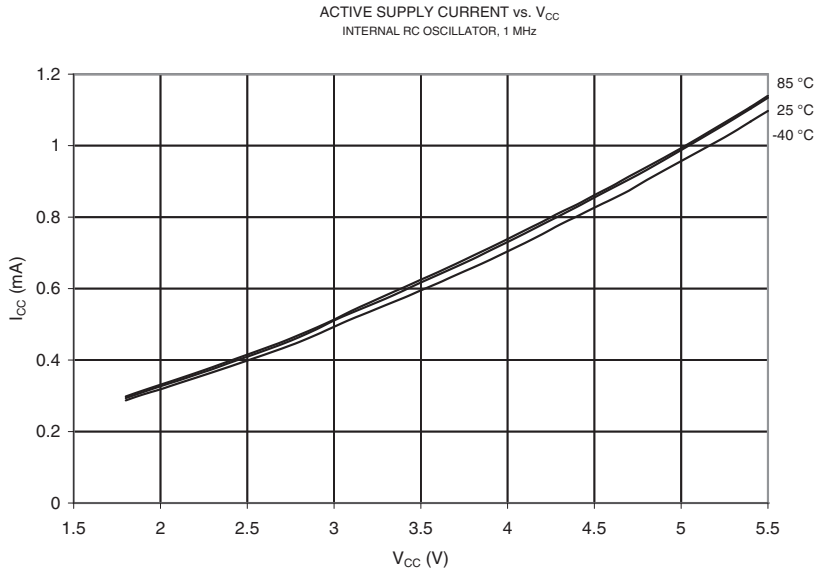
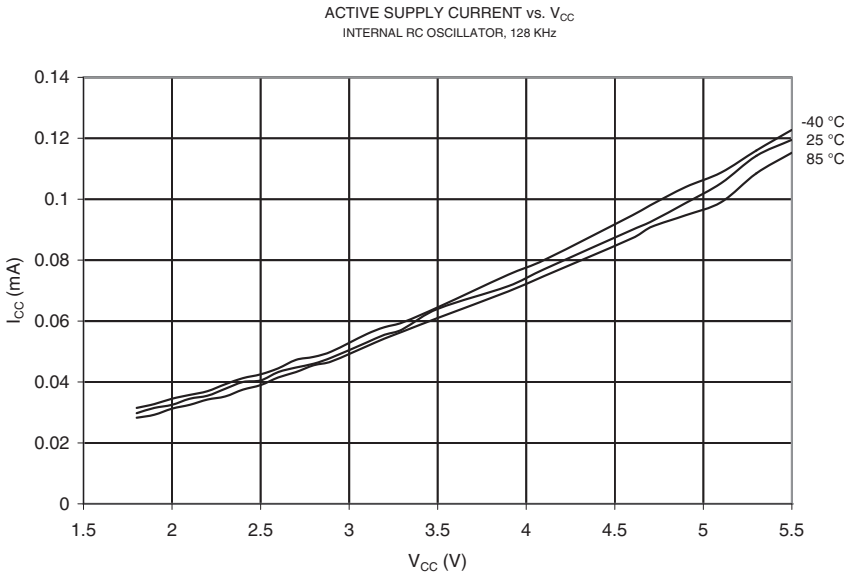


Figure 21-5. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz)



21.3 Idle Supply Current

Figure 21-6. Idle Supply Current vs. Low Frequency (0.1 - 1.0 MHz)

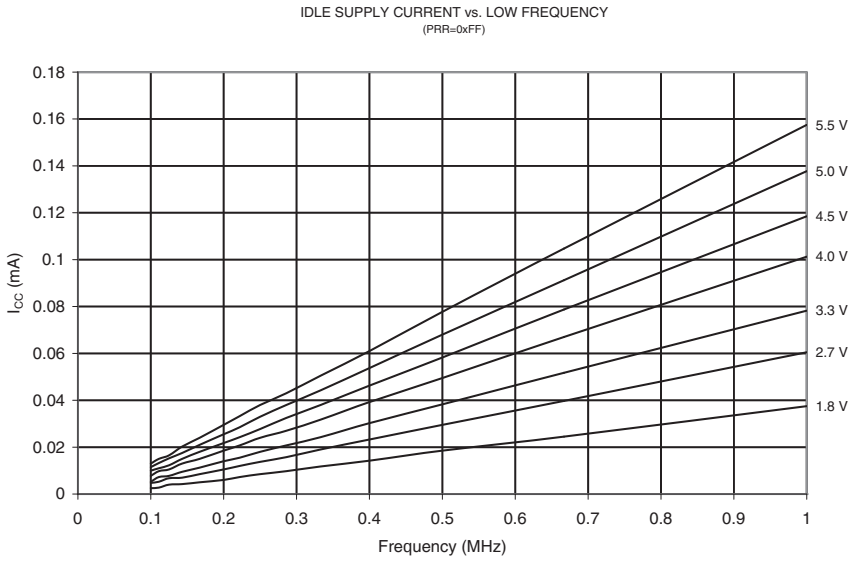


Figure 21-7. Idle Supply Current vs. Frequency (1 - 20 MHz)

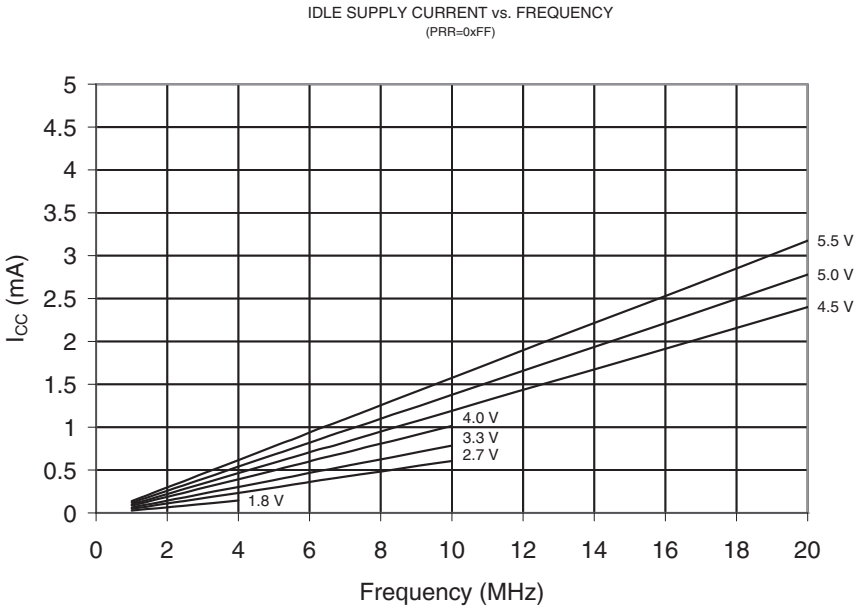


Figure 21-8. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz)

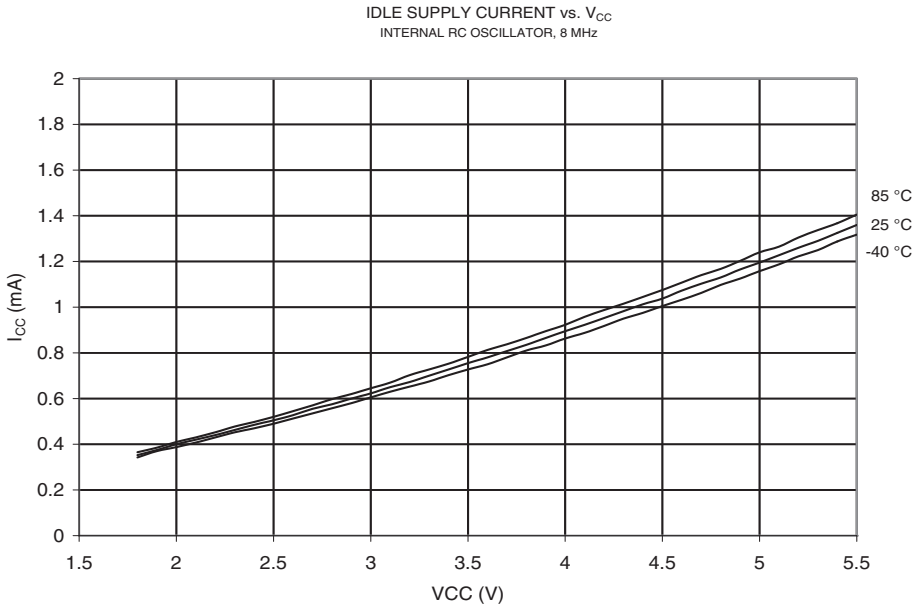


Figure 21-9. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz)

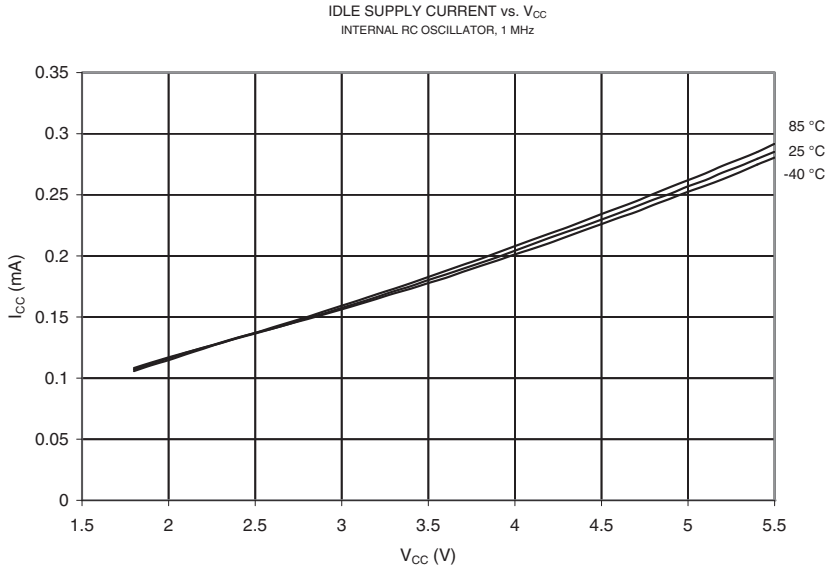
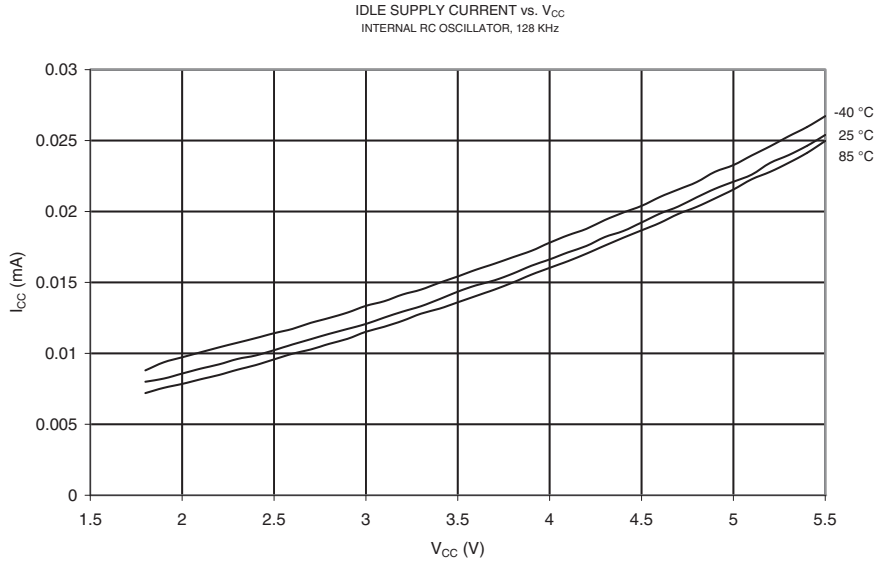


Figure 21-10. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz)



21.4 Power-down Supply Current

Figure 21-11. Power-down Supply Current vs. V_{CC} (Watchdog Timer Disabled)

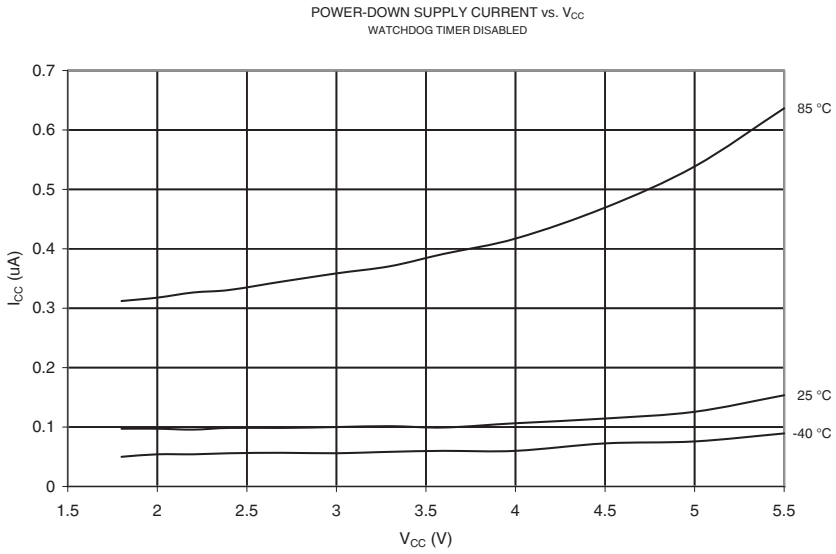
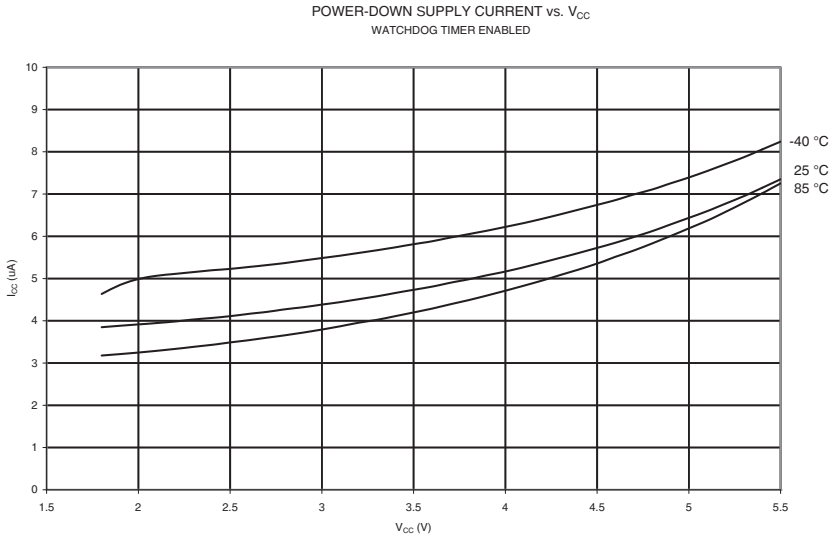
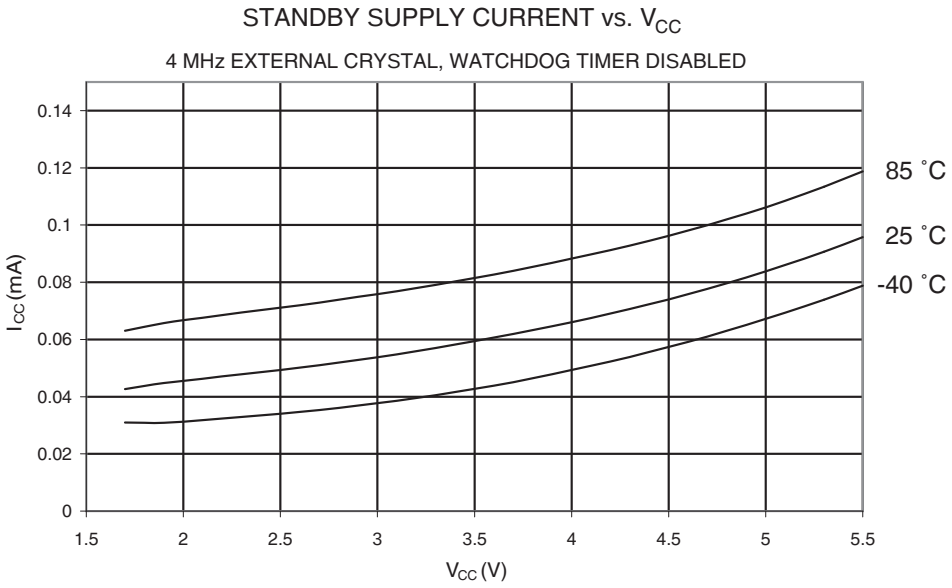


Figure 21-12. Power-down Supply Current vs. V_{CC} (Watchdog Timer Enabled)



21.5 Standby Supply Current

Figure 21-13. Standby Supply Current vs. V_{CC} (4 MHz External Crystal, Watchdog Timer Disabled)



21.6 Pin Pull-up

Figure 21-14. I/O pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 1.8V)

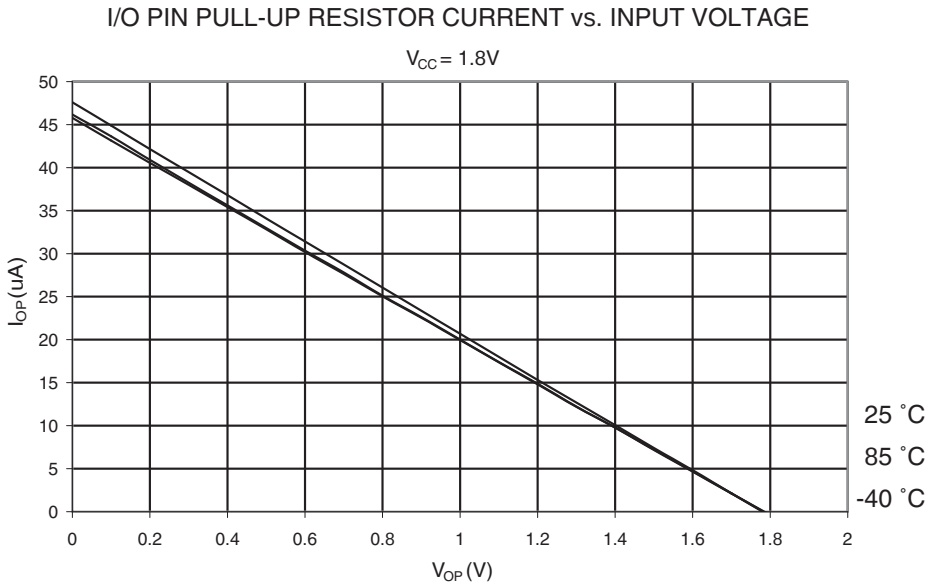


Figure 21-15. I/O Pin Pull-up Resistor Current vs. input Voltage (V_{CC} = 2.7V)

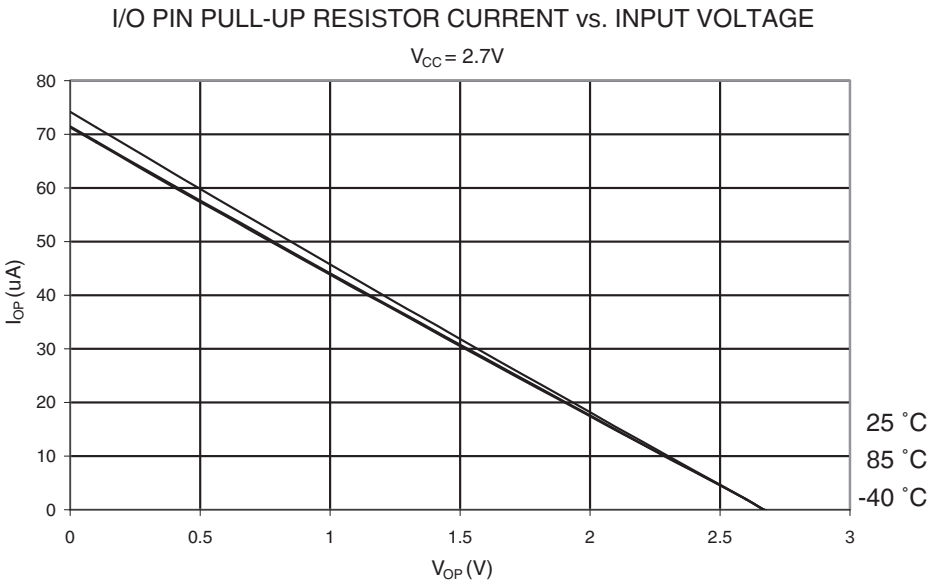


Figure 21-16. I/O pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5V)

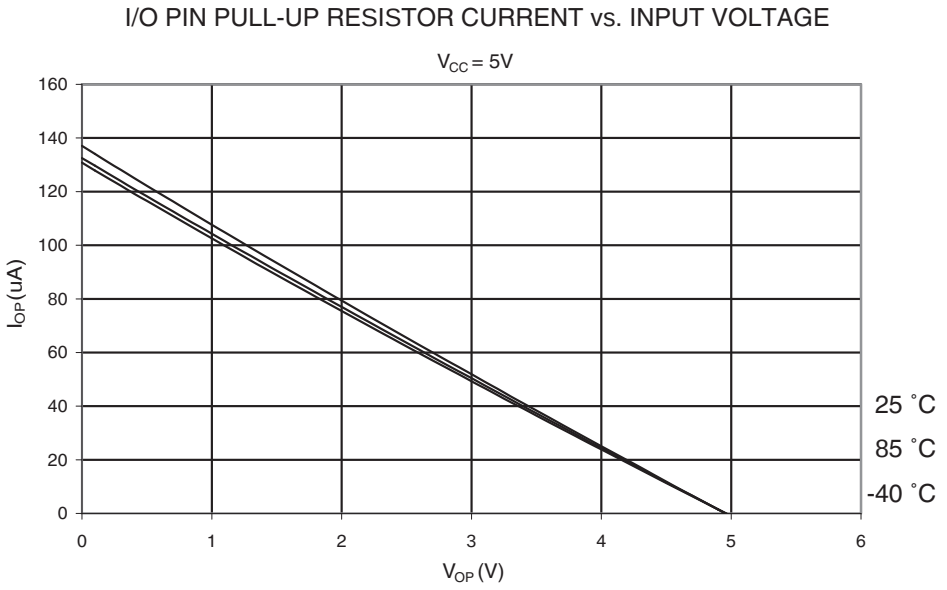


Figure 21-17. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 1.8V)

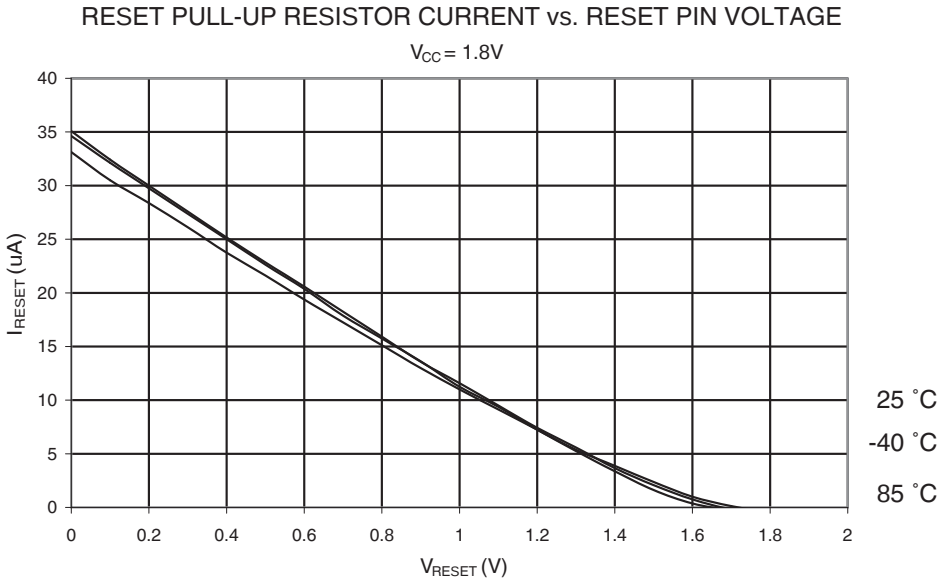


Figure 21-18. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 2.7V)

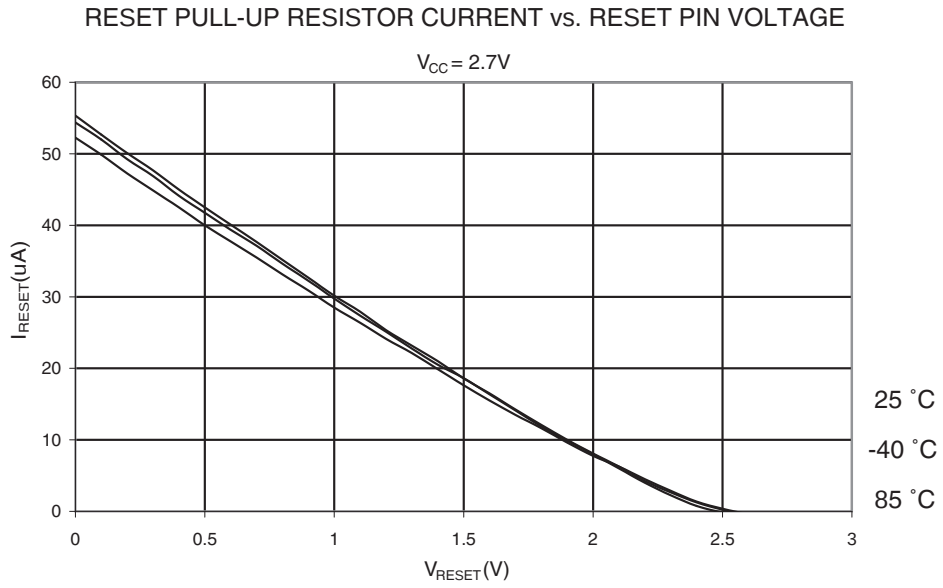
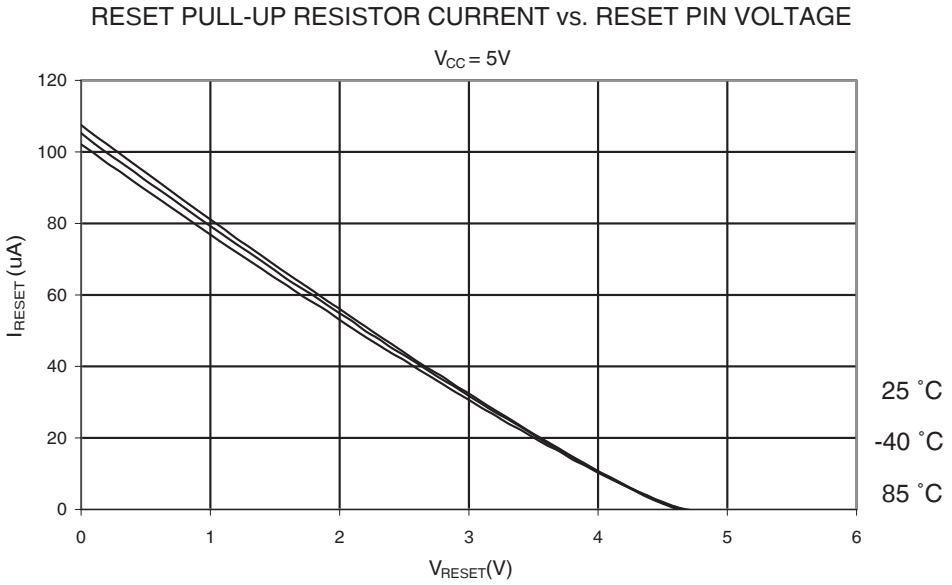


Figure 21-19. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5V)



21.7 Pin Driver Strength

Figure 21-20. I/O Pin Output Voltage vs. Sink Current (V_{CC} = 3V)

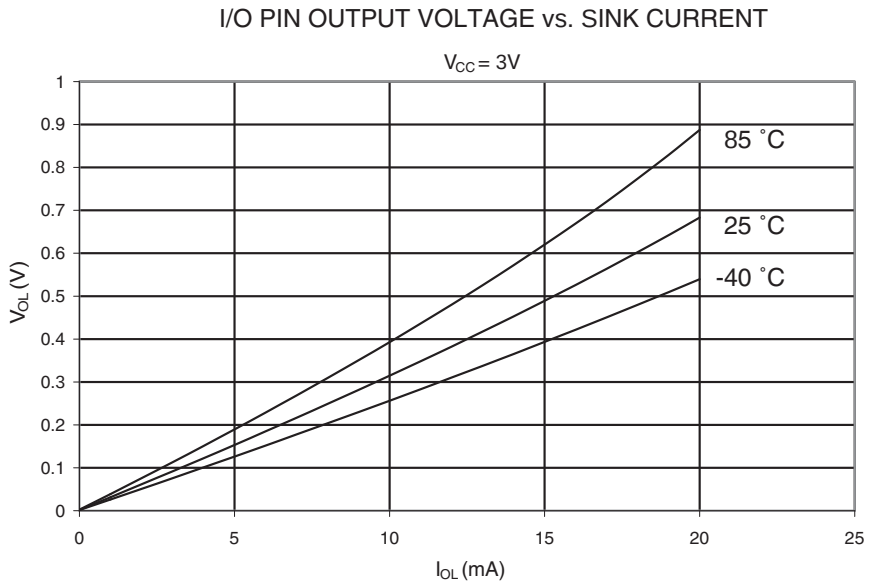


Figure 21-21. I/O pin Output Voltage vs. Sink Current (V_{CC} = 5V)

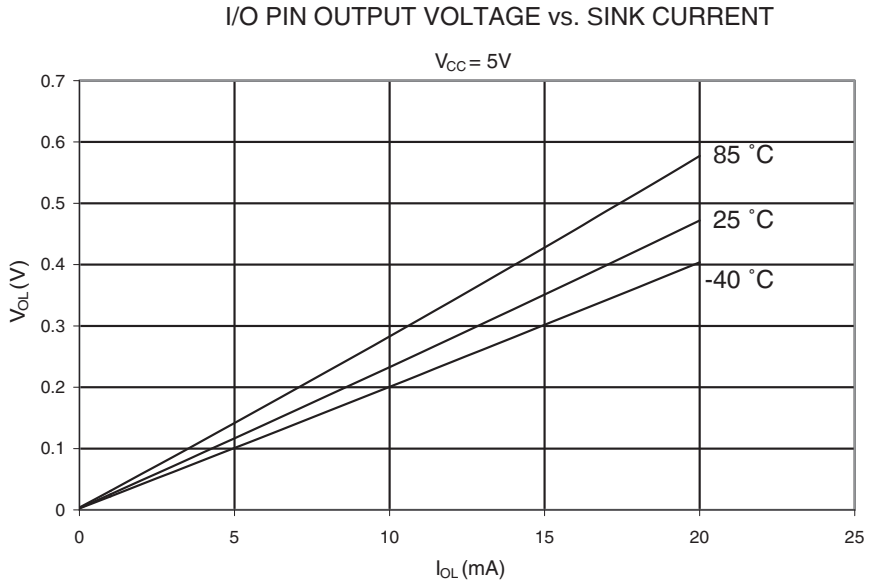


Figure 21-22. I/O Pin Output Voltage vs. Source Current (V_{CC} = 3V)

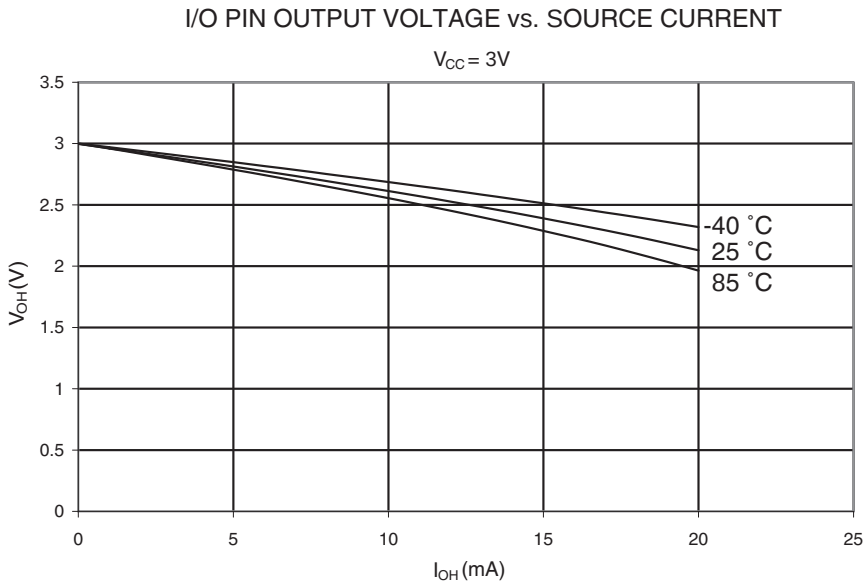


Figure 21-23. I/O Pin output Voltage vs. Source Current (V_{CC} = 5V)

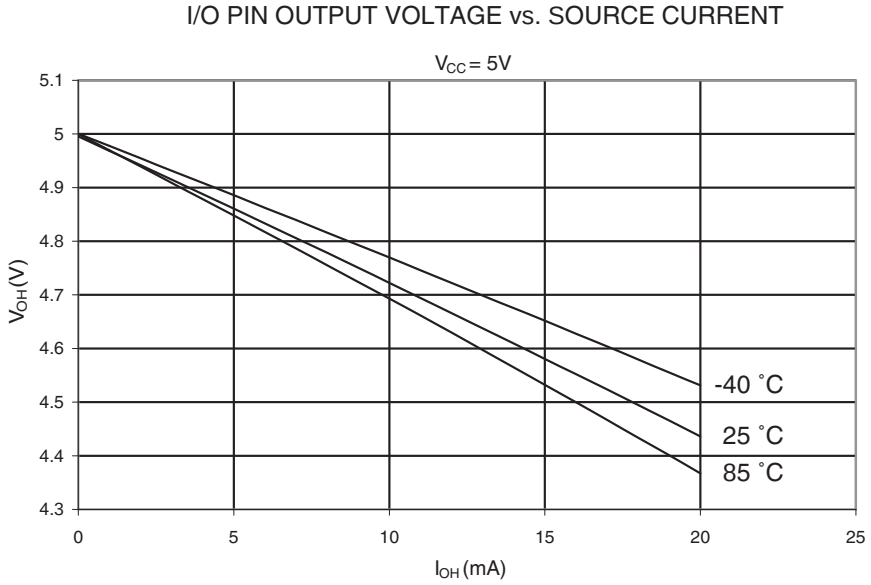


Figure 21-24. Reset Pin Output Voltage vs. Sink Current (V_{CC} = 3V)

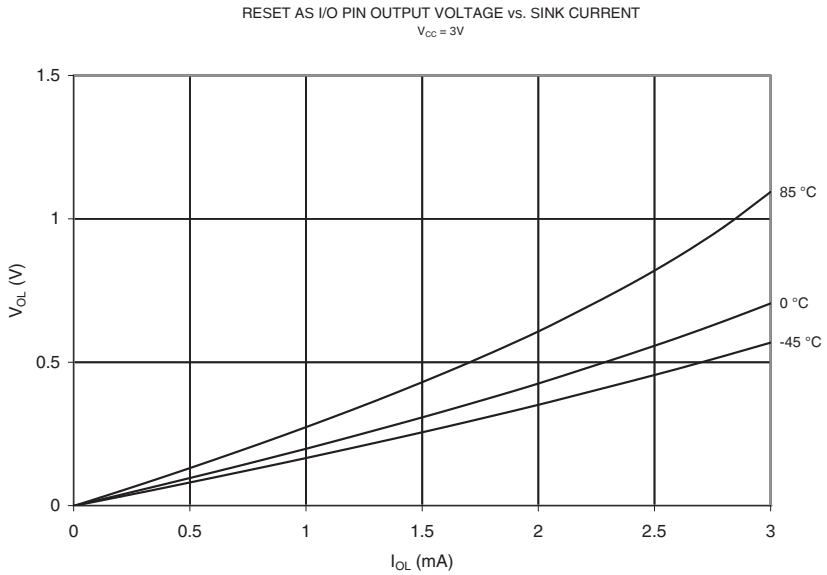


Figure 21-25. Reset Pin Output Voltage vs. Sink Current (V_{CC} = 5V)

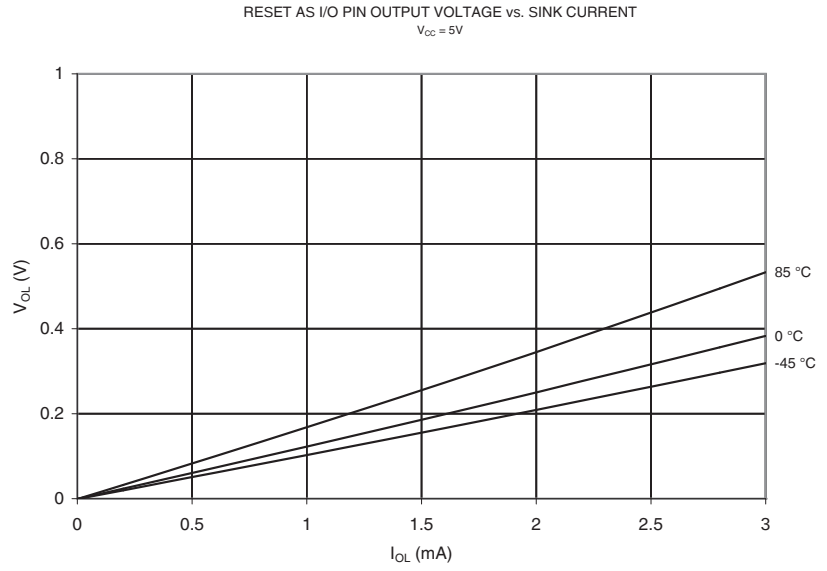


Figure 21-26. Reset Pin Output Voltage vs. Source Current (V_{CC} = 3V)

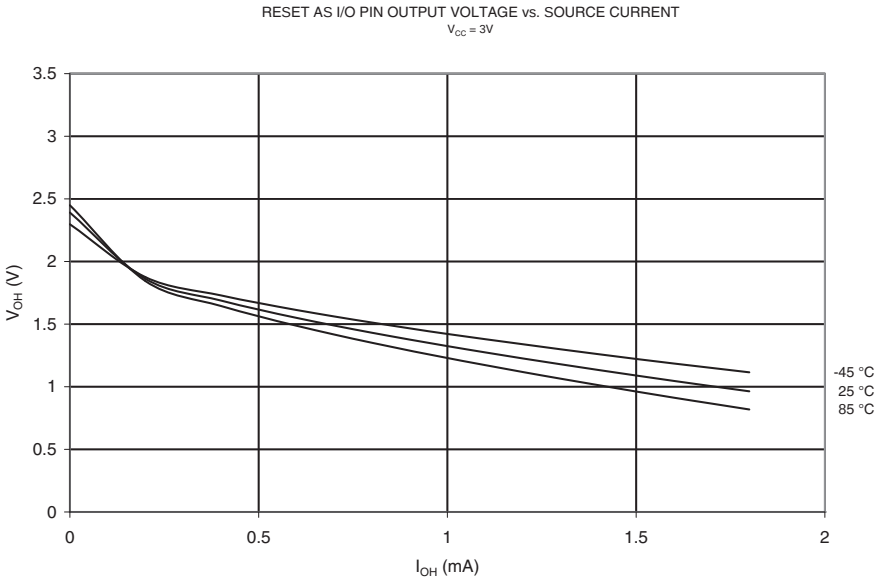
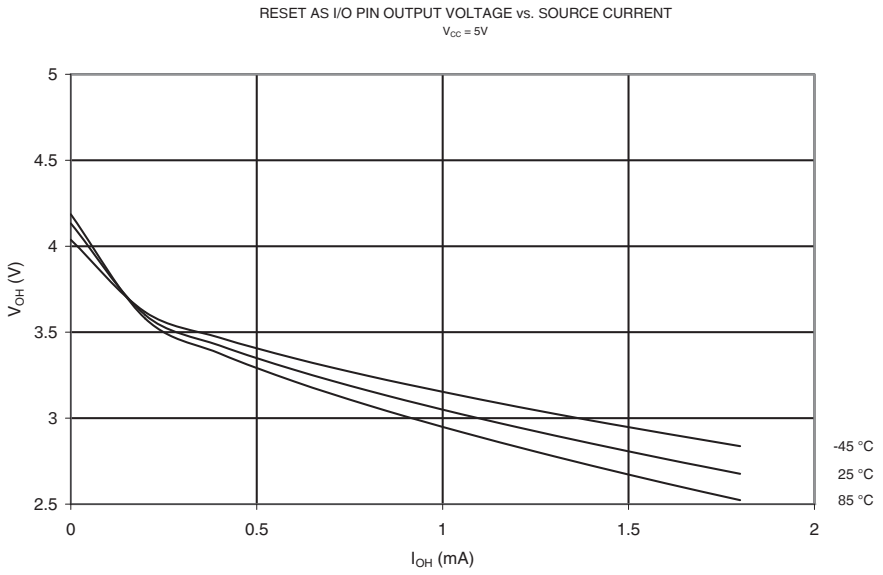


Figure 21-27. Reset Pin Output Voltage vs. Source Current (V_{CC} = 5V)



21.8 Pin Threshold and Hysteresis

Figure 21-28. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IH}, IO Pin Read as '1')

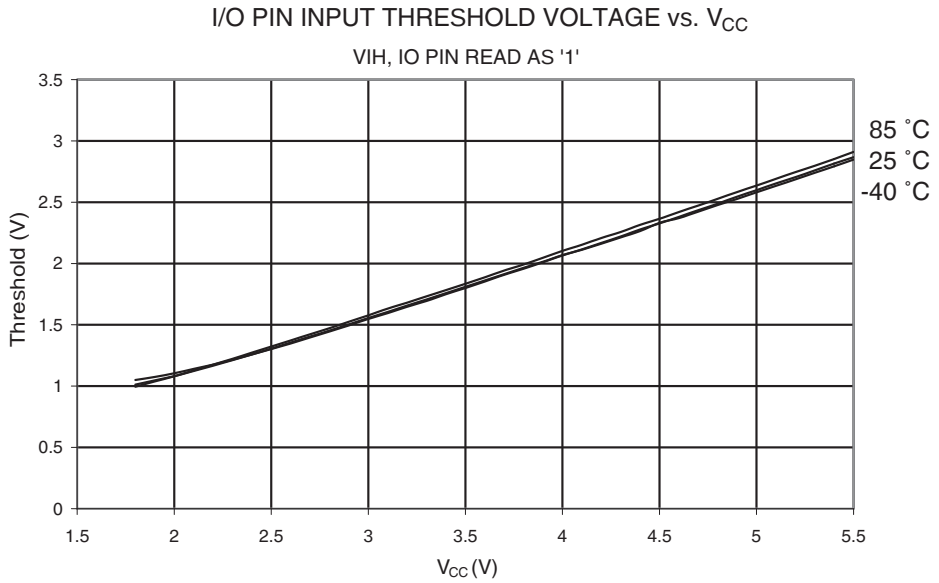


Figure 21-29. I/O Pin Input threshold Voltage vs. V_{CC} (V_{IL}, IO Pin Read as '0')

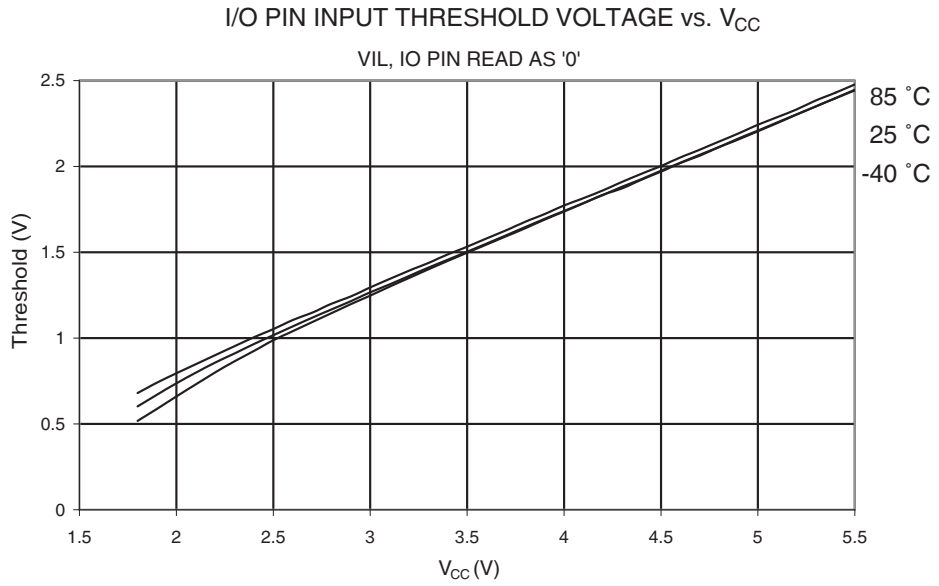


Figure 21-30. I/O Pin Input Hysteresis vs. V_{CC}

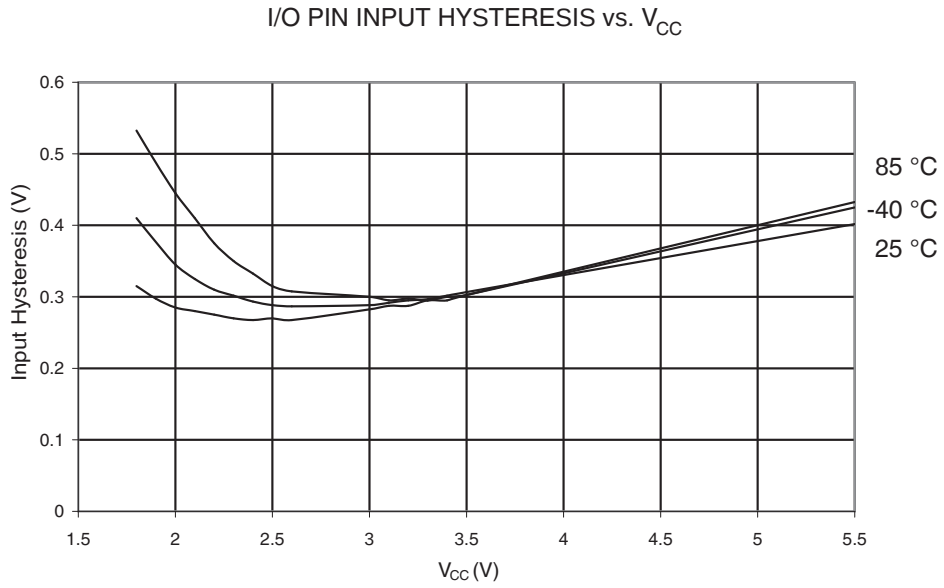


Figure 21-31. Reset Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin Threshold as '1')

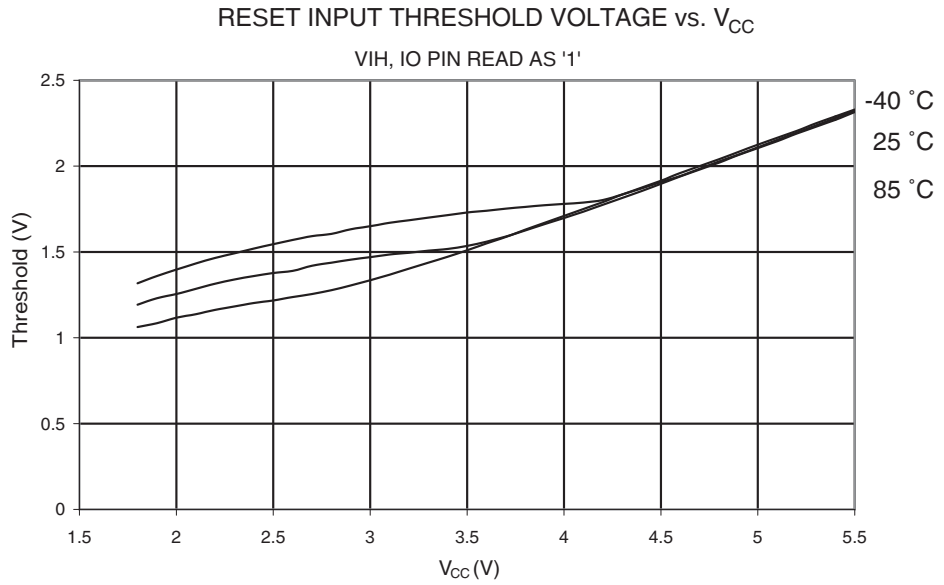


Figure 21-32. Reset Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O pin Read as '0')

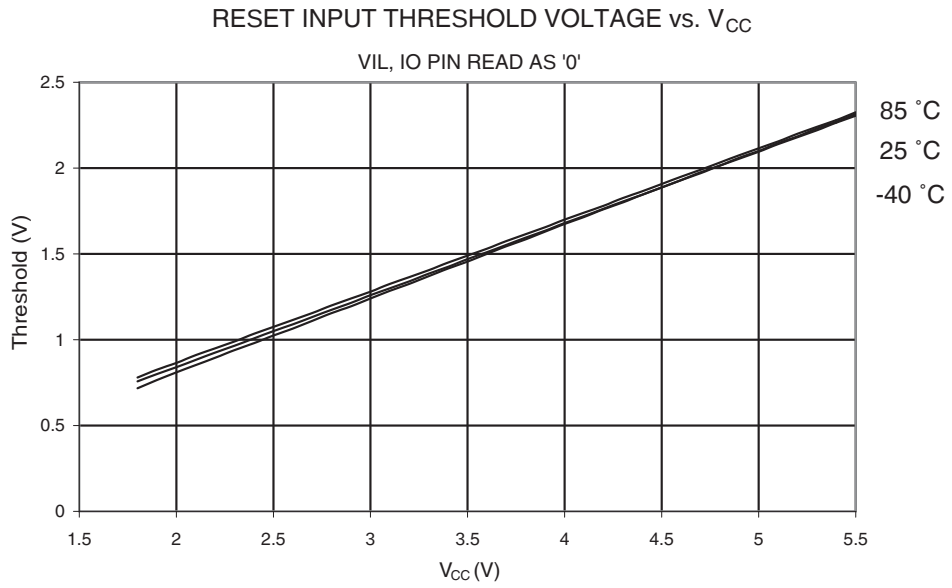


Figure 21-33. Reset Pin Input Hysteresis vs. V_{CC}

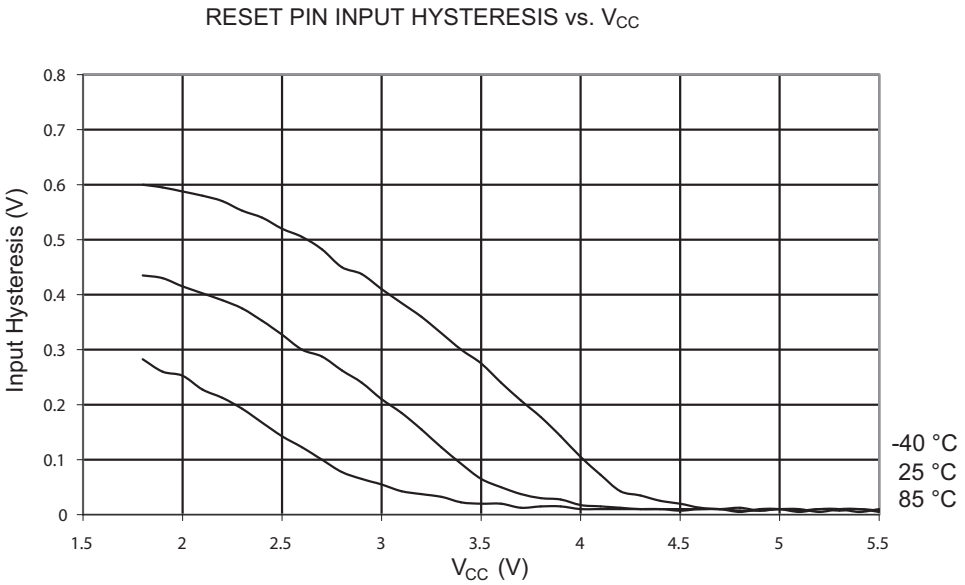
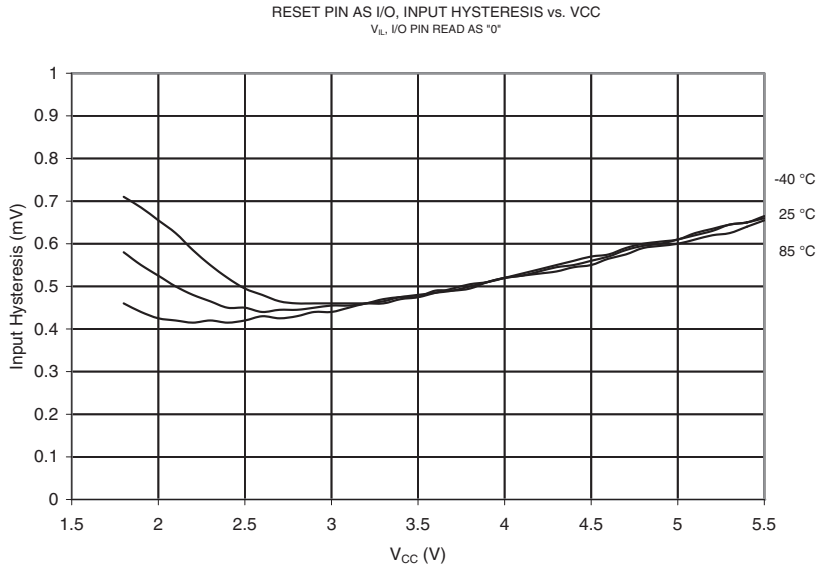


Figure 21-34. Reset Pin Input Hysteresis vs. V_{CC} (Reset Pin Used as I/O)



21.9 BOD Threshold and Analog Comparator Offset

Figure 21-35. BOD Threshold vs. Temperature (BODLEVEL is 4.3V)

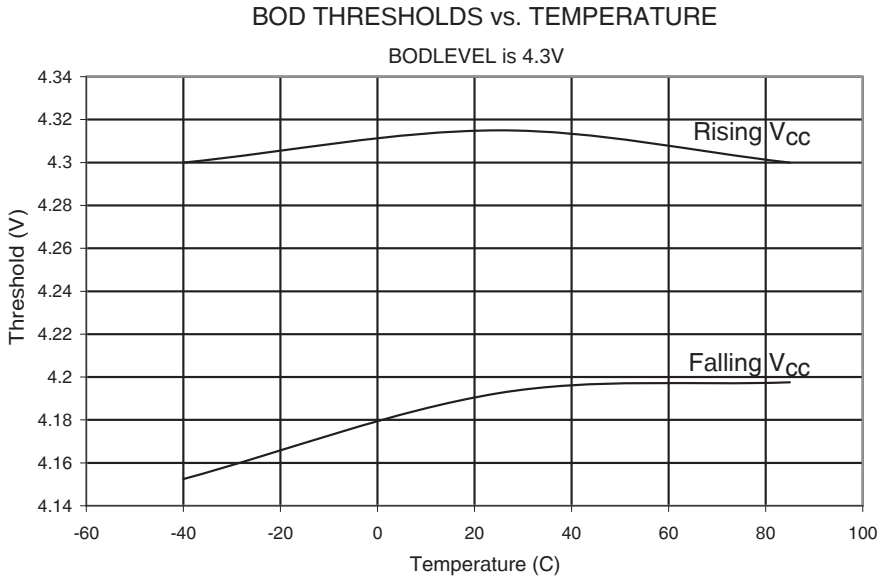


Figure 21-36. BOD Threshold vs, Temperature (BODLEVEL is 2.7V)

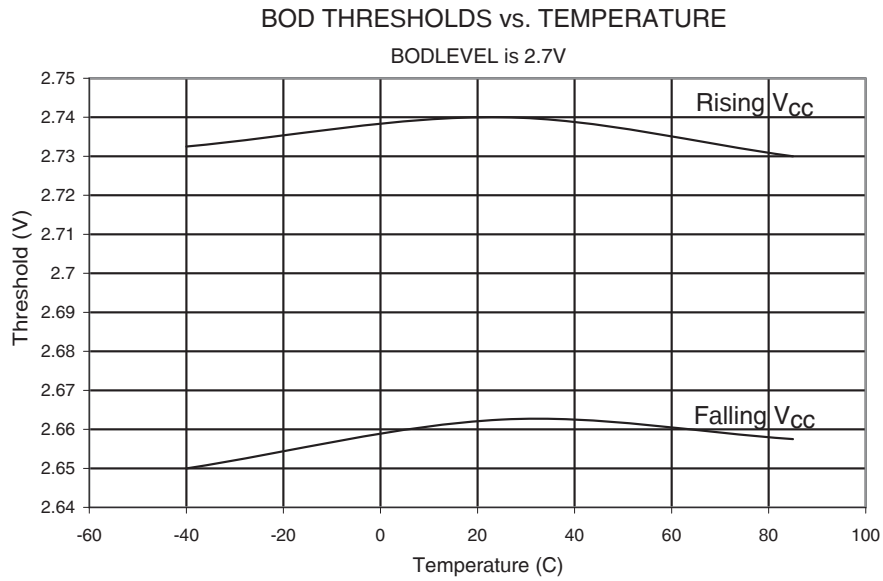
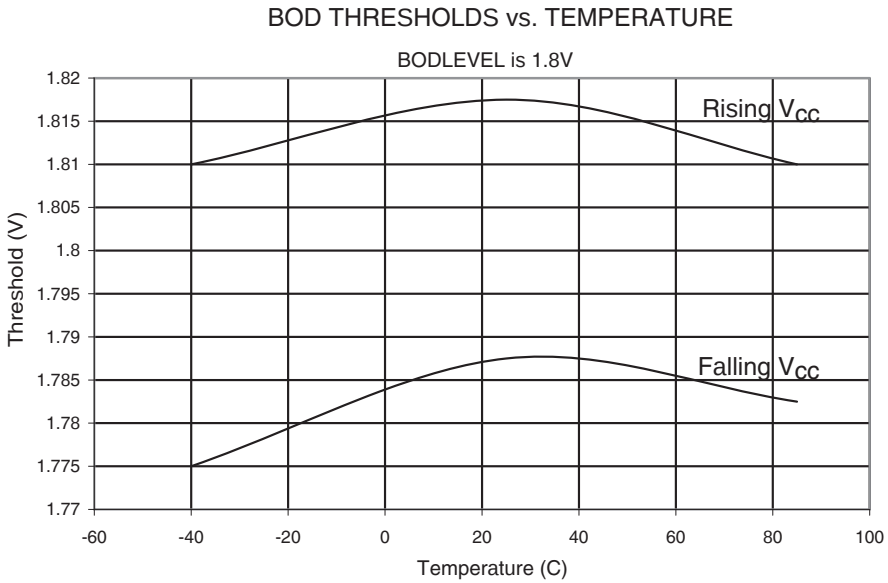


Figure 21-37. BOD Threshold vs. Temperature (BODLEVEL is 1.8V)



21.10 Internal Oscillator Speed

Figure 21-38. Watchdog Oscillator Frequency vs. V_{CC}

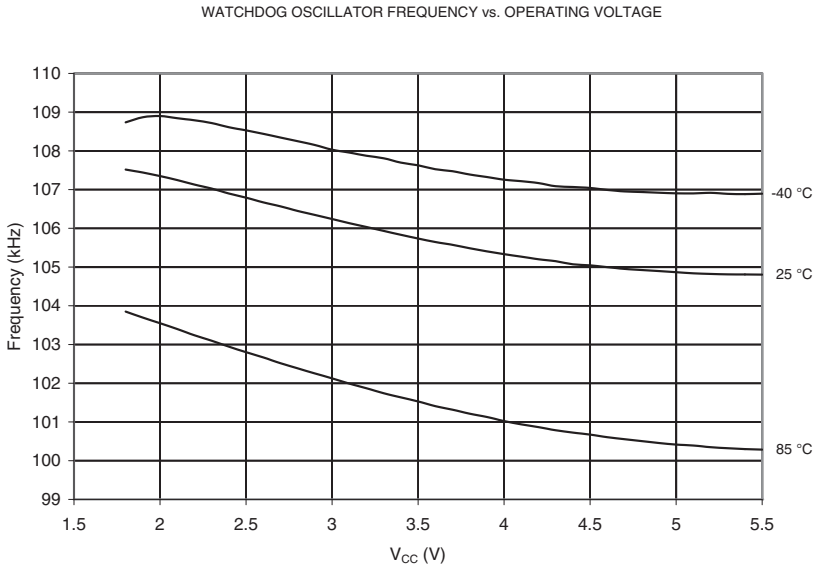


Figure 21-39. Watchdog Oscillator Frequency vs. Temperature

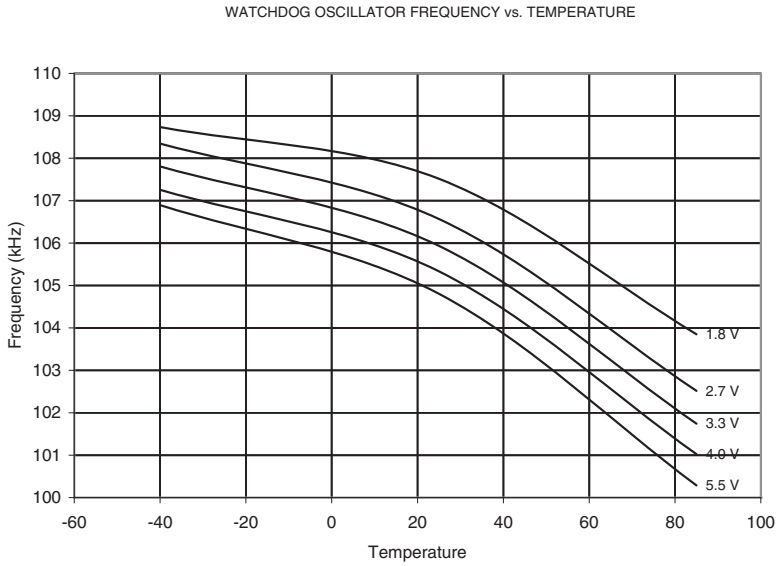


Figure 21-40. Calibrated 8 MHz RC Oscillator Frequency vs. V_{CC}

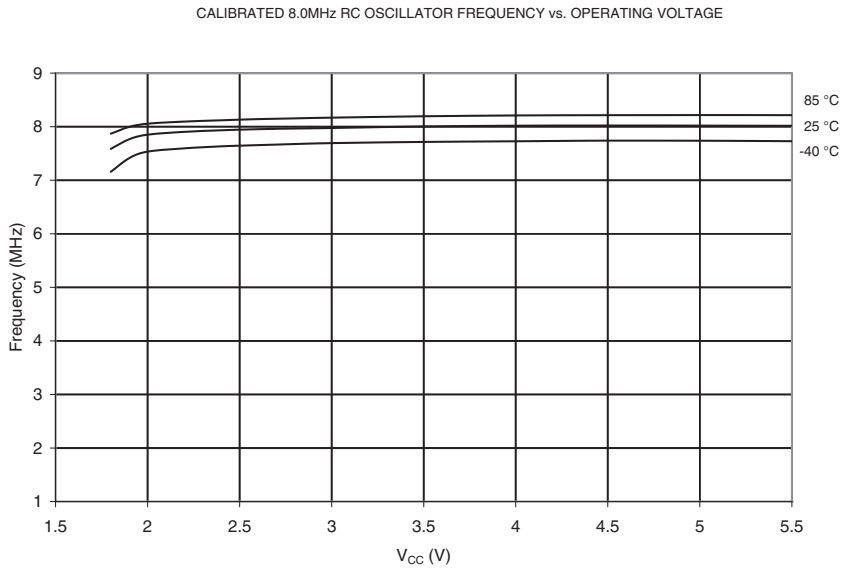


Figure 21-41. Calibrated 8 MHz RC oscillator Frequency vs. Temperature

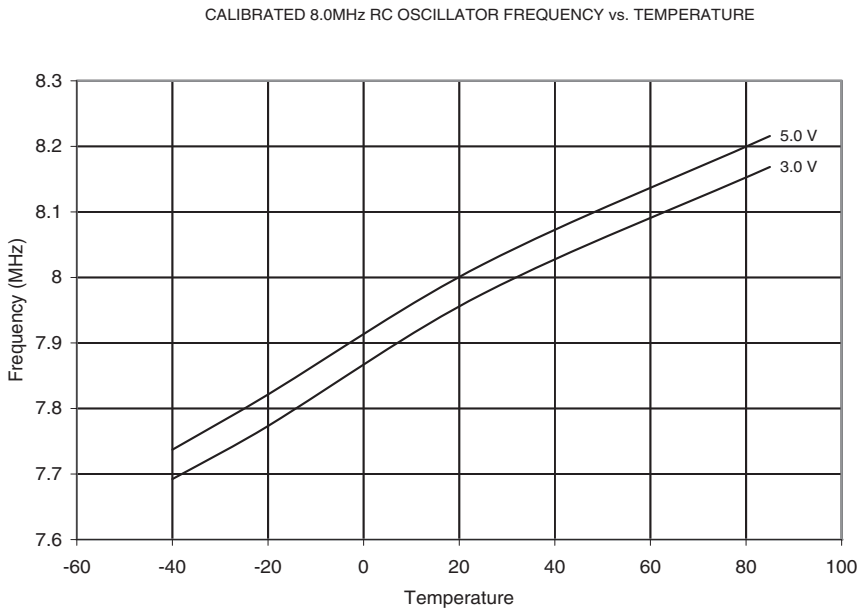
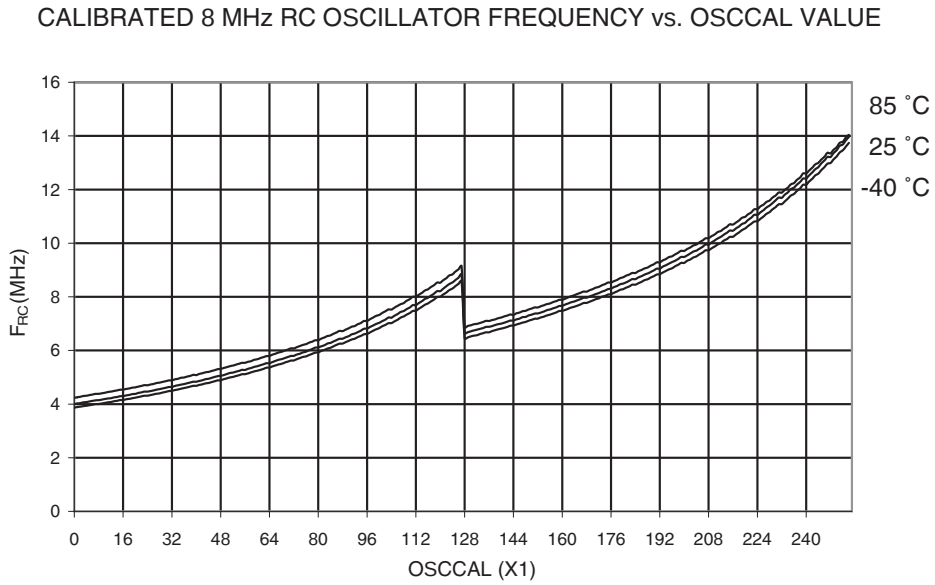


Figure 21-42. Calibrated 8 MHz RC Oscillator Frequency vs, OSCCAL Value



21.11 Current Consumption of Peripheral Units

Figure 21-43. ADC Current vs. V_{CC}

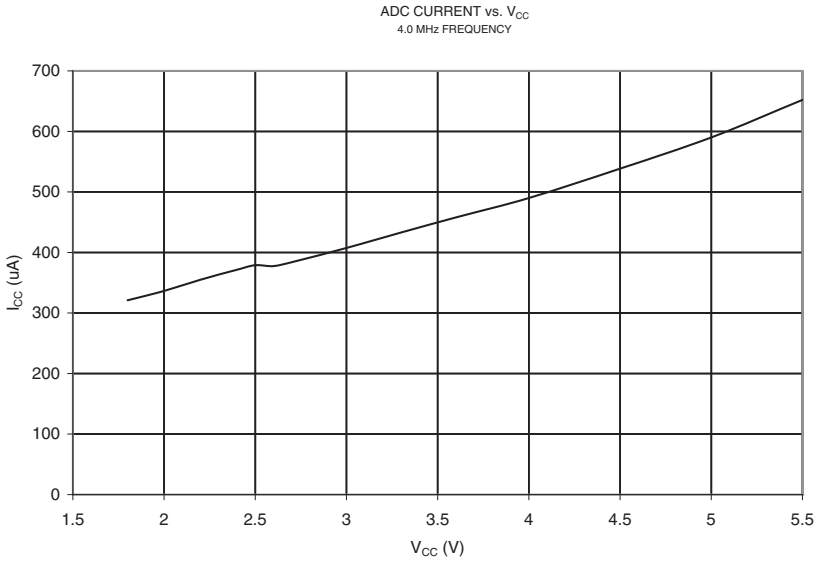


Figure 21-44. AREF External Reference Current vs. V_{CC}

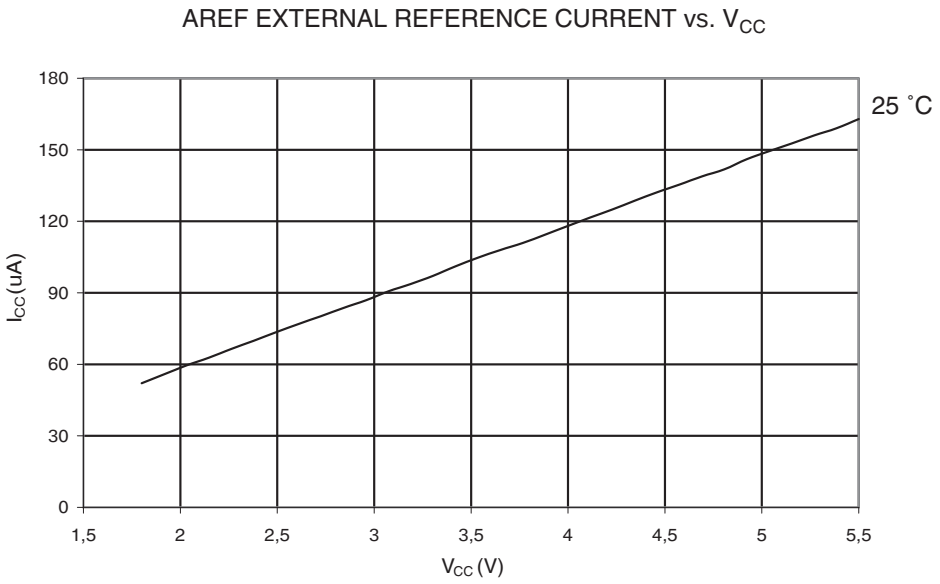


Figure 21-45. Analog Comparator Current vs. V_{CC}

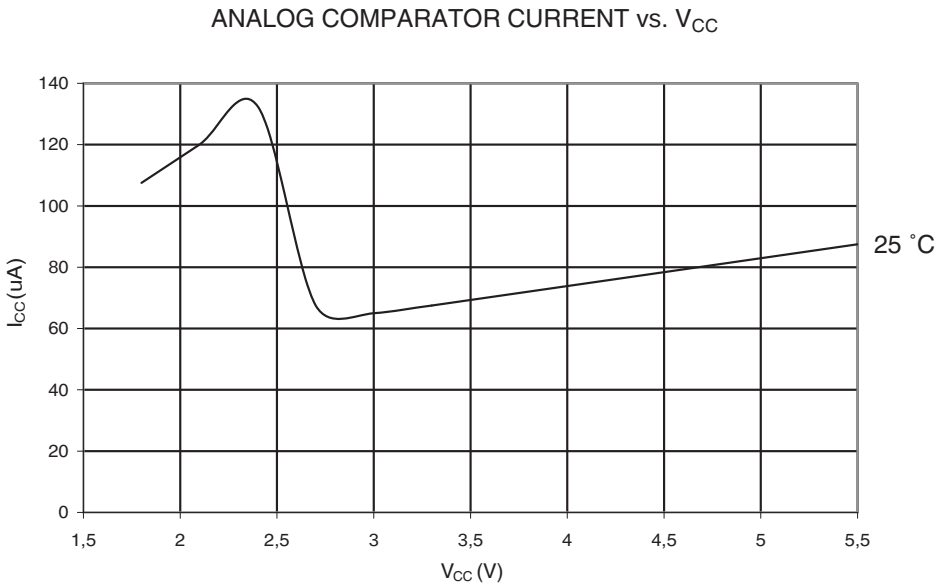


Figure 21-46. Programming Current vs. V_{CC} (ATtiny24)

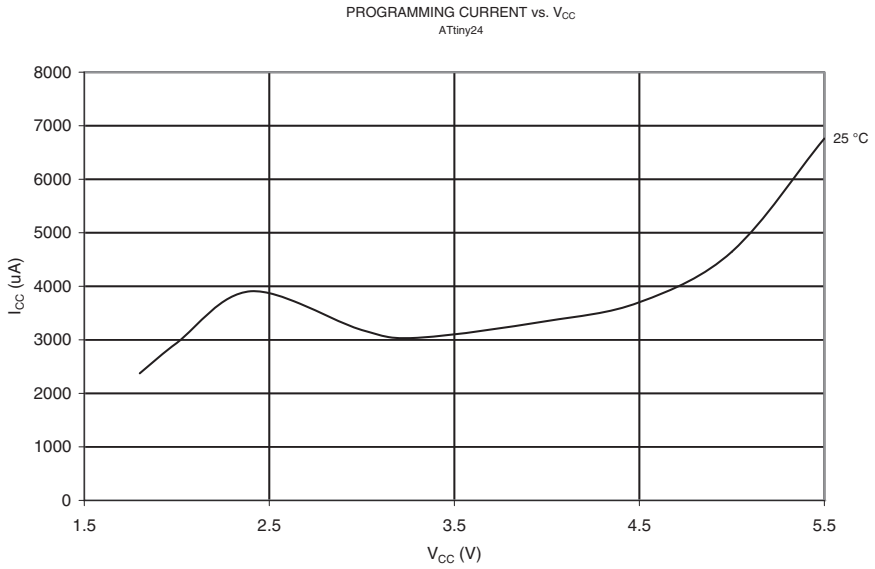


Figure 21-47. Programming Current vs. V_{CC} (ATtiny44)

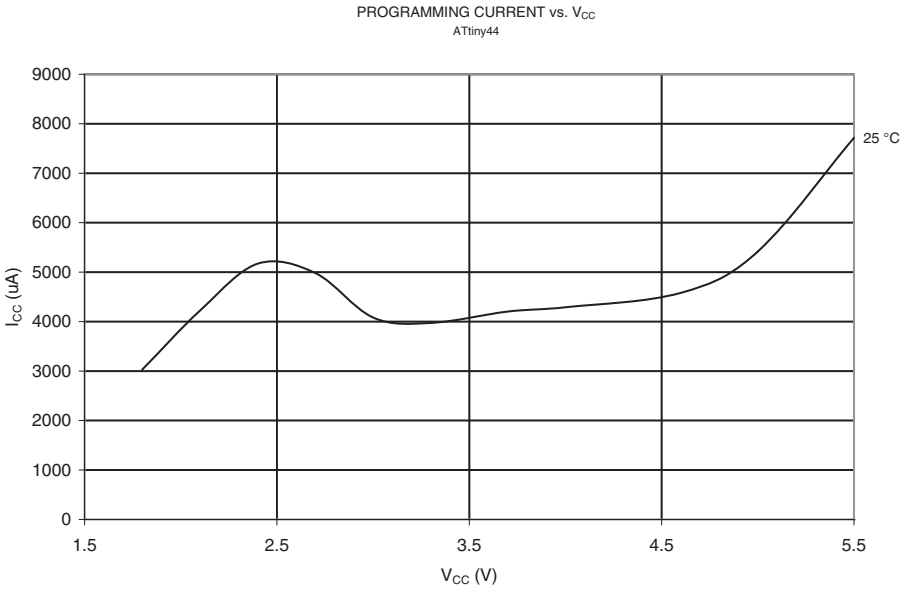


Figure 21-48. Programming Current vs. V_{CC} (ATtiny84)

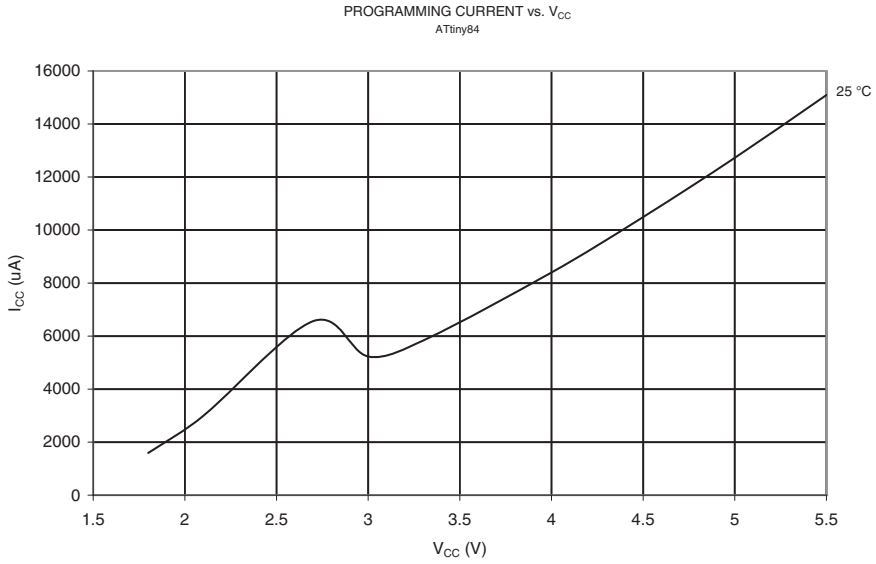


Figure 21-49. Brownout Detector Current vs. V_{CC}

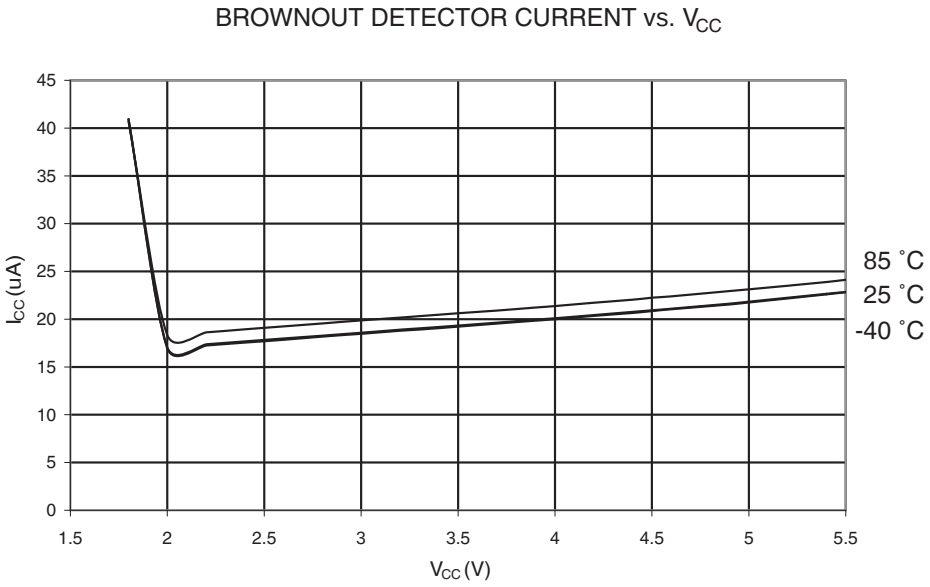
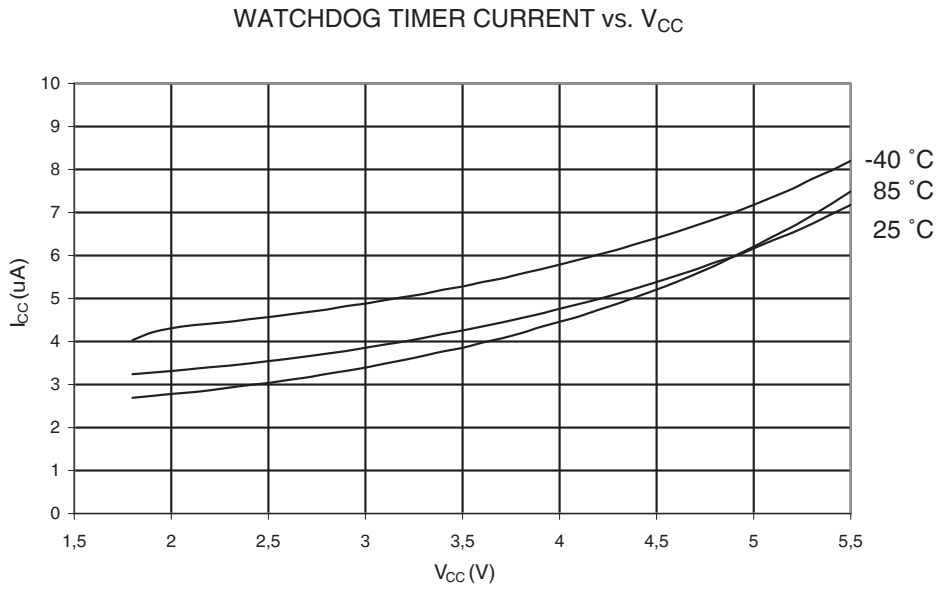


Figure 21-50. Watchdog Timer Current vs. V_{CC}



21.12 Current Consumption in Reset and Reset Pulsewidth

Figure 21-51. Reset Supply Current vs. V_{CC} (0.1 - 1.0 MHz, excluding Current Through the Reset Pull-up)

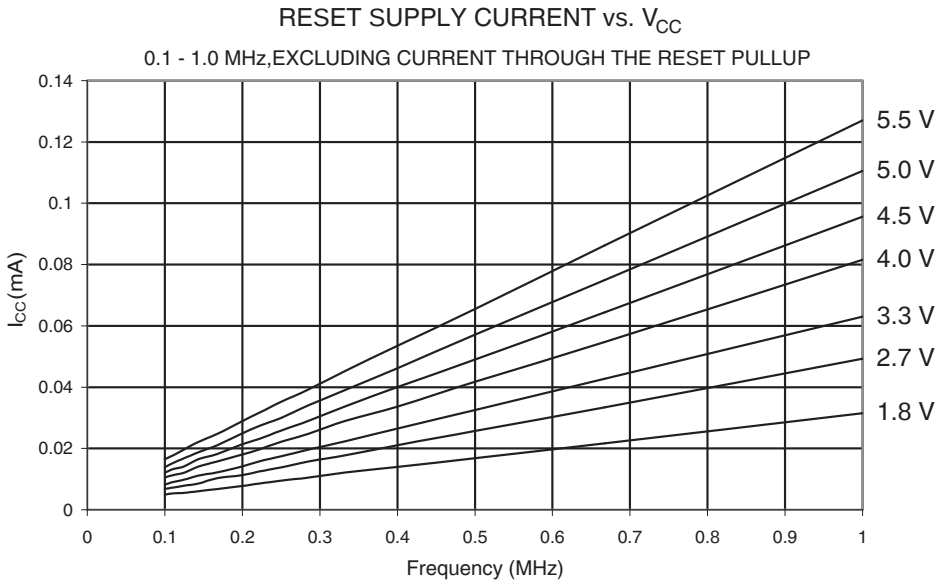


Figure 21-52. Reset Supply Current vs. V_{CC} (1 - 20 MHz, Excluding Current Through the Reset Pull-up)

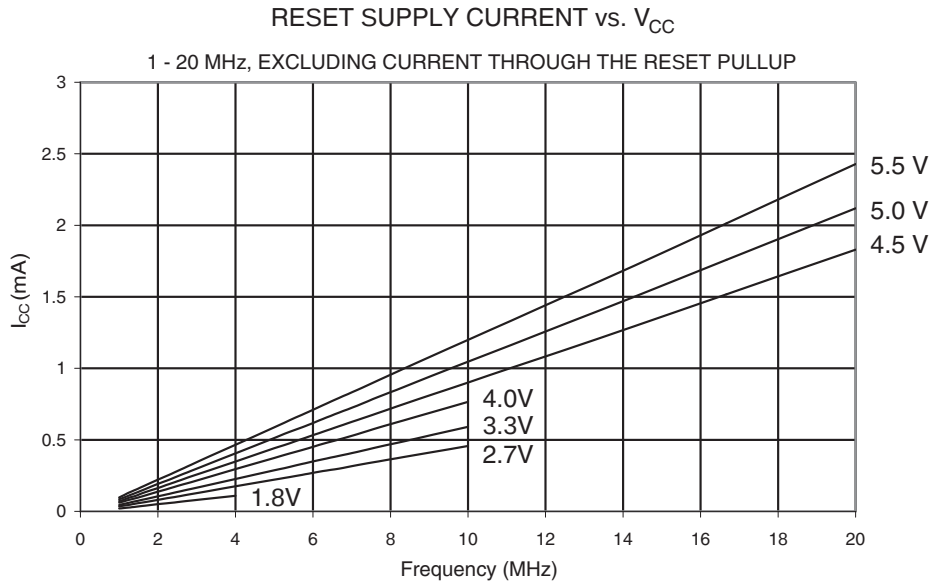
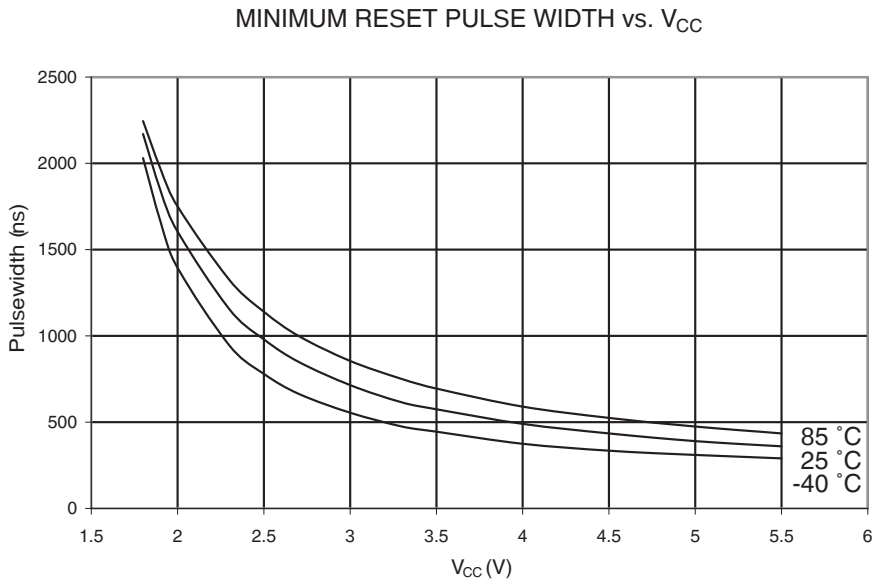


Figure 21-53. Minimum Reset Pulse Width vs. V_{CC}



22. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C	Page 8
0x3E (0x5E)	SPH	–	–	–	–	–	–	SP9	SP8	Page 11
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	Page 11
0x3C (0x5C)	OCR0B	Timer/Counter0 – Output Compare Register B								Page 85
0x3B (0x5B)	GIMSK	–	INT0	PCIE1	PCIE0	–	–	–	–	Page 51
0x3A (0x5A	GIFR	–	INTF0	PCIF1	PCIF0	–	–	–	–	Page 52
0x39 (0x59)	TIMSK0	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	Page 85
0x38 (0x58)	TIFR0		–	–	–	–	OCF0B	OCF0A	TOV0	Page 85
0x37 (0x57)	SPMCSR	–	–	RSIG	CTPB	RFLB	PGWRT	PGERS	SPMEN	Page 157
0x36 (0x56)	OCR0A	Timer/Counter0 – Output Compare Register A								Page 84
0x35 (0x55)	MCUCR	BODS	PUD	SE	SM1	SM0	BODSE	ISC01	ISC00	Pages 36, 51, and 67
0x34 (0x54)	MCUSR	–	–	–	–	WDRF	BORF	EXTRF	PORF	Page 45
0x33 (0x53)	TCCR0B	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	Page 83
0x32 (0x52)	TCNT0	Timer/Counter0								Page 84
0x31 (0x51)	OSCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	Page 30
0x30 (0x50)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	–		WGM01	WGM00	Page 80
0x2F (0x4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	–		WGM11	WGM10	Page 108
0x2E (0x4E)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	Page 110
0x2D (0x4D)	TCNT1H	Timer/Counter1 – Counter Register High Byte								Page 112
0x2C (0x4C)	TCNT1L	Timer/Counter1 – Counter Register Low Byte								Page 112
0x2B (0x4B)	OCR1AH	Timer/Counter1 – Compare Register A High Byte								Page 112
0x2A (0x4A)	OCR1AL	Timer/Counter1 – Compare Register A Low Byte								Page 112
0x29 (0x49)	OCR1BH	Timer/Counter1 – Compare Register B High Byte								Page 112
0x28 (0x48)	OCR1BL	Timer/Counter1 – Compare Register B Low Byte								Page 112
0x27 (0x47)	DWDR	DWDR[7:0]								Page 152
0x26 (0x46)	CLKPR	CLKPCE	–	–	–	CLKPS3	CLKPS2	CLKPS1	CLKPS0	Page 31
0x25 (0x45)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								Page 113
0x24 (0x44)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								Page 113
0x23 (0x43)	GTCCR	TSM	–	–	–	–	–	–	PSR10	Page 116
0x22 (0x42)	TCCR1C	FOC1A	FOC1B	–	–	–	–	–	–	Page 111
0x21 (0x41)	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	Page 45
0x20 (0x40)	PCMSK1	–	–	–	–	PCINT11	PCINT10	PCINT9	PCINT8	Page 52
0x1F (0x3F)	EEARH	–	–	–	–	–	–	–	EEAR8	Page 20
0x1E (0x3E)	EEARL	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	Page 21
0x1D (0x3D)	EEDR	EEPROM Data Register								Page 21
0x1C (0x3C)	EECR	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	Page 21
0x1B (0x3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	Page 67
0x1A (0x3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	Page 67
0x19 (0x39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	Page 68
0x18 (0x38)	PORTB	–	–	–	–	PORTB3	PORTB2	PORTB1	PORTB0	Page 68
0x17 (0x37)	DDRB	–	–	–	–	DDB3	DDB2	DDB1	DDB0	Page 68
0x16 (0x36)	PINB	–	–	–	–	PINB3	PINB2	PINB1	PINB0	Page 68
0x15 (0x35)	GPIOR2	General Purpose I/O Register 2								Page 23
0x14 (0x34)	GPIOR1	General Purpose I/O Register 1								Page 23
0x13 (0x33)	GPIOR0	General Purpose I/O Register 0								Page 23
0x12 (0x32)	PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	Page 53
0x11 (0x31))	Reserved	–								
0x10 (0x30)	USIBR	USI Buffer Register								Page 125
0x0F (0x2F)	USIDR	USI Data Register								Page 124
0x0E (0x2E)	USISR	USISIF	USIOIF	USIPF	USIDC	USICNT3	USICNT2	USICNT1	USICNT0	Page 125
0x0D (0x2D)	USICR	USISIE	USIOIE	USIWM1	USIWM0	USICS1	USICS0	USICLK	USITC	Page 126
0x0C (0x2C)	TIMSK1	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	Page 113
0x0B (0x2B)	TIFR1	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	Page 114
0x0A (0x2A)	Reserved	–								
0x09 (0x29)	Reserved	–								
0x08 (0x28)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	Page 130
0x07 (0x27)	ADMUX	REFS1	REFS0	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0	Page 145
0x06 (0x26)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	Page 147
0x05 (0x25)	ADCH	ADC Data Register High Byte								Page 149
0x04 (0x24)	ADCL	ADC Data Register Low Byte								Page 149
0x03 (0x23)	ADCSRB	BIN	ACME	–	ADLAR	–	ADTS2	ADTS1	ADTS0	Page 131, Page 149
0x02 (0x22)	Reserved	–								
0x01 (0x21)	DIDR0	ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	Page 131, Page 150
0x00 (0x20)	PRR	–	–	–	–	PRTIM1	PRTIM0	PRUSI	PRADC	Page 37

Note:
1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
2. 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.
3. Some of the Status Flags are cleared by writing a logical one to them. Note that, unlike most other AVRs, the CBI and SBI instructions will only operation 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.

23. 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
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z,C,N,V	1

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(z) ← R1:R0	None	
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
MCU CONTROL INSTRUCTIONS					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/Timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

24. Ordering Information

24.1 ATtiny24

Speed (MHz)	Power Supply	Ordering Code[1]	Package[2]	Operational Range
10	1.8 - 5.5V	ATtiny24V-10SSU ATtiny24V-10PU ATtiny24V-10MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATtiny24-20SSU ATtiny24-20PU ATtiny24-20MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)

Notes: 1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

Package Type	
14S1	14-lead, 0.150" Wide Body, Plastic Gull Wing Small Outline Package (SOIC)
14P3	14-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
20M1	20-pad, 4 x 4 x 0.8 mm Body, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

24.2 ATtiny44

Speed (MHz)	Power Supply	Ordering Code[1]	Package[2]	Operational Range
10	1.8 - 5.5V	ATtiny44V-10SSU ATtiny44V-10PU ATtiny44V-10MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATtiny44-20SSU ATtiny44-20PU ATtiny44-20MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)

Notes: 1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

Package Type	
14S1	14-lead, 0.150" Wide Body, Plastic Gull Wing Small Outline Package (SOIC)
14P3	14-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
20M1	20-pad, 4 x 4 x 0.8 mm Body, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

24.3 ATtiny84

Speed (MHz)	Power Supply	Ordering Code[1]	Package[2]	Operational Range
10	1.8 - 5.5V	ATtiny84V-10SSU ATtiny84V-10PU ATtiny84V-10MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATtiny84-20SSU ATtiny84-20PU ATtiny84-20MU	14S1 14P3 20M1	Industrial (-40°C to 85°C)

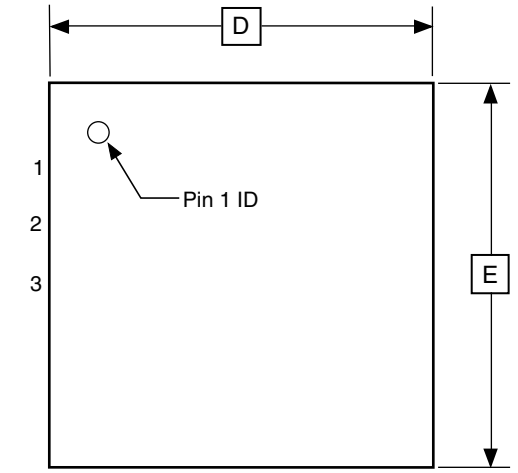
Notes: 1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

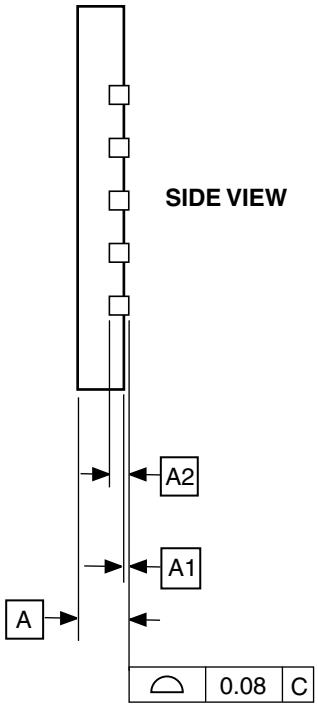
Package Type	
14S1	14-lead, 0.150" Wide Body, Plastic Gull Wing Small Outline Package (SOIC)
14P3	14-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
20M1	20-pad, 4 x 4 x 0.8 mm Body, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

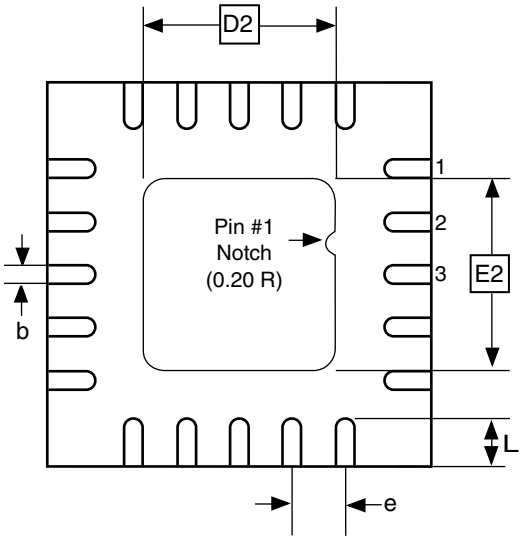
25. Packaging Information

25.1 20M1



TOP VIEW





BOTTOM VIEW

COMMON DIMENSIONS
(Unit of Measure = mm)

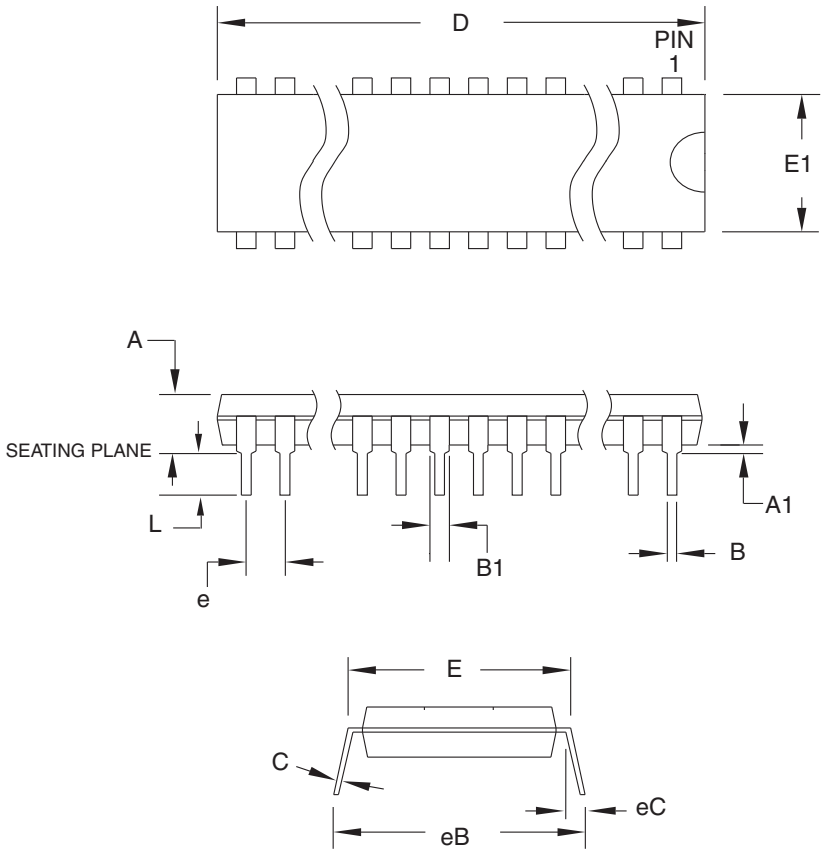
SYMBOL	MIN	NOM	MAX	NOTE
A	0.70	0.75	0.80	
A1	–	0.01	0.05	
A2		0.20 REF		
b	0.18	0.23	0.30	
D		4.00 BSC		
D2	2.45	2.60	2.75	
E		4.00 BSC		
E2	2.45	2.60	2.75	
e		0.50 BSC		
L	0.35	0.40	0.55	

Note: Reference JEDEC Standard MO-220, Fig. 1 (SAW Singulation) WGGD-5.

10/27/04

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**20M1**, 20-pad, 4 x 4 x 0.8 mm Body, Lead Pitch 0.50 mm, 2.6 mm Exposed Pad, Micro Lead Frame Package (MLF)	20M1	A

25.2 14P3



COMMON DIMENSIONS
(Unit of Measure = mm)

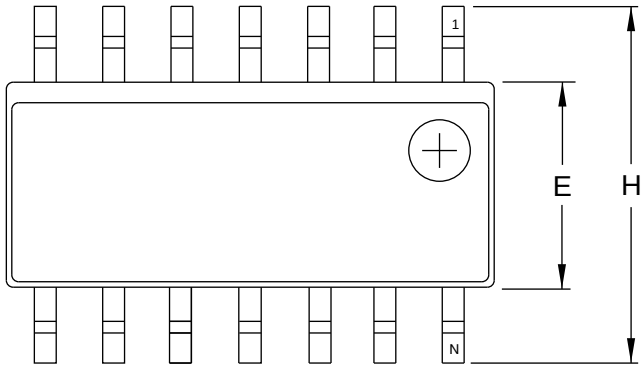
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	5.334	
A1	0.381	–	–	
D	18.669	–	19.685	Note 2
E	7.620	–	8.255	
E1	6.096	–	7.112	Note 2
B	0.356	–	0.559	
B1	1.143	–	1.778	
L	2.921	–	3.810	
C	0.203	–	0.356	
eB	–	–	10.922	
eC	0.000	–	1.524	
e	2.540 TYP			

Notes: 1. This package conforms to JEDEC reference MS-001, Variation AA.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

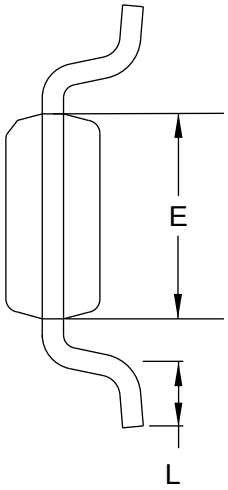
11/02/05

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**14P3**, 14-lead (0.300"/7.62 mm Wide) Plastic Dual Inline Package (PDIP)	14P3	A

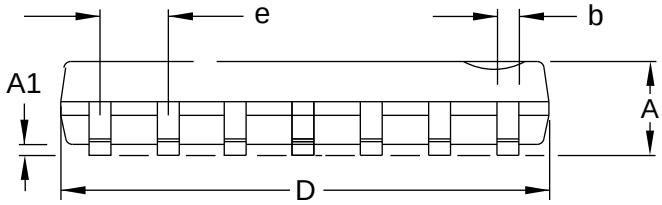
25.3 14S1



Top View



End View



Side View

COMMON DIMENSIONS
(Unit of Measure = mm/inches)

SYMBOL	MIN	NOM	MAX	NOTE
A	1.35/0.0532	–	1.75/0.0688	
A1	0.1/.0040	–	0.25/0.0098	
b	0.33/0.0130	–	0.5/0.0200 5	
D	8.55/0.3367	–	8.74/0.3444	2
E	3.8/0.1497	–	3.99/0.1574	3
H	5.8/0.2284	–	6.19/0.2440	
L	0.41/0.0160	–	1.27/0.0500	4
e	1.27/0.050 BSC			

Notes:
1. This drawing is for general information only; refer to JEDEC Drawing MS-012, Variation AB for additional information.
2. Dimension D does not include mold Flash, protrusions or gate burrs. Mold Flash, protrusion and gate burrs shall not exceed 0.15 mm (0.006") per side.
3. Dimension E does not include inter-lead Flash or protrusion. Inter-lead flash and protrusions shall not exceed 0.25 mm (0.010") per side.
4. L is the length of the terminal for soldering to a substrate.
5. The lead width B, as measured 0.36 mm (0.014") or greater above the seating plane, shall not exceed a maximum value of 0.61 mm (0.024") per side.

2/5/02

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**14S1**, 14-lead, 0.150" Wide Body, Plastic Gull Wing Small Outline Package (SOIC)	14S1	A

26. Errata

The revision letters in this section refer to the revision of the corresponding ATtiny24/44/84 device.

26.1 ATtiny24

26.1.1 Rev. D – E

No known errata.

26.1.2 Rev. C

- **Reading EEPROM when system clock frequency is below 900 kHz may not work**

1. **Reading EEPROM when system clock frequency is below 900 kHz may not work**
   Reading data from the EEPROM at system clock frequency below 900 kHz may result in wrong data read.

   Problem Fix/Work around
   Avoid using the EEPROM at clock frequency below 900 kHz.

26.1.3 Rev. B

- **EEPROM read from application code does not work in Lock Bit Mode 3**
- **Reading EEPROM when system clock frequency is below 900 kHz may not work**

1. **EEPROM read from application code does not work in Lock Bit Mode 3**
   When the Memory Lock Bits LB2 and LB1 are programmed to mode 3, EEPROM read does not work from the application code.

   Problem Fix/Work around
   Do not set Lock Bit Protection Mode 3 when the application code needs to read from EEPROM.

2. **Reading EEPROM when system clock frequency is below 900 kHz may not work**
   Reading data from the EEPROM at system clock frequency below 900 kHz may result in wrong data read.

   Problem Fix/Work around
   Avoid using the EEPROM at clock frequency below 900 kHz.

26.1.4 Rev. A

Not sampled.

26.2 ATtiny44

26.2.1 Rev. B – D

No known errata.

26.2.2 Rev. A

- **Reading EEPROM when system clock frequency is below 900 kHz may not work**

1. **Reading EEPROM when system clock frequency is below 900 kHz may not work**
   Reading data from the EEPROM at system clock frequency below 900 kHz may result in wrong data read.

   Problem Fix/Work around
   Avoid using the EEPROM at clock frequency below 900 kHz.

26.3 ATtiny84

26.3.1 Rev. A – B

No known errata.

27. Datasheet Revision History

27.1 Rev H. 10/09

1. Updated document template. Re-arranged some sections.
2. Updated “Low-Frequency Crystal Oscillator” with the Table 6-8 on page 28
3. Updated Tables:
   – “Active Clock Domains and Wake-up Sources in Different Sleep Modes” on page 33
   – “DC Characteristics” on page 174
   – “Register Summary” on page 213
4. Updated Register Description:
   – “ADMUX – ADC Multiplexer Selection Register” on page 145
5. Signature Imprint Reading Instructions updated in “Reading Device Signature Imprint Table from Firmware” on page 156.
6. Updated Section:
   – Step 1. on page 164
7. Added Table:
   – “Analog Comparator Characteristics” on page 179
8. Updated Figure:
   – “Active Supply Current vs. frequency (1 - 20 MHz)” on page 187
9. Updated Figure 21-30 on page 201 and Figure 21-33 on page 202 under “Pin Threshold and Hysteresis”.
10. Changed ATtiny24/44 device status to “Not Recommended for New Designs. Use: ATtiny24A/44A”.

27.2 Rev G. 01/08

1. Updated sections:
   – “Features” on page 1
   – “RESET” on page 3
   – “Overview” on page 4
   – “About” on page 6
   – “SPH and SPL — Stack Pointer Register” on page 11
   – “Atomic Byte Programming” on page 17
   – “Write” on page 17
   – “Clock Sources” on page 25
   – “Default Clock Source” on page 30
   – “Sleep Modes” on page 33
   – “Software BOD Disable” on page 34
   – “External Interrupts” on page 49
   – “USIBR – USI Data Buffer” on page 125
   – “USIDR – USI Data Register” on page 124
   – “DIDR0 – Digital Input Disable Register 0” on page 131
   – “Features” on page 132

- “Prescaling and Conversion Timing” on page 135
- “Temperature Measurement” on page 144
- “ADMUX – ADC Multiplexer Selection Register” on page 145
- “Limitations of debugWIRE” on page 152
- “Reading Lock, Fuse and Signature Data from Software” on page 155
- “Device Signature Imprint Table” on page 161
- “Enter High-voltage Serial Programming Mode” on page 168
- “Absolute Maximum Ratings*” on page 174
- “DC Characteristics” on page 174
- “Speed Grades” on page 175
- “Clock Characteristics” on page 176
- “Accuracy of Calibrated Internal RC Oscillator” on page 176
- “System and Reset Characteristics” on page 177
- “Supply Current of I/O Modules” on page 185
- “ATtiny24” on page 223
- “ATtiny44” on page 224
- “ATtiny84” on page 225

2. Updated bit definitions in sections:
   - “MCUCR – MCU Control Register” on page 36
   - “MCUCR – MCU Control Register” on page 51
   - “MCUCR – MCU Control Register” on page 67
   - “PINA – Port A Input Pins” on page 68
   - “SPMCSR – Store Program Memory Control and Status Register” on page 157
   - “Register Summary” on page 213
3. Updated Figures:
   - “Reset Logic” on page 39
   - “Watchdog Reset During Operation” on page 42
   - “Compare Match Output Unit, Schematic (non-PWM Mode)” on page 95
   - “Analog to Digital Converter Block Schematic” on page 133
   - “ADC Timing Diagram, Free Running Conversion” on page 137
   - “Analog Input Circuitry” on page 140
   - “High-voltage Serial Programming” on page 167
   - “Serial Programming Timing” on page 183
   - “High-voltage Serial Programming Timing” on page 184
   - “Active Supply Current vs. Low Frequency (0.1 - 1.0 MHz)” on page 186
   - “Active Supply Current vs. frequency (1 - 20 MHz)” on page 187
   - “Active Supply Current vs. VCC (Internal RC Oscillator, 8 MHz)” on page 187
   - “Active Supply Current vs. VCC (Internal RC Oscillator, 1 MHz)” on page 188
   - “Active Supply Current vs. VCC (Internal RC Oscillator, 128 kHz)” on page 188
   - “Idle Supply Current vs. Low Frequency (0.1 - 1.0 MHz)” on page 189
   - “Idle Supply Current vs. Frequency (1 - 20 MHz)” on page 189

- “Idle Supply Current vs. VCC (Internal RC Oscillator, 8 MHz)” on page 190
- “Idle Supply Current vs. VCC (Internal RC Oscillator, 1 MHz)” on page 190
- “Idle Supply Current vs. VCC (Internal RC Oscillator, 128 kHz)” on page 191
- “Power-down Supply Current vs. VCC (Watchdog Timer Disabled)” on page 191
- “Power-down Supply Current vs. VCC (Watchdog Timer Enabled)” on page 192
- “Reset Pin Input Hysteresis vs. VCC” on page 202
- “Reset Pin Input Hysteresis vs. VCC (Reset Pin Used as I/O)” on page 203
- “Watchdog Oscillator Frequency vs. VCC” on page 205
- “Watchdog Oscillator Frequency vs. Temperature” on page 205
- “Calibrated 8 MHz RC Oscillator Frequency vs. VCC” on page 206
- “Calibrated 8 MHz RC oscillator Frequency vs. Temperature” on page 206
- “ADC Current vs. VCC” on page 207
- “Programming Current vs. VCC (ATtiny24)” on page 209
- “Programming Current vs. VCC (ATtiny44)” on page 209
- “Programming Current vs. VCC (ATtiny84)” on page 210

4. Added Figures:
   - “Reset Pin Output Voltage vs. Sink Current (VCC = 3V)” on page 198
   - “Reset Pin Output Voltage vs. Sink Current (VCC = 5V)” on page 198
   - “Reset Pin Output Voltage vs. Source Current (VCC = 3V)” on page 199
   - “Reset Pin Output Voltage vs. Source Current (VCC = 5V)” on page 199
5. Updated Tables:
   - “Device Clocking Options Select” on page 25
   - “Start-up Times for the Crystal Oscillator Clock Selection” on page 29
   - “Start-up Times for the Internal Calibrated RC Oscillator Clock Selection” on page 27
   - “Start-up Times for the External Clock Selection” on page 26
   - “Start-up Times for the 128 kHz Internal Oscillator” on page 27
   - “Active Clock Domains and Wake-up Sources in Different Sleep Modes” on page 33
   - “Watchdog Timer Prescale Select” on page 47
   - “Reset and Interrupt Vectors” on page 48
   - “Overriding Signals for Alternate Functions in PA7..PA5” on page 63
   - “Overriding Signals for Alternate Functions in PA4..PA2” on page 64
   - “Overriding Signals for Alternate Functions in PA1..PA0” on page 64
   - “Port B Pins Alternate Functions” on page 65
   - “Overriding Signals for Alternate Functions in PB3..PB2” on page 66
   - “Overriding Signals for Alternate Functions in PB1..PB0” on page 67
   - “Waveform Generation Modes” on page 110
   - “ADC Conversion Time” on page 138
   - “Temperature vs. Sensor Output Voltage (Typical Case)” on page 144
   - “DC Characteristics. TA = -40°C to +85°C” on page 174
   - “Calibration Accuracy of Internal RC Oscillator” on page 176
   - “Reset, Brown-out, and Internal Voltage Characteristics” on page 177

- "VBOT vs. BODLEVEL Fuse Coding" on page 179
- "ADC Characteristics, Single Ended Channels. T = -40°C - 85°C" on page 180
- "ADC Characteristics, Differential Channels (Bipolar Mode), TA = -40°C to 85°C" on page 182
- "Serial Programming Characteristics, TA = -40°C to 85°C, VCC = 1.8 - 5.5V (Unless Otherwise Noted)" on page 183
- "High-voltage Serial Programming Characteristics TA = 25°C, VCC = 5V (Unless otherwise noted)" on page 184

6. Updated code examples in sections:
   - "Write" on page 17
   - "SPI Master Operation Example" on page 119
7. Updated "Ordering Information" in:
   - "ATtiny84" on page 219

27.3 Rev F. 02/07

1. Updated Figure 1-1 on page 2, Figure 8-7 on page 43, Figure 20-6 on page 184.
2. Updated Table 9-1 on page 48, Table 10-7 on page 65, Table 11-2 on page 80, Table 11-3 on page 81, Table 11-5 on page 81, Table 11-6 on page 82, Table 11-7 on page 82, Table 11-8 on page 83, Table 20-11 on page 182, Table 20-13 on page 184.
3. Updated table references in "TCCR0A – Timer/Counter Control Register A" on page 80.
4. Updated Port B, Bit 0 functions in "Alternate Functions of Port B" on page 65.
5. Updated WDTCR bit name to WDTCSR in assembly code examples.
6. Updated bit5 name in "TIFR1 – Timer/Counter Interrupt Flag Register 1" on page 114.
7. Updated bit5 in "TIFR1 – Timer/Counter Interrupt Flag Register 1" on page 114.
8. Updated "SPI Master Operation Example" on page 119.
9. Updated step 5 in "Enter High-voltage Serial Programming Mode" on page 168.

27.4 Rev E. 09/06

1. All characterization data moved to "Electrical Characteristics" on page 174.
2. All Register Descriptions gathered up in separate sections at the end of each chapter.
3. Updated "System Control and Reset" on page 39.
4. Updated Table 11-3 on page 81, Table 11-6 on page 82, Table 11-8 on page 83, Table 12-3 on page 109 and Table 12-5 on page 110.
5. Updated "Fast PWM Mode" on page 97.
6. Updated Figure 12-7 on page 98 and Figure 16-1 on page 133.
7. Updated "Analog Comparator Multiplexed Input" on page 129.
8. Added note in Table 19-12 on page 165.
9. Updated "Electrical Characteristics" on page 174.
10. Updated "Typical Characteristics" on page 185.

27.5 Rev D. 08/06

1. Updated “Calibrated Internal 8 MHz Oscillator” on page 26.
2. Updated “OSCCAL – Oscillator Calibration Register” on page 30.
3. Added Table 20-2 on page 176.
4. Updated code examples in “SPI Master Operation Example” on page 119.
5. Updated code examples in “SPI Slave Operation Example” on page 121.
6. Updated “Signature Bytes” on page 162.

27.6 Rev C. 07/06

1. Updated Features in “USI – Universal Serial Interface” on page 117.
2. Added “Clock speed considerations” on page 123.
3. Updated Bit description in “ADMUX – ADC Multiplexer Selection Register” on page 145.
4. Added note to Table 18-1 on page 157.

27.7 Rev B. 05/06

1. Updated “Default Clock Source” on page 30
2. Updated “Power Reduction Register” on page 35.
3. Updated Table 20-4 on page 177, Table 9-4 on page 42, Table 16-3 on page 145, Table 19-5 on page 161, Table 19-12 on page 165, Table 19-16 on page 171, Table 20-11 on page 182.
4. Updated Features in “Analog to Digital Converter” on page 132.
5. Updated Operation in “Analog to Digital Converter” on page 132.
6. Updated “Temperature Measurement” on page 144.
7. Updated DC Characteristics in “Electrical Characteristics” on page 174.
8. Updated “Typical Characteristics” on page 185.
9. Updated “Errata” on page 223.

27.8 Rev A. 12/05

Initial revision.

Table of Contents

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
avr@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature