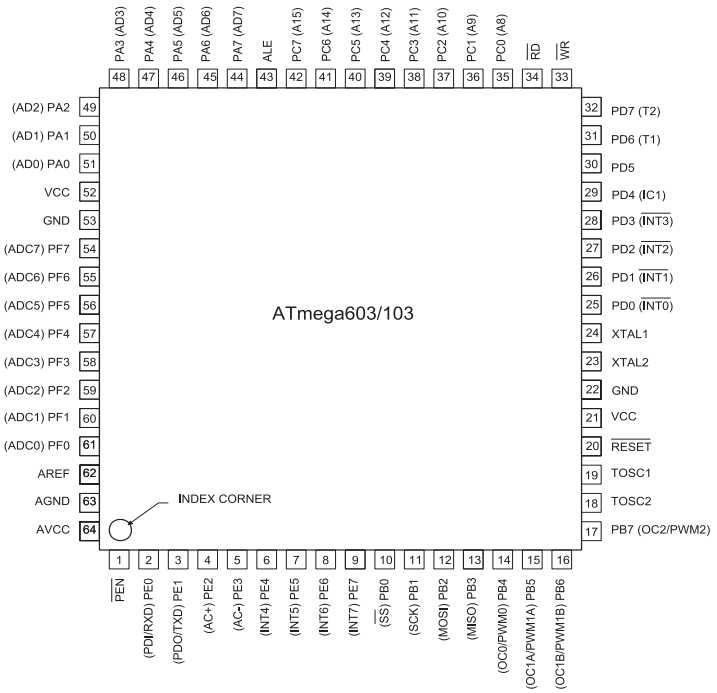
Features

- Utilizes the AVR® Enhanced RISC Architecture
- 121 Powerful Instructions - Most Single Clock Cycle Execution
- 128K bytes of In-System Reprogrammable Flash ATmega103/L
  64K bytes of In-System Reprogrammable Flash ATmega603/L
  - SPI Interface for In System Programming
  - Endurance: 1,000 Write/Erase Cycles
- 4K bytes EEPROM ATmega103/L
  2K bytes of EEPROM ATmega603/L
  - Endurance: 100,000 Write/Erase Cycles
- 4K bytes Internal SRAM
- 32 x 8 General Purpose Working Registers + Peripheral Control Registers
- 32 Programmable I/O Lines, 8 Output Lines, 8 Input Lines
- Programmable Serial UART + SPI Serial Interface
- V_{CC} Supply
  - 2.7 - 6.0V ATmega603L/ATmega103L
  - 4.0 - 6.0V ATmega603/ATmega103
- Fully Static Operation
  - 0 - 6 MHz ATmega603/ATmega103
  - 0 - 4 MHz ATmega603L/ATmega103L
- Up to 6 MIPS Throughput at 6 MHz
- RTC with Separate Oscillator
- Two 8-Bit Timer/Counters with Separate Prescaler and PWM
- One 16-Bit Timer/Counter with Separate Prescaler, Compare, Capture Modes and Dual 8-, 9- or 10-Bit PWM
- Programmable Watchdog Timer with On-Chip Oscillator
- On-Chip Analog Comparator
- 8-Channel, 10-Bit ADC
- Low Power Idle, Power Save and Power Down Modes
- Software Selectable Clock Frequency
- Programming Lock for Software Security

Pin Configuration

TQFP



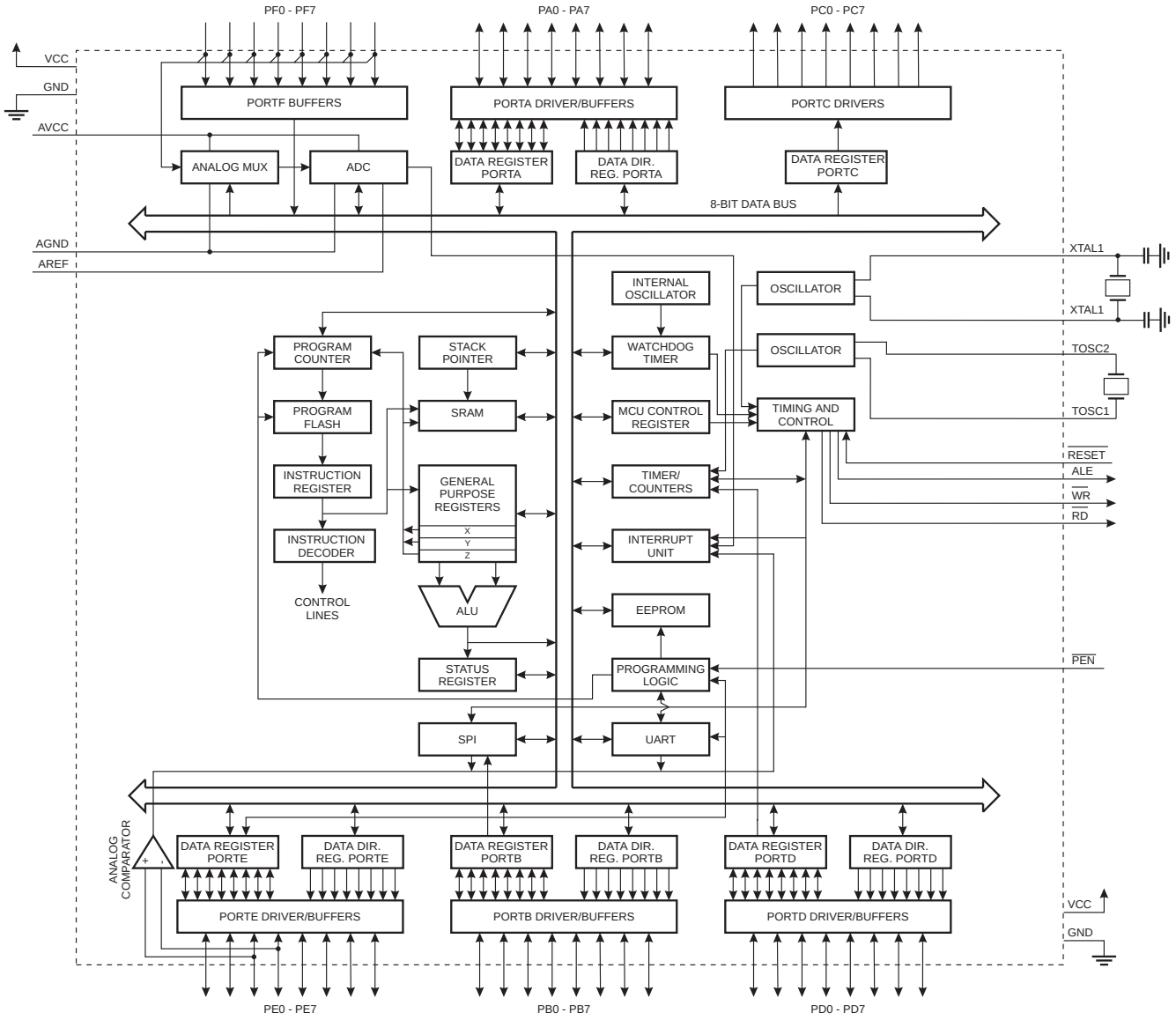
8-Bit AVR® Microcontroller with 64K/128K Bytes In-System Programmable Flash

ATmega603
ATmega603L
ATmega103
ATmega103L
Preliminary

Rev. 0945A–05/98

Block Diagram

Figure 1. The ATmega603/103 Block Diagram



Description

The ATmega603/103 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 ATmega603/103 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

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

The ATmega603/103 provides the following features: 64K/128K bytes of In-system Programmable Flash, 2K/4K bytes EEPROM, 4K bytes SRAM, 32 general purpose I/O lines, 8 Input lines, 8 Output lines, 32 general purpose working registers, 4 flexible timer/counters with compare modes and PWM, UART, programmable Watchdog Timer with internal oscillator, an SPI serial port and three software selectable power saving modes. The Idle Mode stops the CPU while allowing the SRAM, timer/counters, SPI port and interrupt system to continue functioning. The Power Down mode saves the register contents but freezes the oscillator, disabling all other chip functions until the next

interrupt or hardware reset. In Power Save mode, the timer oscillator continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping.

The device is manufactured using Atmel's high-density non-volatile memory technology. The on-chip ISP Flash allows the program memory to be reprogrammed in-system through a serial interface or by a conventional nonvolatile memory programmer. By combining an 8-bit RISC CPU with a large array of ISP Flash on a monolithic chip, the Atmel ATmega603/103 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega603/103 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Comparison Between ATmega 603 and ATmega 103

The ATmega603 has 64K bytes of In-System Programmable Flash, 2K bytes of EEPROM, and 4K bytes of internal SRAM. The ATmega603 does not have the ELPM instruction.

The ATmega103 has 128K bytes of In-System Programmable Flash, 4K bytes of EEPROM, and 4K bytes of internal SRAM. The ATmega103 has the ELPM instruction, necessary to reach the upper half of the Flash memory for constant table lookup.

Table 1 summarizes the different memory sizes for the two devices.

Table 1. Memory Size Summary

Part	Flash	EEPROM	SRAM
ATmega603	64K bytes	2K bytes	4K bytes
ATmega103	128K bytes	4K bytes	4K bytes

Pin Descriptions

VCC

Supply voltage

GND

Ground

Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers can sink 20 mA and can drive LED displays directly. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

Port A serves as Multiplexed Address/Data bus when using external SRAM.

Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O pins with internal pull-up resistors. The Port B output buffers can sink 20 mA. As inputs, Port B pins that are externally pulled low, will source current if the pull-up resistors are activated.

Port B also serves the functions of various special features.

Port C (PC7..PC0)

Port C is an 8-bit Output port. The Port C output buffers can sink 20 mA.

Port C also serves as Address output when using external SRAM.

Port D (PD7..PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors. The Port D output buffers can sink 20 mA. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated.

Port D also serves the functions of various special features.

Port E (PE7..PE0)

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors. The Port E output buffers can sink 20 mA. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated.

Port E also serves the functions of various special features.

Port F (PF7..PF0)

Port F is an 8-bit Input port. Port F also serves as the analog inputs for the ADC.

$\overline{\text{RESET}}$

input. A low on this pin for two machine cycles while the oscillator is running resets the device.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier

TOSC1

Input to the inverting Timer/Counter oscillator amplifier

TOSC2

Output from the inverting Timer/Counter oscillator amplifier

$\overline{\text{WR}}$

External SRAM Write Strobe.

$\overline{\text{RD}}$

External SRAM Read Strobe.

ALE

ALE is the Address Latch Enable used when the External Memory is enabled. The ALE strobe is used to latch the low-order address (8 bits) into an address latch during the first access cycle, and the AD0-7 pins are used for data during the second access cycle.

AVCC
This is the supply voltage to the A/D Converter. It should be externally connected to V_{CC} via a low-pass filter. See page 52 for details on operation of the ADC.

AREF
This is the analog reference input for the ADC converter. For ADC operations, a voltage in the range AGND to AVCC must be applied to this pin.

AGND
If the board has a separate analog ground plane, this pin should be connected to this ground plane. Otherwise, connect to GND.

$\overline{\text{PEN}}$
This is a programming enable pin for the low-voltage serial programming mode. By holding this pin low during a power-on reset, the device will enter the serial programming mode.

Crystal Oscillator

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 2. Either a quartz crystal or a ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 3. For the Timer Oscillator pins, OSC1 and OSC2, the crystal is connected directly between the pins. No external capacitors are needed. The oscillator is optimized for use with a 32,768Hz watch crystal. An external clock signal applied to this pin goes through the same amplifier having a bandwidth of 256kHz. The external clock signal should therefore be in the interval 0Hz - 256kHz.

Figure 2. Oscillator Connections

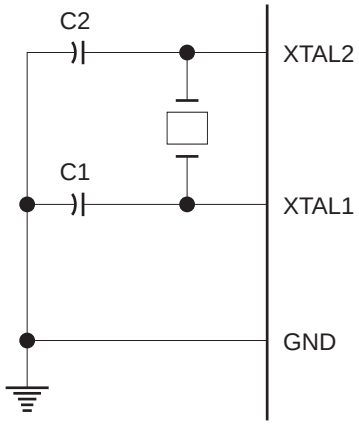
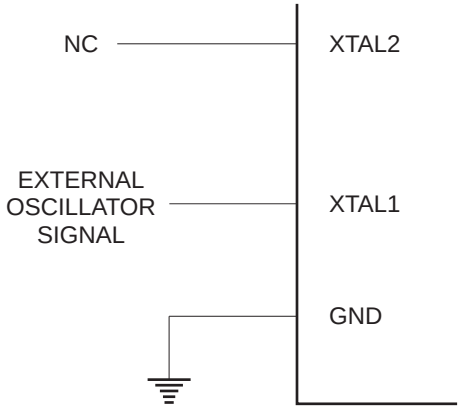


Figure 3. External Clock Drive Configuration



ATmega603/103 Architectural Overview

The fast-access register file contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This means that during one single clock cycle, one ALU (Arithmetic Logic Unit) operation is executed. 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.

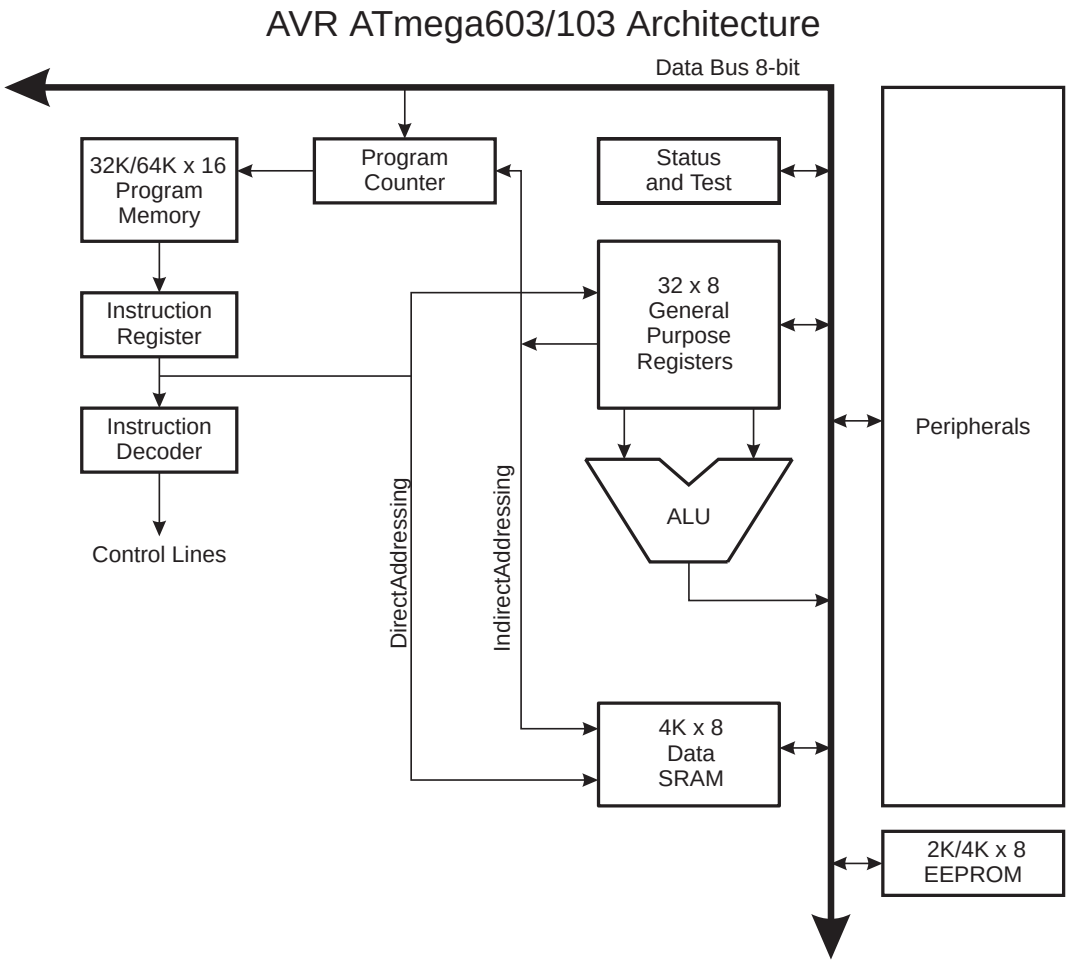
Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing - enabling efficient address calculations. One of the three address pointers is also used as the address pointer for the constant table look up function. These added function registers are the 16-bit X-register, Y-register and Z-register.

The ALU supports arithmetic and logic functions between registers or between a constant and a register. Single register operations are also executed in the ALU. Figure 4 shows the ATmega603/103 AVR Enhanced RISC microcontroller architecture.

In addition to the register operation, the conventional memory addressing modes can be used on the register file as well. This is enabled by the fact that the register file is assigned the 32 lowermost Data Space addresses, allowing them to be accessed as though they were ordinary memory locations.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, Timer/Counters, A/D-converters, 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, $20 - $5F.

Figure 4. The ATmega603/103 AVR Enhanced RISC Architecture



The AVR uses a Harvard architecture concept - with separate memories and buses for program and data. The program memory is 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 programmable Flash memory. With a few exceptions, AVR instructions have a single 16-bit word format, meaning that every program memory address contains a single 16-bit instruction.

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 16-bit stack pointer SP is read/write accessible in the I/O space.

The 4000 bytes data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.

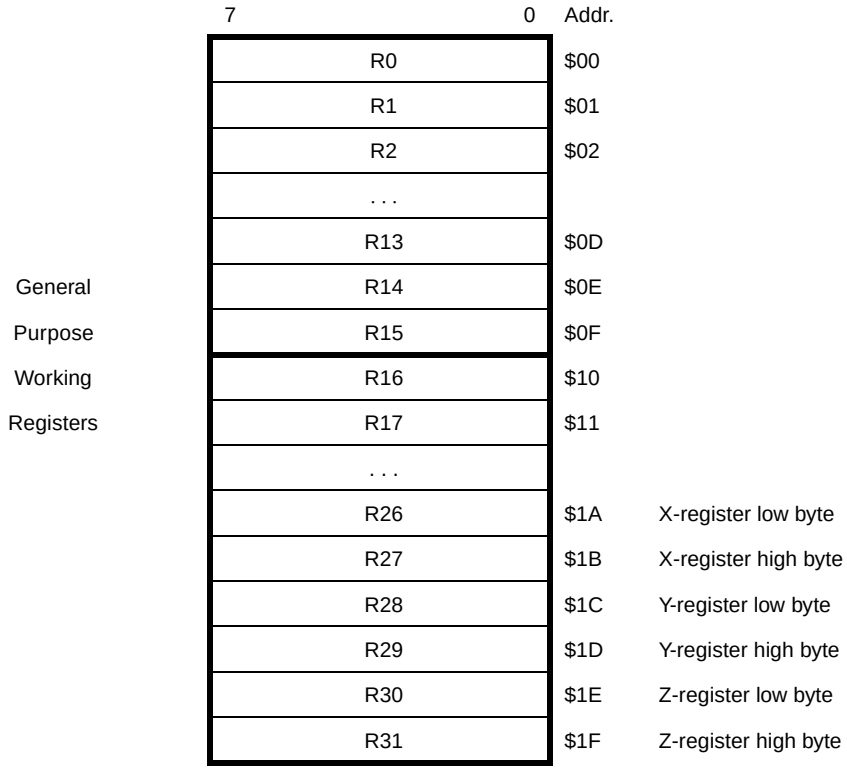
A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All the different interrupts have a separate interrupt vector in the interrupt vector table at the beginning of the program memory. The different interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

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

The General Purpose Register File

Figure 5 shows the structure of the 32 general purpose working registers in the CPU.

Figure 5. AVR CPU general purpose working registers



All the register operating instructions in the instruction set have direct and single cycle access to all registers. The only exception is the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI and ORI between a constant and a register and the LDI instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file - R16..R31. The general SBC, SUB, CP, AND and OR and all other operations between two registers or on a single register apply to the entire register file.

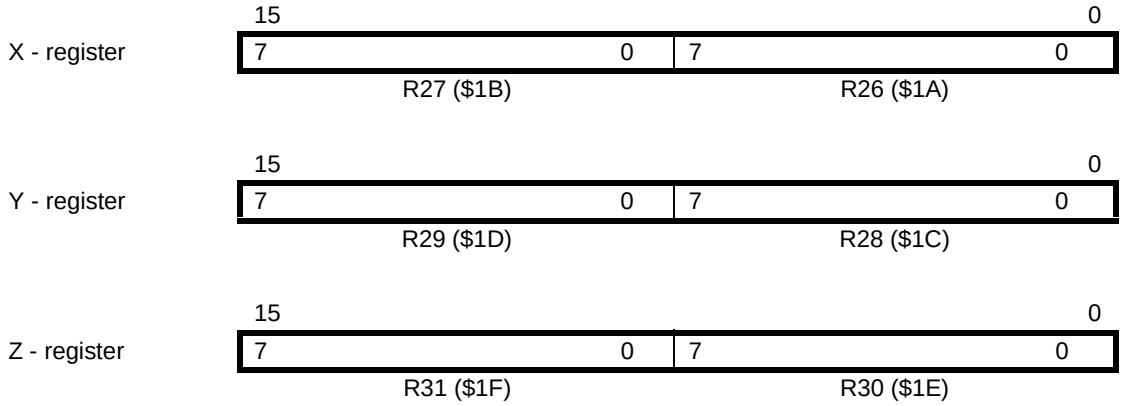
As shown in Figure 5, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X,Y and Z registers can be set to index any register in the file.

The 4K bytes of SRAM available for general data are implemented as addresses $0060 to $0FFF.

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

The registers R26..R31 have some added functions to their general purpose usage. These registers are address pointers for indirect addressing of the SRAM. The three indirect address registers X, Y and Z are defined as:

Figure 6. The X, Y and Z Registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment and decrement (see the descriptions for the different instructions).

The 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, ALU operations between registers in the register file are executed. The ALU operations are divided into three main categories - arithmetic, logical and bit-functions.

The ISP Flash Program Memory

The ATmega603/103 contains 128K bytes on-chip In-system Programmable Flash memory for program storage. Since all instructions are single or double 16-bit words, the Flash is organized as 64K x 16. The Flash memory has an endurance of at least 1000 write/erase cycles.

Constant tables can be allocated in the entire program memory space (see the LPM - Load Program Memory and ELPM Extended Load Program Memory instruction descriptions).

Memory Configurations

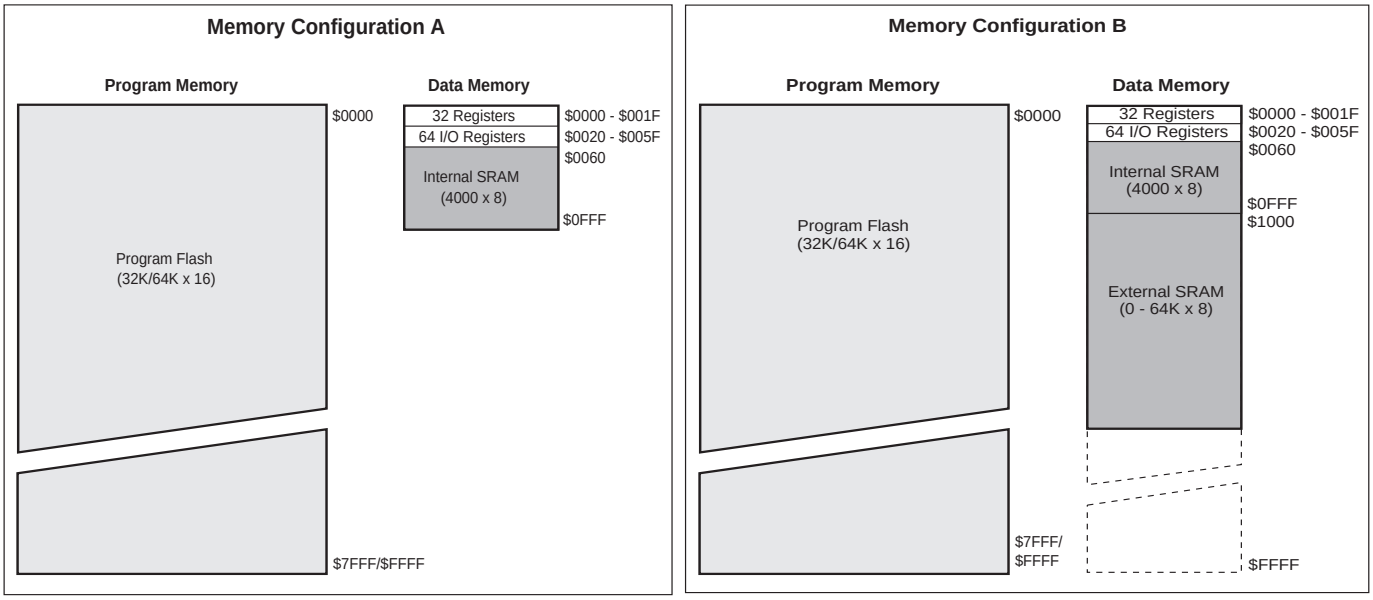
The ATmega603/103 supports two different memory configurations as listed in the following table:

Table 2. Memory Configurations

Configuration	Internal SRAM Data Memory	External SRAM Data Memory
A	4000	None
B	4000	up to 64K (1)

Note: 1. When using 64K of External SRAM, 60K will be available.

Figure 7. Memory Configurations



The 4096 first Data Memory locations address both the Register file, the I/O Memory and the internal data SRAM. The first 96 locations address the register file and I/O memory, and the next 4000 locations address the internal data SRAM.

An optional external data SRAM can be used with the ATmega603/103. This SRAM will occupy an area in the remaining address locations in the 64K address space. This area starts at the address following the internal SRAM. If a 64K external SRAM is used, 4K of the external memory is lost as the addresses are occupied by internal memory.

When the addresses accessing the SRAM memory space exceeds the internal data memory locations, the external data SRAM is accessed using the same instructions as for the internal data memory access. When the internal data memories are accessed, the read and write strobe pins ($\overline{RD}$ and $\overline{WR}$) are inactive during the whole access cycle. External SRAM operation is enabled by setting the SRE bit in the MCUCR register.

Accessing external SRAM takes one additional clock cycle per byte compared to access of the internal SRAM. This means that the commands LD, ST, LDS, STS, PUSH and POP take one additional clock cycle. If the stack is placed in external SRAM, interrupts, subroutine calls and returns take two clock cycles extra because the two-byte program counter is pushed and popped. When external SRAM interface is used with wait state, two additional clock cycles is used per byte. This has the following effect: Data transfer

instructions take two extra clock cycles, whereas interrupt, subroutine calls and returns will need four clock cycles more than specified in the instruction set manual.

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 Indirect with Displacement mode features a 63 address locations reach 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 and incremented.

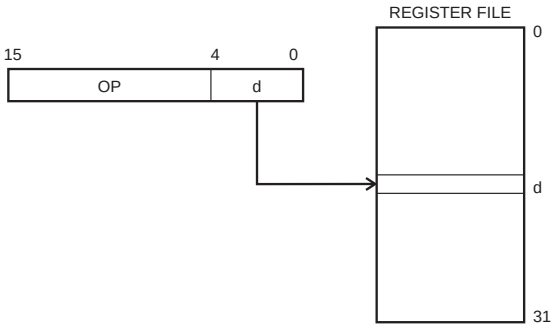
The entire data address space including the 32 general purpose working registers and the 64 I/O registers are all accessible through all these addressing modes. See the next section for a detailed description of the different addressing modes.

The Program and Data Addressing Modes

The ATmega603/103 AVR Enhanced RISC microcontroller supports powerful and efficient addressing modes for access to the program memory (Flash) and data memory (SRAM, Register File and I/O Memory). This section describes the different addressing modes supported by the AVR architecture. In the figures, OP means the operation code part of the instruction word. To simplify, not all figures show the exact location of the addressing bits.

Register Direct, Single Register Rd

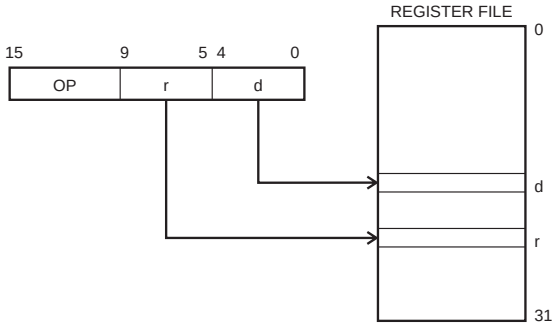
Figure 8. Direct Single Register Addressing



The operand is contained in register d (Rd).

Register Direct, Two Registers Rd and Rr

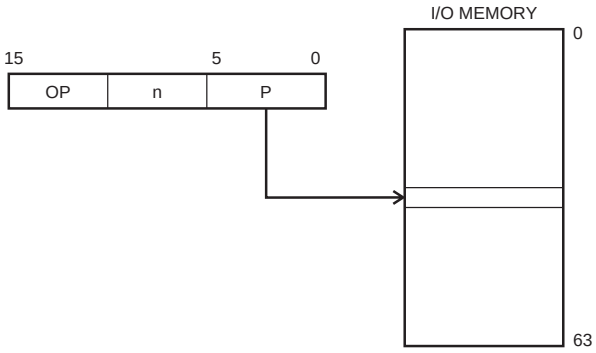
Figure 9. Direct Register Addressing, Two Registers



Operands are contained in register r (Rr) and d (Rd). The result is stored in register d (Rd).

I/O Direct

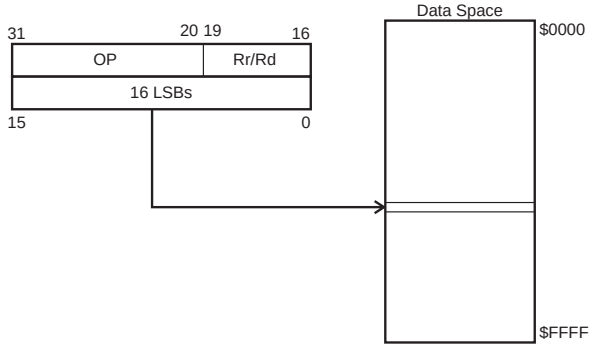
Figure 10. I/O Direct Addressing



Operand address is contained in 6 bits of the instruction word. n is the destination or source register address.

Data Direct

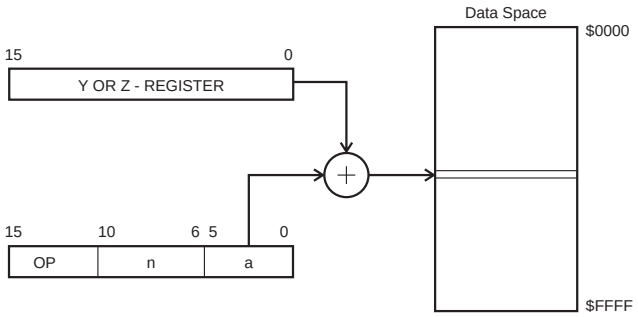
Figure 11. Direct Data Addressing



A 16-bit Data Address is contained in the 16 LSBs of a two-word instruction. Rd/Rr specify the destination or source register.

Data Indirect with Displacement

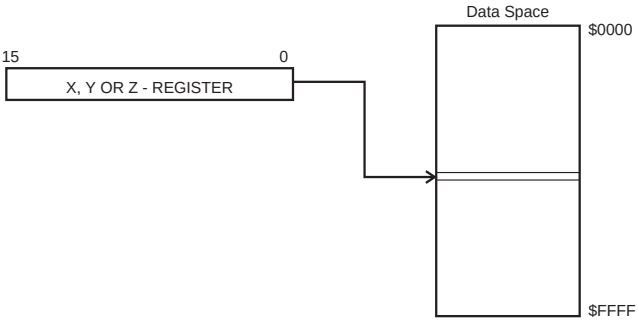
Figure 12. Data Indirect with Displacement



Operand address is the result of the Y or Z-register contents added to the address contained in 6 bits of the instruction word.

Data Indirect

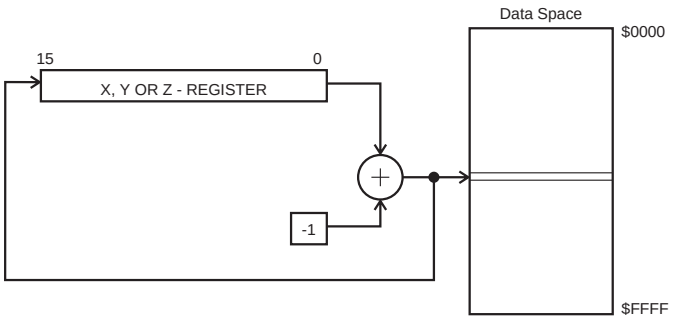
Figure 13. Data Indirect Addressing



Operand address is the contents of the X, Y or the Z-register.

Data Indirect With Pre-Decrement

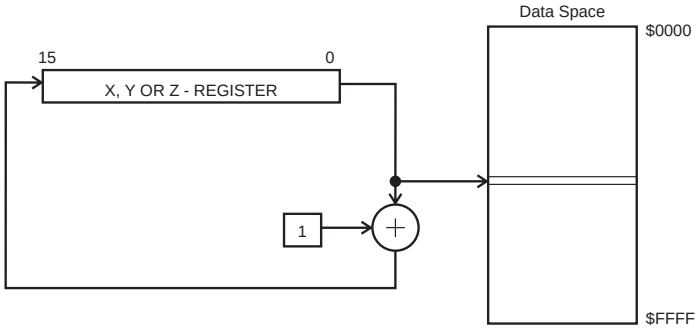
Figure 14. Data Indirect Addressing With Pre-Decrement



The X, Y or the Z-register is decremented before the operation. Operand address is the decremented contents of the X, Y or the Z-register.

Data Indirect With Post-Increment

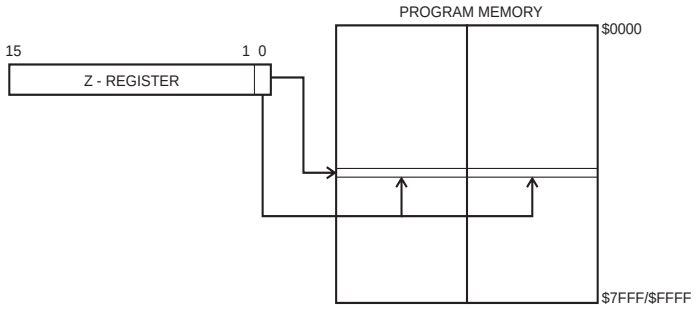
Figure 15. Data Indirect Addressing With Post-Increment



The X, Y or the Z-register is incremented after the operation. Operand address is the content of the X, Y or the Z-register prior to incrementing.

Constant Addressing Using the LPM and ELPM Instructions

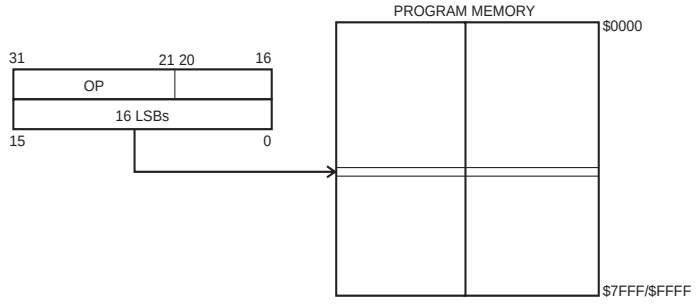
Figure 16. Code Memory Constant Addressing



Constant byte address is specified by the Z-register contents. The 15 MSBs select word address (0 - 32K) and LSB, select low byte if cleared (LSB = 0) or high byte if set (LSB = 1). If ELPM is used, LSB of the RAM Page Z register - RAMPZ is used to select low or high memory page (RAMPZ0 = 0: Low Page, RAMPZ0 = 1: High Page). ELPM does not apply to the ATmega603.

Direct Program Address, JMP and CALL

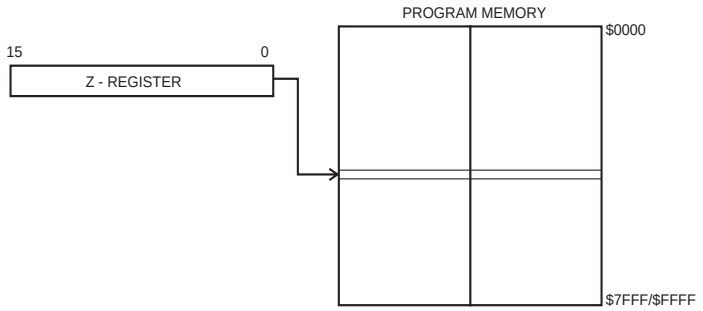
Figure 17. Direct Program Memory Addressing



Program execution continues at the address immediate in the instruction words.

Indirect Program Addressing, IJMP and ICALL

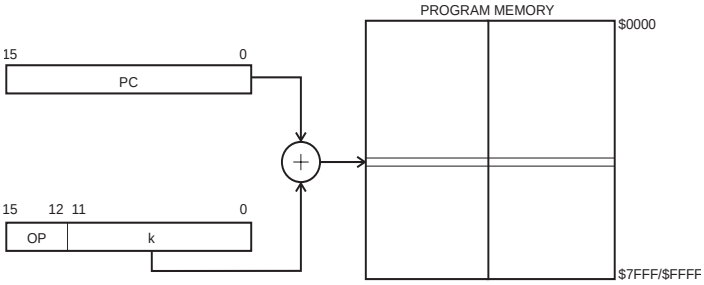
Figure 18. Indirect Program Memory Addressing



Program execution continues at address contained by the Z-register (i.e. the PC is loaded with the contents of the Z-register).

Relative Program Addressing, RJMP and RCALL

Figure 19. Relative Program Memory Addressing



Program execution continues at address PC + k + 1. The relative address k is -2048 to 2047.

The EEPROM Data Memory

The EEPROM memory 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 on page 40 specifying the EEPROM address register, the EEPROM data register, and the EEPROM control register.

Memory Access Times and Instruction Execution Timing

This section describes the general access timing concepts for instruction execution and internal memory access.

The AVR CPU is driven by the System Clock Ø, directly generated from the external clock crystal for the chip. No internal clock division is used.

Figure 20 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 20. The Parallel Instruction Fetches and Instruction Executions

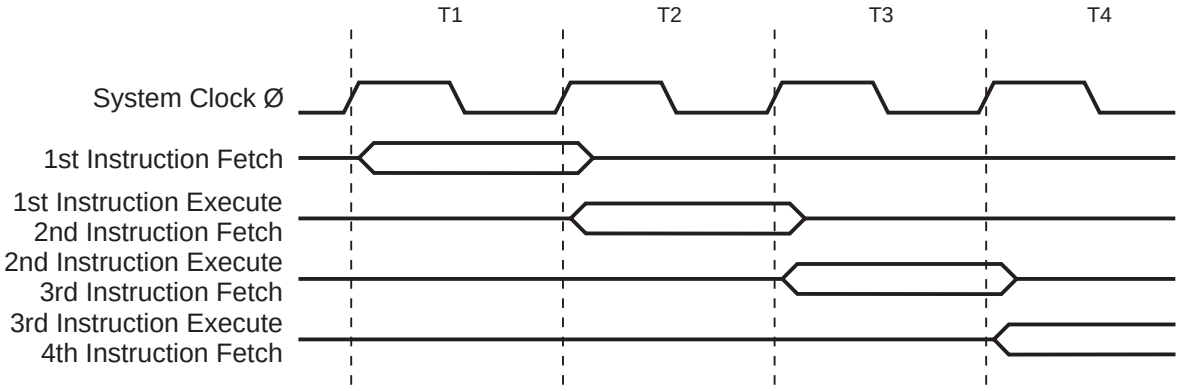
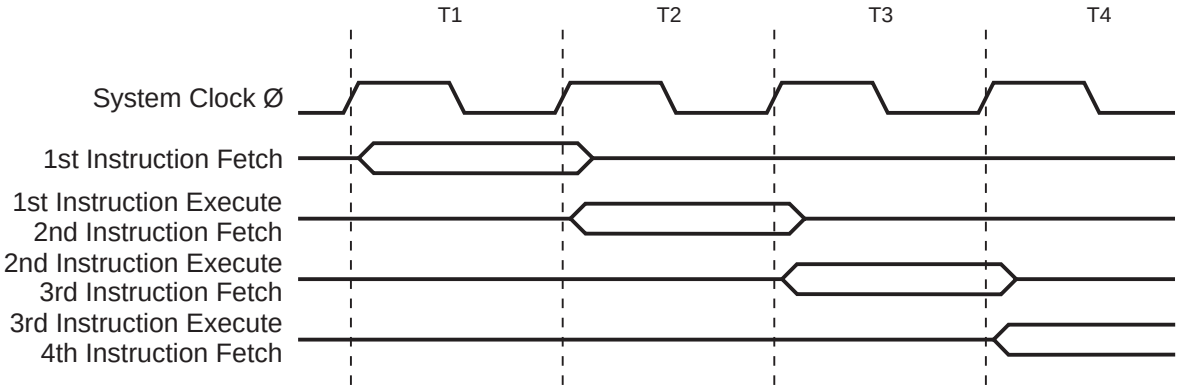


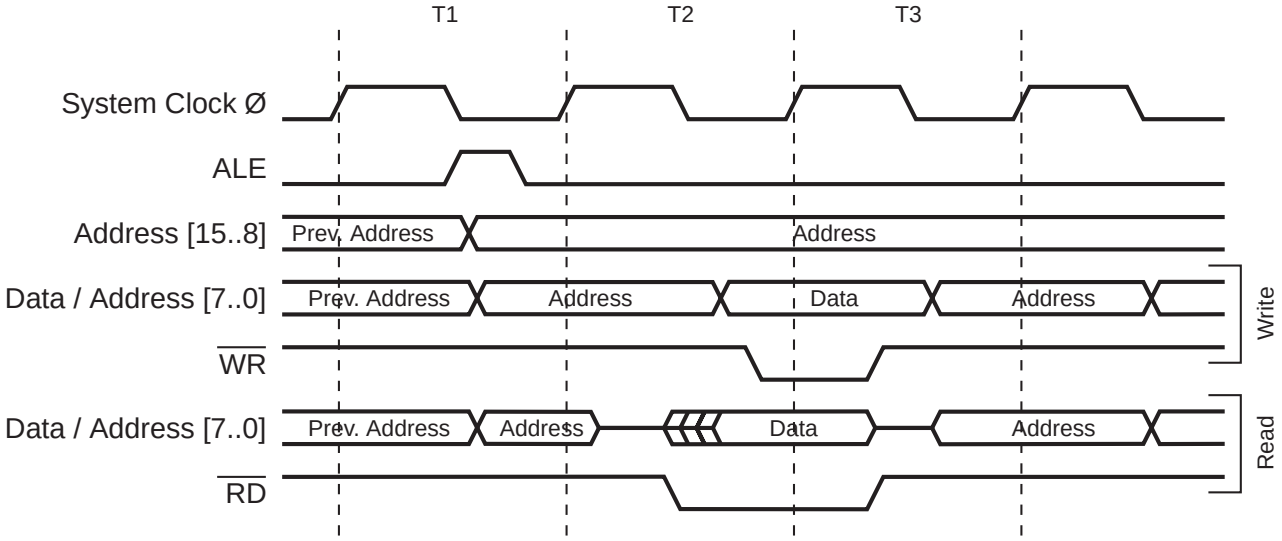
Figure 21 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 21. Single Cycle ALU Operation



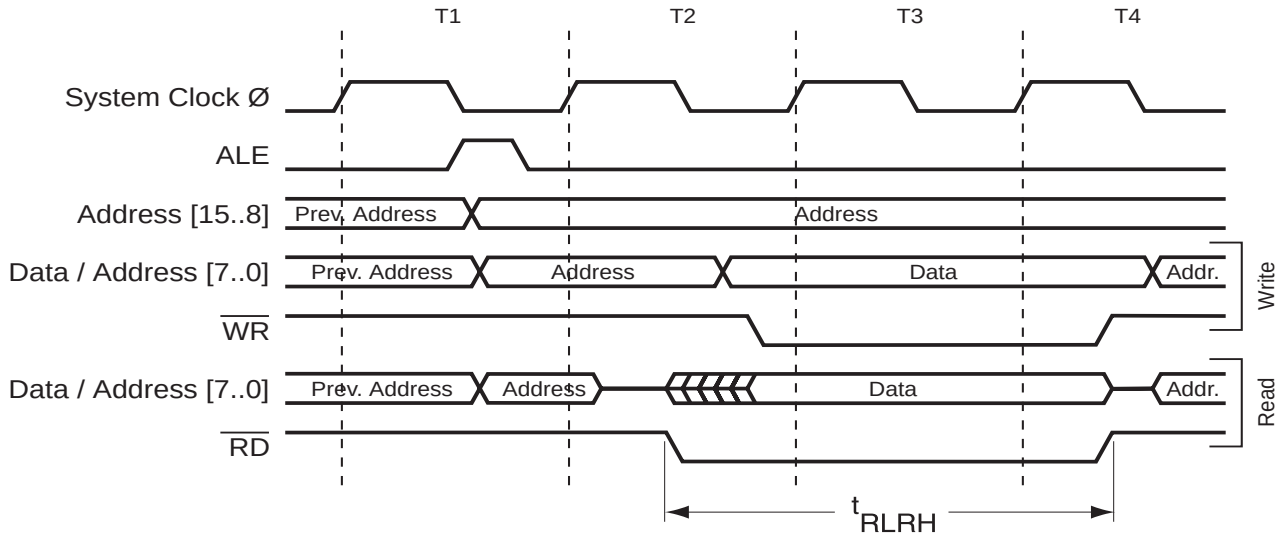
The internal data SRAM access is performed in two System Clock cycles as described in Figure 22.

Figure 22. External Data SRAM Memory Cycles without Wait State



The external data SRAM memory access cycle with the Wait State bit enabled (Wait State active) is shown in Figure 23.

Figure 23. External Data SRAM Memory Cycles with Wait State



I/O Memory

The I/O space definition of the ATmega603/103 is shown in the following table:

Table 3. ATmega603/103 I/O Space

I/O Address (SRAM Address)	Name	Function
$3F ($5F)	SREG	Status REGister
$3E ($5E)	SPH	Stack Pointer High
$3D ($5D)	SPL	Stack Pointer Low
$3C ($5C)	XDIV	XTAL Divide Control Register
$3B ($5B)	RAMPZ	RAM Page Z Select Register
$3A ($5A)	EICR	External Interrupt Control Register
$39 ($59)	EIMSK	External Interrupt MaSK register
$38 ($58)	EIFR	External Interrupt Flag Register
$37 ($57)	TIMSK	Timer/Counter Interrupt MaSK register
$36 ($56)	TIFR	Timer/Counter Interrupt Flag register
$35 ($55)	MCUCR	MCU General Control Register
$34 ($54)	MCUSR	MCU Status Register
$33 ($53)	TCCR0	Timer/Counter0 Control Register
$32 ($52)	TCNT0	Timer/Counter0 (8-bit)
$31 ($51)	OCR0	Timer/Counter0 Output Compare Register
$30 ($50)	ASSR	Asynchronous Mode Status Register
$2F ($4F)	TCCR1A	Timer/Counter1 Control Register A
$2E ($4E)	TCCR1B	Timer/Counter1 Control Register B
$2D ($4D)	TCNT1H	Timer/Counter1 High Byte
$2C ($4C)	TCNT1L	Timer/Counter1 Low Byte
$2B ($4B)	OCR1AH	Timer/Counter1 Output Compare Register A High Byte
$2A ($4A)	OCR1AL	Timer/Counter1 Output Compare Register A Low Byte
$29 ($49)	OCR1BH	Timer/Counter1 Output Compare Register B High Byte
$28 ($48)	OCR1BL	Timer/Counter1 Output Compare Register B Low Byte
$27 ($47)	ICR1H	Timer/Counter1 Input Capture Register High Byte
$26 ($46)	ICR1L	Timer/Counter1 Input Capture Register Low Byte
$25 ($45)	TCCR2	Timer/Counter2 Control Register
$24 ($44)	TCNT2	Timer/Counter2 (8-bit)
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register
$21 ($41)	WDTCR	Watchdog Timer Control Register
$1F ($3F)	EEARH	EEPROM Address Register High
$1E ($3E)	EEARL	EERPOM Address Register Low
$1D ($3D)	EEDR	EEPROM Data Register
$1C ($3C)	EECR	EEPROM Control Register

Table 3. ATmega603/103 I/O Space (Continued)

I/O Address (SRAM Address)	Name	Function
$1B ($3B)	PORTA	Data Register, Port A
$1A ($3A)	DDRA	Data Direction Register, Port A
$19 ($39)	PINA	Input Pins, Port A
$18 ($38)	PORTB	Data Register, Port B
$17 ($37)	DDRB	Data Direction Register, Port B
$16 ($36)	PINB	Input Pins, Port B
$15 ($35)	PORTC	Data Register, Port C
$12 ($32)	PORTD	Data Register, Port D
$11 ($31)	DDRD	Data Direction Register, Port D
$10 ($30)	PIND	Input Pins, Port D
$0F ($2F)	SPDR	SPI I/O Data Register
$0E ($2E)	SPSR	SPI Status Register
$0D ($2D)	SPCR	SPI Control Register
$0C ($2C)	UDR	UART I/O Data Register
$0B ($2B)	USR	UART Status Register
$0A ($2A)	UCR	UART Control Register
$09 ($29)	UBRR	UART Baud Rate Register
$08 ($28)	ACSR	Analog Comparator Control and Status Register
$07 ($27)	ADMUX	ADC Multiplexer Select Register
$06 ($26)	ADCSR	ADC Control and Status Register
$05 ($25)	ADCH	ADC Data Register High
$04 ($24)	ADCL	ADC Data Register Low
$03 ($23)	PORTE	Data Register, Port E
$02 ($22)	DDRE	Data Direction Register, Port E
$01 ($21)	PINE	Input Pins, Port E
$00 ($20)	PINF	Input Pins, Port F

Note: Reserved and unused locations are not shown in the table

All the different ATmega603/103 I/Os and peripherals are placed in the I/O space. The different I/O locations are directly accessed by the IN and OUT instructions transferring data between the 32 general purpose working registers and the I/O space. When using IN and OUT, the I/O register address $00 - $3F are used. As the I/O register is also represented in the SRAM address space, they can also be addressed as ordinary SRAM locations within the address space $20 - $5F. The SRAM address is obtained by adding $20 to the direct I/O address. The SRAM address is given in parentheses after the I/O direct address throughout this document. I/O registers within the address range $00 ($20) - $1F ($3F) 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. Refer to the instruction set chapter for more details.

The different I/O and peripherals control registers are explained in the following sections.

The Status Register - SREG

The AVR status register - SREG - at I/O space location $3F ($5F) is defined as:

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	**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 (one) for the interrupts to be enabled. The individual interrupt enable control is then performed in the interrupt mask registers - GIMSK and TIMSK. If the global interrupt enable register is cleared (zero), none of the interrupts are enabled independent of the GIMSK and TIMSK values. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts.

- **Bit 6 - T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T bit as source and 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. 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 after the different arithmetic and logic operations. See the Instruction Set Description for detailed information.

- **Bit 1 - Z: Zero Flag**

The zero flag Z indicates a zero result after the different arithmetic and logic operations. 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.

The Stack Pointer - SP

The general AVR 16-bit Stack Pointer is effectively built up of two 8-bit registers in the I/O space locations $3E ($5E) and $3D ($5D). As the ATmega603/103 supports up to 64 kB memory, all 16-bits are used.

Bit	15	14	13	12	11	10	9	8	
$3E ($5E)	**SP15**	**SP14**	**SP13**	**SP12**	**SP11**	**SP10**	**SP9**	**SP8**	**SPH**
$3D ($5D)	**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	
	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	
	0	0	0	0	0	0	0	0	

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 is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when data is pushed onto the Stack with subroutine CALL and 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 IRET.

The RAM Page Z Select Register - RAMPZ

Bit	7	6	5	4	3	2	1	0	
$3B ($5B)	-	-	-	-	-	-	-	**RAMPZ0**	**RAMPZ**
Read/Write	R	R	R	R	R	R	R	R/W	
Initial value	0	0	0	0	0	0	0	0	

The RAMPZ register is normally used to select which 64K RAM Page is accessed by the Z pointer. As the ATmega603/103 does not support more than 64K of SRAM memory, this register is used only to select which page in the program memory is accessed when the ELPM instruction is used. The different settings of the RAMPZ0 bit have the following effects:

RAMPZ0 = 0: Program memory address $0000- $7FFF (lower 64K bytes) is accessed by ELPM

RAMPZ0 = 1: Program memory address $8000- $FFFF (higher 64K bytes) is accessed by ELPM

Note that LPM is not affected by the RAMPZ setting.

The ATmega603 does not contain the RAMPZ register, and it does not have the ELPM instruction. The ordinary LPM instruction can reach the entire program memory in the ATmega603.

MCU Control Register - MCUCR

The MCU Control Register contains control bits for general MCU functions.

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	**SRE**	**SRW**	**SE**	**SM1**	**SM0**	-	-	-	**MCUCR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - SRE: External SRAM Enable**

When the SRE bit is set (one), the external data SRAM is enabled, and the pin functions AD0-7 (Port A), A8-15 (Port C), and $\overline{WR}$/$\overline{RD}$ (Port D) are activated as the alternate pin functions. Then the SRE bit overrides any pin direction settings in the respective data direction registers. When the SRE bit is cleared (zero), the external data SRAM is disabled, and the normal pin and data direction settings are used.

- **Bit 6 - SRW: External SRAM Wait State**

When the SRW bit is set (one), a one cycle wait state is inserted in the external data SRAM access cycle. When the SRW bit is cleared (zero), the external data SRAM access is executed with a three-cycle scheme. See Figure 22 External Data SRAM Memory Cycles without Wait State and Figure 23: External Data SRAM Memory Cycles with Wait State.

- **Bit 5 - SE: Sleep Enable**

The SE bit must be set (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 programmers purpose, it is recommended to set the Sleep Enable SE bit just before the execution of the SLEEP instruction.

- **Bits 4,3 - SM1/SM0: Sleep Mode Select bits 1 and 0**

This bit selects between the three available sleep modes as shown in the following table:

Table 4. Sleep Mode Select

SM1	SM0	Sleep Mode
0	0	Idle Mode
0	1	Reserved
1	0	Power Down
1	1	Power Save

Bits 2..0 - Res: Reserved bits

These bits are reserved bits in the ATmega603/103 and always read zero.

XTAL Divide Control Register - XDIV

The XTAL Divide Control Register is used to divide the XTAL clock frequency by a number in the range 1 - 129. This feature can be used to decrease power consumption when the requirement for processing power is low.

Bit	7	6	5	4	3	2	1	0	
$3C ($5C)	**XDIVEN**	**XDIV6**	**XDIV5**	**XDIV4**	**XDIV3**	**XDIV2**	**XDIV1**	**XDIV0**	**XDIV**
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 - XDIVEN: XTAL Divide Enable**

When the XDIVEN bit is set (one), the clock frequency of the CPU and all peripherals is divided by the factor defined by the setting of XDIV6 - XDIV0. This bit can be set and cleared run-time to vary the clock frequency as suitable to the application.

- **Bits 6..0 - XDIV6..XDIV0: XTAL Divide Select Bits 6 - 0**

These bits define the division factor that applies when the XDIVEN bit is set (one). If the value of these bits is denoted *d*, the following formula defines the resulting CPU clock frequency f_{clk}:

$$f_{clk} = \frac{\text{XTAL}}{129 - d}$$

The value of these bits can only be changed when XDIVEN is zero. When XDIVEN is set to one, the value written simultaneously into XDIV6..XDIV0 is taken as the division factor. When XDIVEN is cleared to zero, the value written simultaneously into XDIV6..XDIV0 is rejected. As the divider divides the master clock input to the MCU, the speed of all peripherals is reduced when a division factor is used.

Reset and Interrupt Handling

The ATmega603/103 provides 23 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 set (one) together with the I-bit in the status register in order to enable the interrupt.

The lowest addresses in the program memory space are automatically defined as the Reset and Interrupt vectors. The complete list of vectors is shown in Table 5. 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 etc.

Table 5. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	$0000	RESET	Hardware Pin and Watchdog Reset
2	$0002	INT0	External Interrupt Request 0
3	$0004	INT1	External Interrupt Request 1
4	$0006	INT2	External Interrupt Request 2
5	$0008	INT3	External Interrupt Request 3
6	$000A	INT4	External Interrupt Request 4
7	$000C	INT5	External Interrupt Request 5
8	$000E	INT6	External Interrupt Request 6
9	$0010	INT7	External Interrupt Request 7
10	$0012	TIMER2 COMP	Timer/Counter2 Compare Match
11	$0014	TIMER2 OVF	Timer/Counter2 Overflow
12	$0016	TIMER1 CAPT	Timer/Counter1 Capture Event
13	$0018	TIMER1 COMPA	Timer/Counter1 Compare Match A
14	$001A	TIMER1 COMPB	Timer/Counter1 Compare Match B
15	$001C	TIMER1 OVF	Timer/Counter1 Overflow
16	$001E	TIMER0 COMP	Timer/Counter0 Compare Match
17	$0020	TIMER0 OVF	Timer/Counter0 Overflow
18	$0022	SPI, STC	SPI Serial Transfer Complete
19	$0024	UART, RX	UART, Rx Complete
20	$0026	UART, UDRE	UART Data Register Empty
21	$0028	UART, TX	UART, Tx Complete
22	$002A	ADC	ADC Conversion Complete
23	$002C	EE READY	EEPROM Ready
24	$002E	ANALOG COMP	Analog Comparator

The most typical and general program setup for the Reset and Interrupt Vector Addresses are:

```
Address   Labels     Code              Comments
$0000               jmp     RESET     ; Reset Handler
$0002               jmp     EXT_INT0  ; IRQ0 Handler
$0004               jmp     EXT_INT1  ; IRQ1 Handler
$0006               jmp     EXT_INT2  ; IRQ2 Handler
$0008               jmp     EXT_INT3  ; IRQ3 Handler
$000A               jmp     EXT_INT4  ; IRQ4 Handler
$000C               jmp     EXT_INT5  ; IRQ5 Handler
$000E               jmp     EXT_INT6  ; IRQ6 Handler
$0010               jmp     EXT_INT7  ; IRQ7 Handler
$0012               jmp     TIM2_COMP ; Timer2 Compare Handler
$0014               jmp     TIM2_OVF  ; Timer2 Overflow Handler
$0016               jmp     TIM1_CAPT ; Timer1 Capture Handler
$0018               jmp     TIM1_COMPA ; Timer1 CompareA Handler
$001A               jmp     TIM1_COMPB ; Timer1 CompareB Handler
$001C               jmp     TIM1_OVF  ; Timer1 Overflow Handler
$001E               jmp     TIM0_COMP ; Timer0 Compare Handler
$0020               jmp     TIM0_OVF  ; Timer0 Overflow Handler
$0022               jmp     SPI_STC   ; SPI Transfer Complete Handler
$0024               jmp     UART_RXC  ; UART RX Complete Handler
$0026               jmp     UART_DRE  ; UDR Empty Handler
$0028               jmp     UART_TXC  ; UART TX Complete Handler
$002A               jmp     ADC       ; ADC Conversion Complete Handler
$002C               jmp     EE_RDY    ; EEPROM Ready Handler
$002E               jmp     ANA_COMP  ; Analog Comparator Handler
;
$0030    MAIN:      <instr>  xxx       ; Main program start
  ...     ...         ...    ...
```

Reset Sources

The ATmega603/103 has three sources of reset:

- Power-On Reset. The MCU is reset when a supply voltage is applied to the VCC and GND pins.
- External Reset. The MCU is reset when a low level is present on the pin for more than two XTAL cycles
- Watchdog Reset. The MCU is reset when the Watchdog timer period expires and the Watchdog is enabled.

During reset, all I/O registers except the MCU Status register are then set to their initial values, and the program starts execution from address $0000. The instruction placed in address $0000 must be a JMP - absolute 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 24 shows the reset logic. Table 6 defines the timing and electrical parameters of the reset circuitry.

Figure 24. Reset Logic

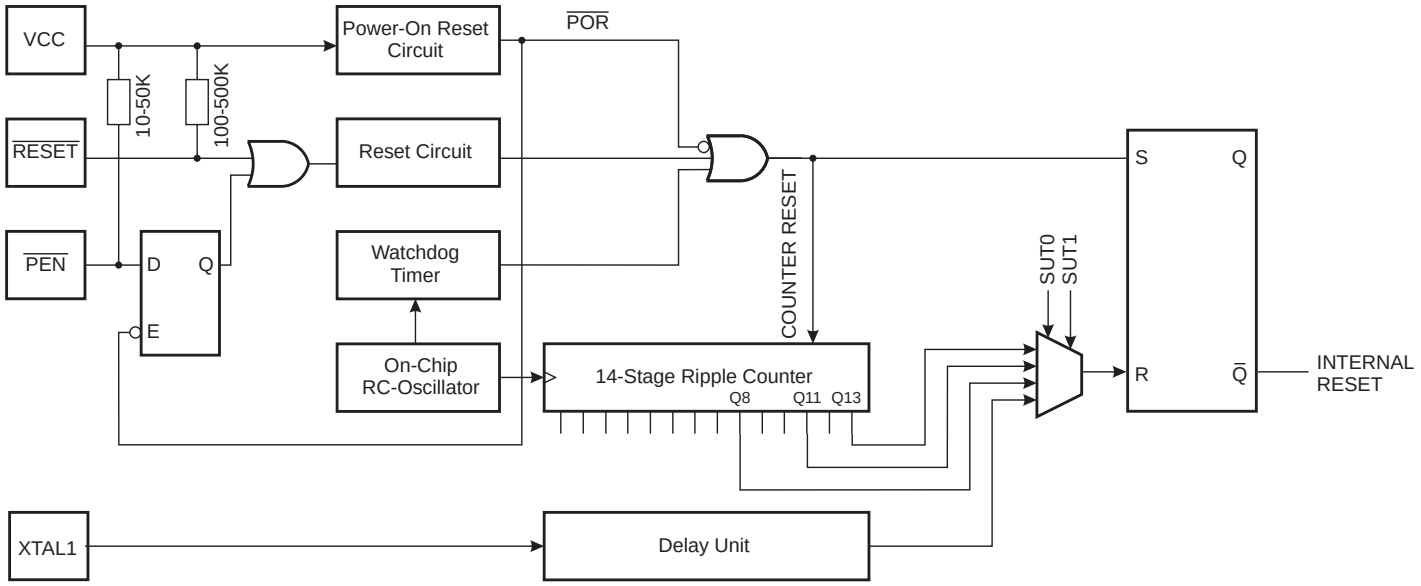


Table 6. Reset Characteristics (V_{CC} = 5V)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{POT}	Power-On Reset Threshold Voltage		1.8	2	2.2	V
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage			$V_{CC}/2$		V
V_{BO}	Brown-out Reset Voltage			2.5		V
T_{TOUT}	Reset Delay Time-Out Period	SUT1/0 = 00		5		CPU cycles
		SUT1/0 = 01	0.4	0.5	0.6	ms
		SUT1/0 = 10	3.2	4.0	4.8	
		SUT1/0 = 11	12.8	16.0	19.2	

Power-On Reset

A Power-On Reset (POR) circuit ensures that the device is not started until V_{CC} has reached a safe level. As shown in Figure 24, an internal timer clocked from the Watchdog timer oscillator prevents the MCU from starting until after a certain period after V_{CC} has reached the Power-On Threshold voltage - V_{POT}, regardless of the V_{CC} rise time (see Figure 25 and Figure 26). The Fuse bits SUT1 and SUT0 is used to select start-up time as indicated in Table 5. The user can select the start-up time according to typical oscillator start-up time. The setting SUT 1/0 = 00 which starts the MCU after 5 clock cycles is used when an external clock signal is applied to the XTAL1 pin. This enables very fast start-up from the sleep modes power down or power save if the clock signal is present during sleep. For details, refer to the programming specification starting on page 72.

If the built-in start-up delay is sufficient, $\overline{\text{RESET}}$ can be connected to V_{CC} directly or via an external pull-up resistor. By holding the pin low for a period after V_{CC} has been applied, the Power-On Reset period can be extended. Refer to Figure 27 for a timing example on this.

Figure 25. MCU Start-Up, $\overline{\text{RESET}}$ Tied to VCC. Rapidly Rising V_{CC}

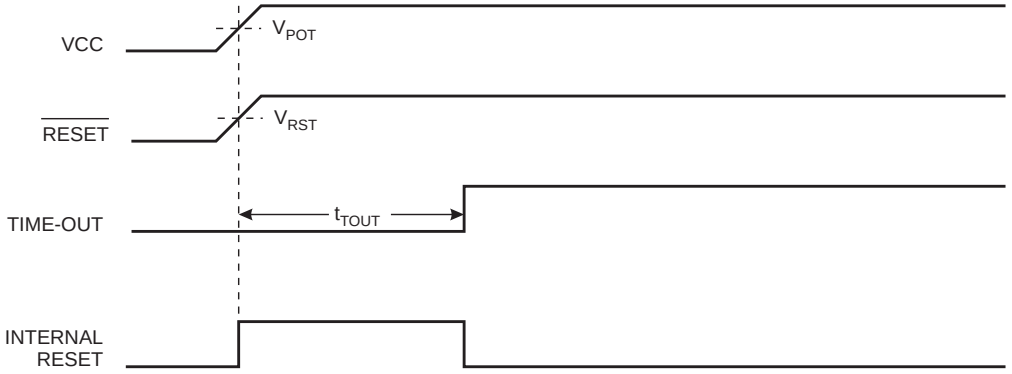


Figure 26. MCU Start-Up, $\overline{\text{RESET}}$ Tied to VCC. Slowly Rising V_{CC}

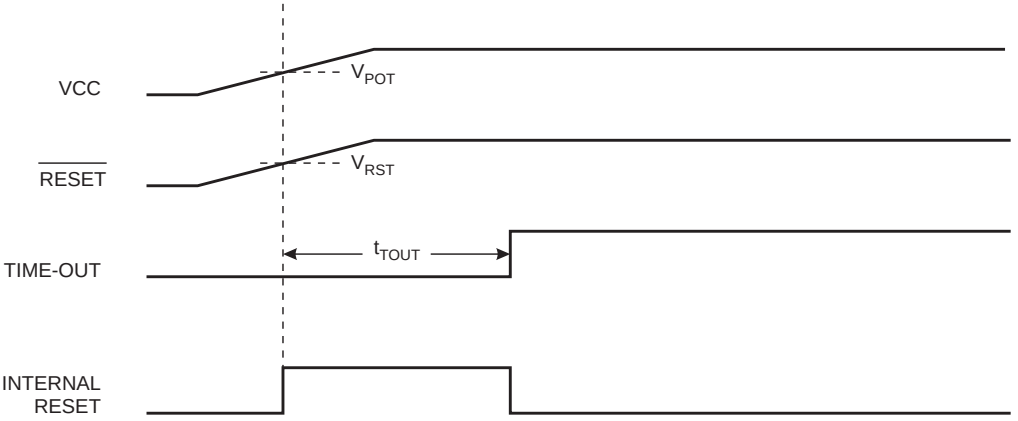
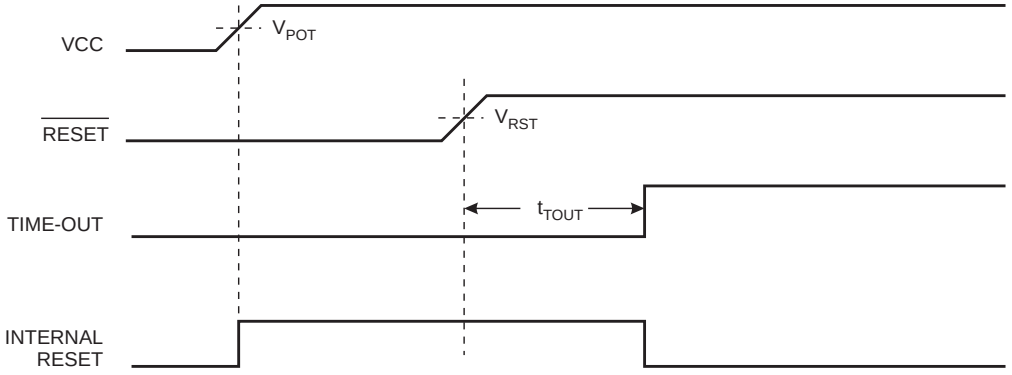


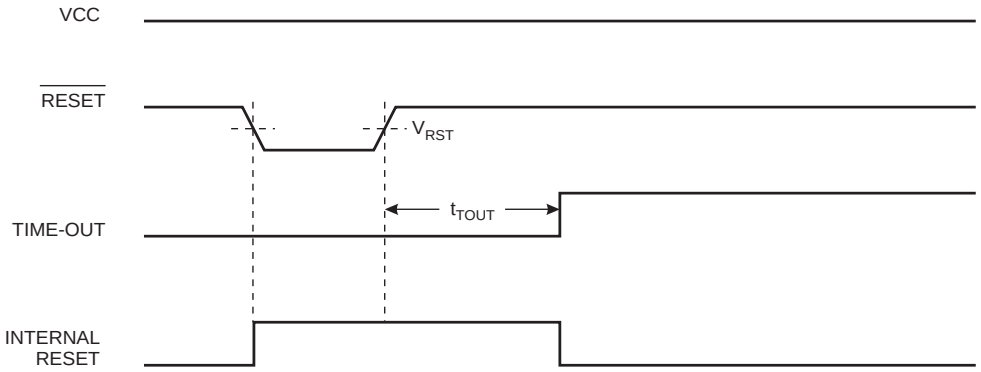
Figure 27. MCU Start-Up, $\overline{\text{RESET}}$ Controlled Externally



External Reset

An external reset is generated by a low level on the $\overline{\text{RESET}}$ pin. The pin must be held low for at least two crystal clock cycles. When the applied signal reaches the Reset Threshold Voltage - V_{RST} on its positive edge, the delay timer starts the MCU after the Time-out period t_{TOUT} has expired.

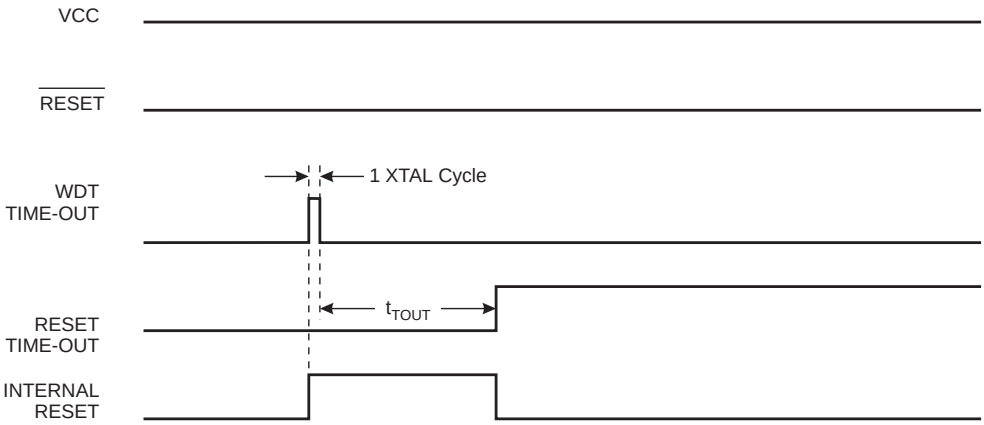
Figure 28. External Reset During Operation



Watchdog Reset

When the Watchdog times out, it will generate a short reset pulse of 1 XTAL cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period t_{TOUT}. Refer to page 38 for details on operation of the Watchdog.

Figure 29. Watchdog Reset During Operation



MCU Status Register - MCUSR

The MCU Status Register provides information on which reset source caused an MCU reset.

Bit	7	6	5	4	3	2	1	0	
$34 ($54)	-	-	-	-	-	-	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial value	0	0	0	0	0	0	See bit description		

- **Bits 7..2 - Res: Reserved Bits**

These bits are reserved bits in the ATmega603/103 and always read as zero.

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

After a power on reset, this bit is undefined (X). It will be set by an external reset. A watchdog reset will leave this bit unchanged.

- **Bit 0 - PORF: Power On Reset Flag**

This bit is set by a power on reset. A watchdog reset or an external reset will leave this bit unchanged.

To summarize, the following table shows the value of these two bits after the three modes of reset:

Table 7. PORF and EXTRF Values after Reset

Reset Source	PORF	EXTRF
Power On Reset	1	undefined
External Reset	unchanged	1
Watchdog Reset	unchanged	unchanged

To make use of these bits to identify a reset condition, the user software should clear both the PORF and EXTRF bits as early as possible in the program. Checking the PORF and EXTRF values is done before the bits are cleared. If

the bit is cleared before an external or watchdog reset occurs, the source of reset can be found by using the following truth table:

Table 8. Reset Source Identification

PORF	EXTRF	Reset Source
0	0	Watchdog Reset
0	1	External Reset
1	0	Power On Reset
1	1	Power On Reset

Interrupt Handling

The ATmega603/103 has two dedicated 8-bit Interrupt Mask control registers; EIMSK - External Interrupt Mask register and TIMSK - Timer/Counter Interrupt Mask register. In addition, other enable and mask bits can be found in the peripheral control registers.

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared (zero) and all interrupts are disabled. The user software can set the I-bit inside an interrupt routine to enable nested interrupts. The RETI command will exit the interrupt routine and set the I-bit to one.

When the Program Counter is vectored to the actual interrupt vector in order to execute the interrupt handling routine, hardware clears the corresponding flag that generated the interrupt. Some of the interrupt flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared.

The External Interrupt Mask Register - EIMSK

Bit	7	6	5	4	3	2	1	0	
$39 ($59)	**INT7**	**INT6**	**INT5**	**INT4**	**INT3**	**INT2**	**INT1**	**INT0**	**EIMSK**
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..4 - INT7 - INT4: External Interrupt Request 7-4 Enable**

When an INT7- INT4 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the corresponding external pin interrupt is enabled. The Interrupt Sense Control bits in the External Interrupt Control Register - EICR defines whether the external interrupt is activated on rising or falling edge or level sensed. Activity on any of these pins will trigger an interrupt request even if the pin is enabled as an output. This provides a way of generating a software interrupt.

- **Bits 3..0 - INT3 - INT0: External Interrupt Request 3-0 Enable**

When an INT3 - INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the corresponding external pin interrupt is enabled. The external interrupts are always low level triggered interrupts. Activity on any of these pins will trigger an interrupt request even if the pin is enabled as an output. This provides a way of generating a software interrupt. When enabled, a level triggered interrupt will generate an interrupt request as long as the pin is held low.

The External Interrupt Flag Register - EIFR

Bit	7	6	5	4	3	2	1	0	
$38 ($58)	**INTF7**	**INTF6**	**INTF5**	**INTF4**	**-**	**-**	**-**	**-**	**EIFR**
Read/Write	R/W	R/W	R/W	R/W	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..4 - INTF7 - INTF4: External Interrupt 7-4 Flags**

When an event on the INT7 - INT4 pins triggers an interrupt request, the corresponding interrupt flag, INTF7 - INTF4 becomes set (one). If the I-bit in SREG and the corresponding interrupt enable bit, INT7 - INT4 in EIMSK, are set (one), the MCU will jump to the interrupt vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag is cleared by writing a logical one to it.

- **Bits 3..0 - Res: Reserved Bits**

These bits are reserved bits in the ATmega603/103 and always read as zero.

The External Interrupt Control Register - EICR

Bit	7	6	5	4	3	2	1	0	
$3A ($5A)	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	EICR
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 - ISCX1, ISCX0: External Interrupt 7-4 Sense Control bits**

The External Interrupts 7 - 4 are activated by the external pins INT7 - INT4 if the SREG I-flag and the corresponding interrupt mask in the EIMSK is set. The level and edges on the external pins that activate the interrupts are defined in the following table:

Table 9. Interrupt 1 Sense Control

ISCX1	ISCX0	Description
0	0	The low level of INTX generates an interrupt request.
0	1	Reserved
1	0	The falling edge of INTX generates an interrupt request.
1	1	The rising edge of INTX generates an interrupt request.

Note: X = 7, 6, 5 or 4.
When changing the ISC11/ISC10 bits, the interrupt must be disabled by clearing its Interrupt Enable bit in the GIMSK Register. Otherwise an interrupt can occur when the bits are changed.

If enabled, a level triggered interrupt will generate an interrupt request as long as the pin is held low.

The Timer/Counter Interrupt Mask Register - TIMSK

Bit	7	6	5	4	3	2	1	0	
$37 ($57)	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
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 - OCIE2: Timer/Counter2 Output Compare Interrupt Enable**

When the OCIE2 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter2 Compare Match interrupt is enabled. The corresponding interrupt (at vector $0012) is executed if a Compare match in Timer/Counter2 occurs. The Compare Flag in Timer/Counter2 is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

- **Bit 6 - TOIE2: Timer/Counter2 Overflow Interrupt Enable**

When the TOIE2 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter2 Overflow interrupt is enabled. The corresponding interrupt (at vector $0014) is executed if an overflow in Timer/Counter2 occurs. The Timer/Counter2 Overflow Flag is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

- **Bit 5 - TICIE1: Timer/Counter1 Input Capture Interrupt Enable**

When the TICIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 Input Capture Event Interrupt is enabled. The corresponding interrupt (at vector $0016) is executed if a capture-triggering event occurs on pin 29, PD4(IC1). The Input Capture Flag in Timer/Counter1 is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

- **Bit 4 - OCE1A: Timer/Counter1 Output CompareA Match Interrupt Enable**

When the OCIE1A bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareA Match interrupt is enabled. The corresponding interrupt (at vector $0018) is executed if a CompareA match in Timer/Counter1 occurs. The CompareA Flag in Timer/Counter1 is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

- **Bit 3 - OCIE1B: Timer/Counter1 Output CompareB Match Interrupt Enable**

When the OCIE1B bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareB Match interrupt is enabled. The corresponding interrupt (at vector $001A) is executed if a CompareB match in Timer/Counter1 occurs. The CompareB Flag in Timer/Counter1 is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

- **Bit 2 - TOIE1: Timer/Counter1 Overflow Interrupt Enable**

When the TOIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 Overflow interrupt

is enabled. The corresponding interrupt (at vector $001C) is executed if an overflow in Timer/Counter1 occurs. The Timer/Counter1 Overflow Flag is set (one) in the Timer/Counter Interrupt Flag Register - TIFR. When Timer/Counter1 is in PWM mode, the Timer Overflow flag is set when the counter changes counting direction at $0000.

- **Bit 1 - OCIE0: Timer/Counter0 Output Compare Interrupt Enable**

When the OCIE0 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter0 Compare Match interrupt is enabled. The corresponding interrupt (at vector $001E) is executed if a Compare match in Timer/Counter0 occurs. The Compare Flag in Timer/Counter2 is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

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

When the TOIE0 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt (at vector $0020) is executed if an overflow in Timer/Counter0 occurs. The Timer/Counter0 Overflow Flag is set (one) in the Timer/Counter Interrupt Flag Register - TIFR.

The Timer/Counter Interrupt Flag Register - TIFR

Bit	7	6	5	4	3	2	1	0	
$36 ($56)	**OCF2**	**TOV2**	**ICF1**	**OCF1A**	**OCF1B**	**TOV1**	**OCF0**	**TOV0**	**TIFR**
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 - OCF2: Output Compare Flag 2:**

The OCF2 bit is set (one) when compare match occurs between Timer/Counter2 and the data in OCR2 - Output Compare Register 2. OCF2 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF2 is cleared by writing a logic one to the flag. When the I-bit in SREG, and OCIE2 (Timer/Counter2 Compare Interrupt Enable), and the OCF2 are set (one), the Timer/Counter2 Output Compare Interrupt is executed.

- **Bit 6 - TOV2: Timer/Counter2 Overflow Flag**

The TOV2 bit is set (one) when an overflow occurs in Timer/Counter2. TOV2 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV2 is cleared by writing a logic one to the flag. When the I-bit in SREG, and TOIE2 (Timer/Counter1 Overflow Interrupt Enable), and TOV2 are set (one), the Timer/Counter2 Overflow Interrupt is executed. In PWM mode, this bit is set when Timer/Counter1 changes counting direction at $00.

- **Bit 5 - ICF1: Input Capture Flag 1**

The ICF1 bit is set (one) to flag an input capture event, indicating that the Timer/Counter1 value has been transferred to the input capture register - ICR1. ICF1 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ICF1 is cleared by writing a logic one to the flag.

- **Bit 4 - OCF1A: Output Compare Flag 1A**

The OCF1A bit is set (one) when compare match occurs between the Timer/Counter1 and the data in OCR1A - Output Compare Register 1A. OCF1A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF1A is cleared by writing a logic one to the flag. When the I-bit in SREG, and OCIE1A (Timer/Counter1 Compare Interrupt Enable), and the OCF1A are set (one), the Timer/Counter1 Compare match Interrupt is executed.

- **Bit 3 - OCF1B: Output Compare Flag 1B**

The OCF1B bit is set (one) when compare match occurs between the Timer/Counter1 and the data in OCR1B - Output Compare Register 1B. OCF1B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF1B is cleared by writing a logic one to the flag.. When the I-bit in SREG, and OCIE1B (Timer/Counter1 Compare match InterruptB Enable), and the OCF1B are set (one), the Timer/Counter1 Compare match Interrupt is executed.

- **Bit 2 - TOV1: Timer/Counter1 Overflow Flag**

The TOV1 is set (one) when an overflow occurs in Timer/Counter1. TOV1 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV1 is cleared by writing a logic one to the flag. When the I-bit in SREG, and TOIE1 (Timer/Counter1 Overflow Interrupt Enable), and TOV1 are set (one), the Timer/Counter1 Overflow Interrupt is executed. In PWM mode, this bit is set when Timer/Counter1 changes counting direction at $0000.

- **Bit 1 - OCIF0: Output Compare Flag 0**

The OCF0 bit is set (one) when compare match occurs between Timer/Counter0 and the data in OCR0 - Output Compare Register 0. OCF0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0 is cleared by writing a logic one to the flag. When the I-bit in SREG, and OCIE0 (Timer/Counter2 Compare Interrupt Enable), and the OCF0 are set (one), the Timer/Counter0 Output Compare Interrupt is executed.

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

The bit TOV0 is set (one) 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, and TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set (one), the

Timer/Counter0 Overflow interrupt is executed. In PWM mode, this bit is set when Timer/Counter1 changes counting direction at $00.

Interrupt Response Time
The interrupt execution response for all the enabled *AVR* interrupts is 4 clock cycles minimum. 4 clock cycles after the interrupt flag has been set, the program vector address for the actual interrupt handling routine is executed. During this 4 clock cycle period, the Program Counter (2 bytes) is pushed onto the Stack, and the Stack Pointer is decremented by 2. The vector is a jump to the interrupt routine, and this jump takes 3 clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served.

A return from an interrupt handling routine (same as for a subroutine call routine) takes 4 clock cycles. During these 4 clock cycles, the Program Counter (2 bytes) is popped back from the Stack, and the Stack Pointer is incremented by 2. 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 - SREG - is not handled by the *AVR* hardware, neither for interrupts nor for subroutines. For the interrupt handling routines requiring a storage of the SREG, this must be performed by user software.

For Interrupts triggered by events that can remain static (E.g. the Output Compare Register1 A matching the value of Timer/Counter1) the interrupt flag is set when the event occurs. If the interrupt flag is cleared and the interrupt condition persists, the flag will not be set until the event occurs the next time.

Sleep Modes

To enter any of the three sleep modes, the SE bit in MCUCR must be set (one) and a SLEEP instruction must be executed. The SM1 and SM0 bits in the MCUCR register selects which sleep mode, Idle, Power Down or Power Save, is activated by the SLEEP instruction.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU awakes, executes the interrupt routine, and resumes execution from the instruction following SLEEP. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector. The contents of the register file, SRAM and I/O memory are unaltered when the device wakes up from sleep. See Table 4 on how to select sleep mode.

Note that if a *level* triggered interrupt is used for wake-up from power down or power save, the low level must be held for a time longer than the reset delay time-out period t_{TOUT}. Otherwise, the device will not wake up.

Idle Mode
When the SM1/SM0 bits are set to 00, the SLEEP instruction forces the MCU into the Idle Mode stopping the CPU but allowing Timer/Counters, Watchdog and the interrupt system to continue operating. This enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and UART Receive Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the analog comparator can be powered down by setting the ACD-bit in the Analog Comparator Control and Status register - ACSR. This will reduce power consumption in Idle Mode. When the MCU wakes up from Idle mode, the CPU starts program execution immediately.

Power Down Mode
When the SM1/SM0 bit is 10, the SLEEP instruction forces the MCU into the Power Down Mode. In this mode, the external oscillator is stopped. The user can select whether the watchdog shall be enabled during power-down mode. If the watchdog is enabled, it will wake up the MCU when the Watchdog Time-out period expires. If the watchdog is disabled, only an external reset or an external level triggered interrupt can wake up the MCU. When an external clock source is applied to XTAL1, wake-up from Power Down can done without the delay that is normally used to stabilize the XTAL oscillator. This wake-up mode is enabled by programming the SUT0/SUT1 fuses in the Flash. Refer to page 72 for a description on programming.

Power Save Mode
When the SM1/SM0 bits are 11, the SLEEP instruction forces the MCU into the Power Save Mode. This mode is identical to Power Down, with one exception:

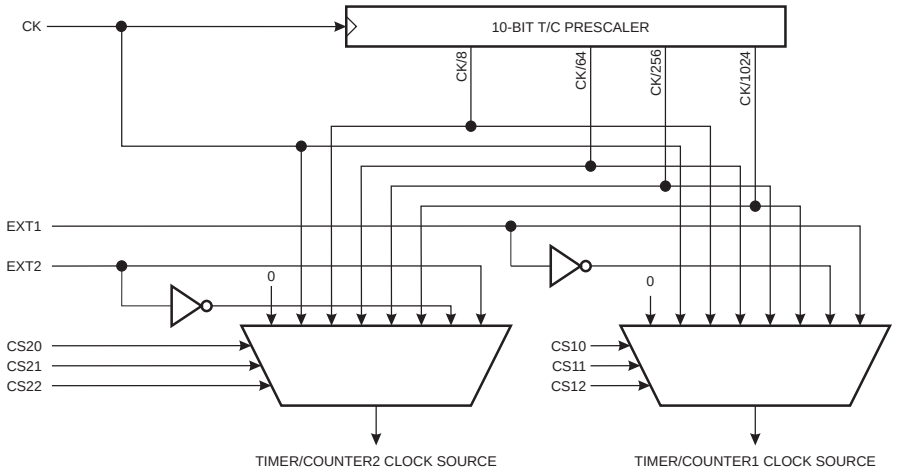
If Timer/Counter0 is clocked asynchronously, i.e. the AS0 bit in ASSR is set, Timer/Counter0 will run during sleep. The device can wake up from either Timer Overflow or Output Compare interrupt from Timer/Counter0.

Timer / Counters

The ATmega603/103 provides three general purpose Timer/Counters - two 8-bit T/Cs and one 16-bit T/C. Timer/Counter0 can as an option be asynchronously clocked from an external oscillator. This oscillator is optimized for use with a 32.768 kHz crystal, enabling use of Timer/Counter0 as a Real Time Clock (RTC). Timer/Counter0 has its own prescaler. Timer/Counters 1 and 2 have individual prescaling selection from the same 10-bit prescaling timer. These Timer/Counters can either be used as a timer with an internal clock timebase or as a counter with an external pin connection which triggers the counting.

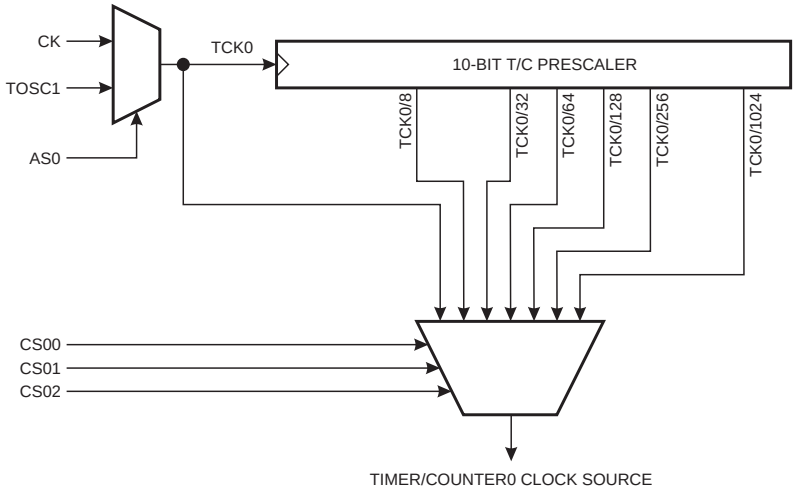
The Timer/Counter Prescalers

Figure 30. Prescaler for Timer/Counter 1 and Timer/Counter2



For Timer/Counters 1 and 2, the four different prescaled selections are: CK/8, CK/64, CK/256 and CK/1024 where CK is the oscillator clock. For Timer/Counters 1 and 2, added selections as CK, external source and stop, can be selected as clock sources.

Figure 31. The Timer/Counter0 Prescaler



The clock source for Timer/Counter0 is named TCK0. TCK0 is by default connected to the main system clock CK. By setting the AS0 bit in ASSR, Timer/Counter 0 is asynchronously clocked from the TOSC1 pin. This enables use of Timer/Counter0 as a Real Time Clock (RTC). A crystal can be connected between the TOSC1 and TOSC2 pins to serve as an independent clock source for Timer/Counter0. This oscillator is optimized for use with a 32.768 kHz crystal.

The 8-Bit Timer/Counters T/C0 and T/C2

Figure 32 shows the block diagram for Timer/Counter0.

Figure 32. Timer/Counter0 Block Diagram

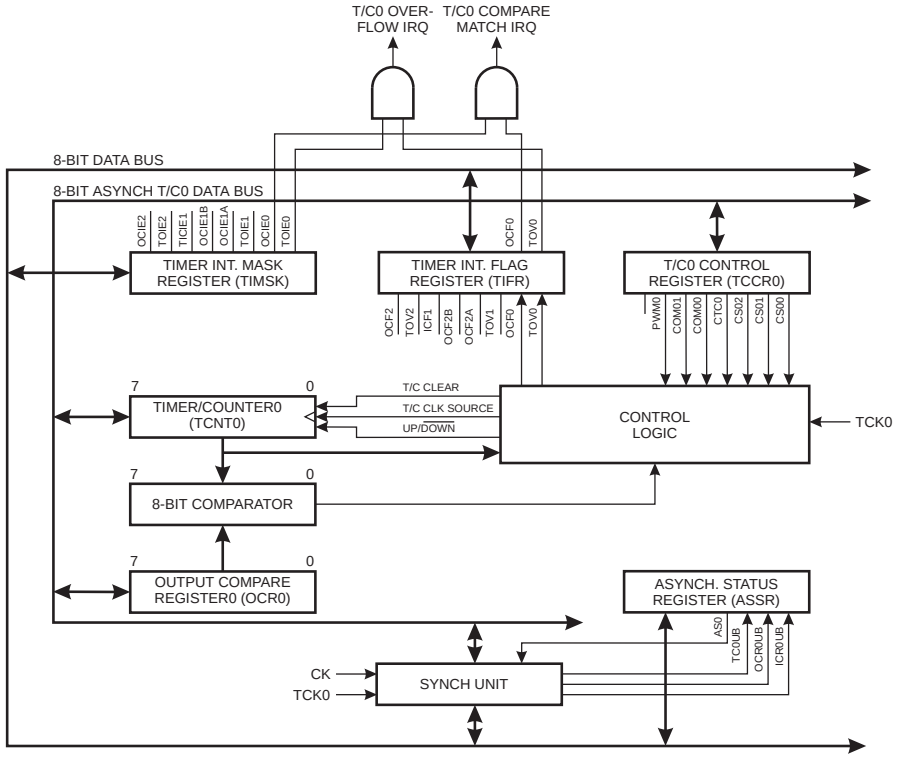
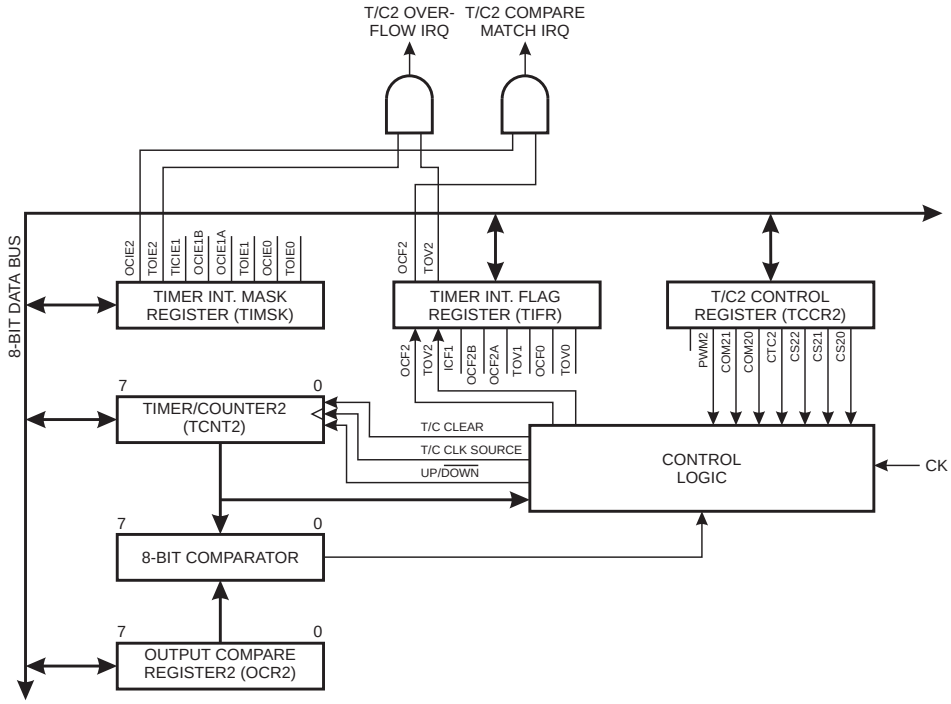


Figure 33. Timer/Counter2 Block Diagram



Note: Figure 33 shows the block diagram for Timer/Counter2.

The 8-bit Timer/Counter0 can select clock source from TCK0 or prescaled TCK0.

The 8-bit Timer/Counter2 can select clock source from CK, prescaled CK, or an external pin.

Both Timer/Counters can be stopped as described in the specification for the Timer/Counter Control Registers - TCCR0 and TCCR2.

The different status flags (overflow, compare match and capture event) are found in the Timer/Counter Interrupt Flag Register - TIFR. Control signals are found in the Timer/Counter Control Registers - TCCR0 and TCCR2. The interrupt enable/disable settings are found in the Timer/Counter Interrupt Mask Register - TIMSK.

When Timer/Counter2 is externally clocked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock.

The 8-bit Timer/Counters feature a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make these units useful for lower speed functions or exact timing functions with infrequent actions.

Both Timer/Counters support two Output Compare functions using the Output Compare Registers - OCR0 and OCR2 as the data source to be compared to the Timer/Counter contents. The Output Compare functions include optional clearing of the counter on compare match, and action on the Output Compare Pins - PB4(OC0/PWM0) and PB7(OC2/PWM2) - on compare match.

Timer/Counter0 and 2 can also be used as 8-bit Pulse Width Modulators. In this mode the Timer/Counter and the output compare register serve as a glitch-free, stand-alone PWM with centered pulses. Refer to page 30 for a detailed description on this function.

The Timer/Counter0 Control Register - TCCR0

Bit	7	6	5	4	3	2	1	0	
33 ($53)	-	PWM0	COM01	COM00	CTC0	CS02	CS01	CS00	TCCR0
Read/Write	R	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/Counter2 Control Register - TCCR2

Bit	7	6	5	4	3	2	1	0	
$25 ($45)	-	PWM2	COM21	COM20	CTC2	CS22	CS21	CS20	TCCR2
Read/Write	R	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 - Res: Reserved Bit**

This bit is a reserved bit in the ATmega603/103 and always reads as zero.

- **Bit 6 - PWM0 / PWM2: Pulse Width Modulator Enable**

When set (one) this bit enables PWM mode for Timer/Counter0 or Timer/Counter2. This mode is described on page 30.

- **Bits 5,4 - COM01, COM00 / COM21, COM20: Compare Output Mode, bits 1 and 0**

The COMn1 and COMn0 control bits determine any output pin action following a compare match in Timer/Counter2. Any output pin actions affect pins PB4(OC0/PWM0) or PB7(OC2/PWM2). Since this is an alternative function to an I/O port, the corresponding direction control bit must be set (one) to control an output pin. The control configuration is shown in Table 10.

Table 10. Compare Mode Select

COMn1	COMn0	Description
0	0	Timer/Countern disconnected from output pin OCn/PWMn
0	1	Toggle the OCn/PWMn output line.
1	0	Clear the OCn/PWMn output line (to zero).
1	1	Set the OCn/PWMn output line (to one).

Note: n = 0 or 2
In PWM mode, these bits have a different function. Refer to Table 13 for a detailed description.
When changing the COMn1/COMn0 bits, the Output Compare n Interrupt must be disabled by clearing its Interrupt Enable bit in the TIMSK Register. Otherwise an interrupt can occur when the bits are changed.

- **Bit 3 - CTC0 / CTC2: Clear Timer/Counter on Compare match**

When the CTC0 or CTC2 control bit is set (one), the Timer/Counter is reset to $00 in the CPU clock cycle after a compare match. If the control bit is cleared, the Timer continues counting and is unaffected by a compare match. Since the compare match is detected in the CPU clock cycle following the match, this function will behave differently when a prescaling higher than 1 is used for the timer. When a prescaling of 1 is used, and the compareA register is set to C, the timer will count as follows if CTC0/2 is set:

... | C-1 | C | C+1 | 0 | 1 | ...

When the prescaler is set to divide by 8, the timer will count like this:

... | C-1, C-1, C-1, C-1, C-1, C-1, C-1, C-1 | C, C, C, C, C, C, C, C | C+1, 0, 0, 0, 0, 0, 0, 0, 0 | 1, 1, 1, ...

In PWM mode, this bit has no effect.

- **Bits 2,1,0 - CS02, CS01, CS00 / CS22, CS21, CS20: Clock Select bits 2,1 and 0**

The Clock Select2 bits 2,1 and 0 define the prescaling source of the Timer/Counter.

Table 11. Timer/Counter0 Prescale Select

CS02	CS01	CS00	Description
0	0	0	Timer/Counter0 is stopped.
0	0	1	TCK0
0	1	0	TCK0 / 8
0	1	1	TCK0 / 32
1	0	0	TCK0 / 64
1	0	1	TCK0 / 128
1	1	0	TCK 0/ 256
1	1	1	TCK0 / 1024

Table 12. Timer/Counter2 Prescale Select

CS22	CS21	CS20	Description
0	0	0	Timer/Counter2 is stopped.
0	0	1	CK
0	1	0	CK / 8
0	1	1	CK / 64
1	0	0	CK / 256
1	0	1	CK / 1024
1	1	0	External Pin PD7(T2), falling edge
1	1	1	External Pin PD7(T2), rising edge

The Stop condition provides a Timer Enable/Disable function. The CK down divided modes are scaled directly from the CK oscillator clock. If the external pin modes are used for Timer/Counter2, transitions on PD7/(T2) will clock the counter even if the pin is configured as an output.

Timer/Counter0 - TCNT0

Bit	7	6	5	4	3	2	1	0	
$32 ($42)	MSB							LSB	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	

Timer/Counter2 - TCNT2

Bit	7	6	5	4	3	2	1	0	
$24 ($44)	MSB							LSB	TCNT2
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	

These 8-bit register contains the value of the Timer/Counters.

Both Timer/Counters are realized as up or up/down (in PWM mode) counters with read and write access. If the Timer/Counter is written to and a clock source is selected, it continues counting in the timer clock cycle after it is preset with the written value.

Timer/Counter0 Output Compare Register - OCR0

Bit	7	6	5	4	3	2	1	0	
$31 ($51)	MSB							LSB	OCR0
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	

Timer/Counter2 Output Compare Register - OCR2

Bit	7	6	5	4	3	2	1	0	
$23 ($43)	MSB							LSB	OCR2
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 are 8-bit read/write registers.

The Timer/Counter Output Compare Registers contain the data to be continuously compared with the Timer/Counter. Actions on compare matches are specified in TCCR0 and TCCR2. A compare match does only occur if Timer/Counter1 counts to the OCR value. A software write that sets the Timer/Counter and Output Compare Register to the same value does not generate a compare match.

A compare match will set the compare interrupt flag in the CPU clock cycle following the compare event.

Precaution must be taken when Timer/Counter0 operates in Asynchronous mode, i.e. the AC0 bit in ASSR is set(one). When writing OCR0, the value is transferred to the register on the TCK0 clock following the write operation.

Timer/Counter 0 and 2 in PWM mode

When the PWM mode is selected, the Timer/Counter and the Output Compare Register - OCR0 or OCR2 form an 8-bit, free-running, glitch-free and phase correct PWM with outputs on the PB4(OC0/PWM0) or PB7(OC2/PWM2) pin. The Timer/Counter acts as an up/down counter, counting up from $00 to $FF, when it turns and counts down again to zero before the cycle is repeated. When the counter value matches the contents of the Output Compare register, the PB4(OC0/PWM0) or PB7(OC2/PWM2) pin is set or cleared according to the settings of the COM01/COM00 or COM21/COM20 bits in the Timer/Counter Control Registers TCCR0 and TCCR2. Refer to Table 13 for details.

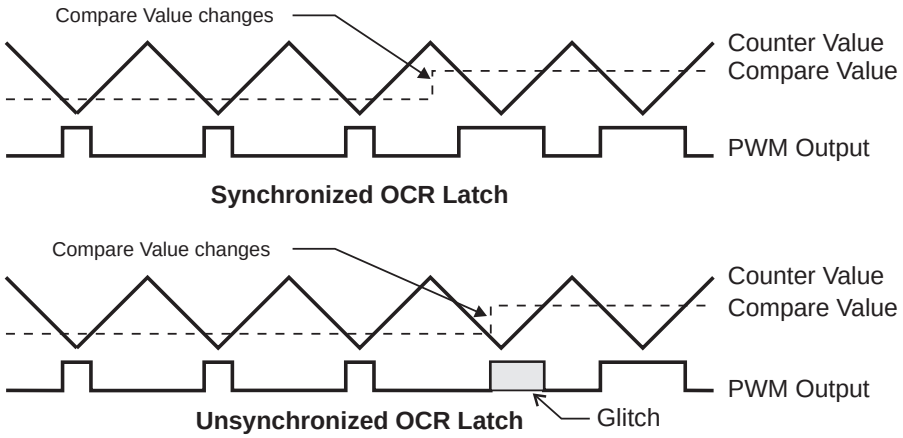
Table 13. Compare Mode Select in PWM Mode

COMn1	COMn0	Effect on Compare/PWM Pin
0	0	Not connected
0	1	Not connected
1	0	Cleared on compare match, upcounting. Set on compare match, downcounting (non-inverted PWM).
1	1	Cleared on compare match, downcounting. Set on compare match, upcounting (inverted PWM).

Note: n = 0 or 2

Note that in PWM mode, the Output Compare register is transferred to a temporary location when written. The value is latched when the Timer/Counter reaches $FF. This prevents the occurrence of odd-length PWM pulses (glitches) in the event of an unsynchronized OCR0 or OCR2 write. See Figure 34 for an example.

Figure 34. Effects on Unsynchronized OCR Latching



During the time between the write and the latch operation, a read from OCR0 or OCR2 will read the contents of the temporary location. This means that the most recently written value always will read out of OCR0/2

When the OCR register contains $00 or $FF, the PWM output is held low or high according to the settings of COM21/COM20 or COM11/COM10. This is shown in Table 14:

Table 14. PWM Outputs OCRn = $00 or $FF

COMn1	COMn0	OCRn	Output PWMn
1	0	$00	L
1	0	$FF	H
1	1	$00	H
1	1	$FF	L

Note: n = 0 or 2

In PWM mode, the Timer Overflow Flag, TOV0 or TOV2, is set when the counter changes direction at $00. Timer Overflow Interrupt0 and 2 operates exactly as in normal Timer/Counter mode, i.e. it is executed when TOV0 or TOV2 is set provided that Timer Overflow Interrupt and global interrupts are enabled. This does also apply to the Timer Output Compare flags and interrupts.

The frequency of the PWM will be Timer Clock Frequency divided by 510.

Asynchronous Operation of Timer/Counter0

When Timer/Counter0 operates synchronously, all operations and timing are identical to Timer/Counter2. During asynchronous operation, however, some considerations must be taken.

- WARNING: When switching between asynchronous and synchronous clocking of Timer/Counter0, the timer registers, TCNT0, OCR0 and TCCR0 might get corrupted. Safe procedure for swithcing clock source:

1. Disable the timer 0 interrupts OCIE0 and TOIE0.
2. Select clock source by setting ASO as appropriate.
3. Write new values to TCNT0, OCR0 and TCCR0.
4. If switching to asynchronous operation: Wait for TCNT0UB, OCR0UB and TCR0UB.
5. Enable interrupts if needed.

- The oscillator is optimized for use with a 32,768Hz watch crystal. An external clock signal applied to this pin goes through the same amplifier having a bandwidth of

256kHz. The external clock signal should therefore be in the interval 0Hz - 256kHz. The frequency of the clock signal applied to the TOSC1 pin must be lower than one fourth of the CPU main clock frequency. Observe that CPU clock frequency can be lower than XTAL frequency if the XTAL divider is enabled.

- When writing to one of the registers TCNT0, OCR0, or TCCR0, the value is transferred to a temporary register, and latched after two positive edges on TOSC1. The user should not write a new value before the contents of the temporary register have been transferred to its destination. Each of the three mentioned registers have their individual temporary register, which means that e.g. writing to TCNT0 does not disturb an OCR0 write in progress. To detect that a transfer to the destination register has taken place, a Asynchronous Status Register - ASSR has been implemented.
- When entering a sleep mode after having written to TCNT0, OCR0 or TCCR0, the user must wait until the written register has been updated if Timer/Counter0 is used to wake up the device. Otherwise, the MCU will go to sleep before the changes have had any effect. This is extremely important if the output compare0 interrupt is used to wake up the device; Output compare is disabled during write to OCR0 or TCNT0. If the write cycle is not finished (i.e. the user goes to sleep before the OCR0UB bit returns to zero), the device will never get a compare match and the MCU will not wake up.
- If Timer/Counter0 is used to wake up the device from Power Save mode, precautions must be taken if the user wants to re-enter Power Save mode; The interrupt logic needs one TOSC1 cycle to get reset. If the time between wake up and re-entering Power Save mode is less than one TOSC1 cycle, the interrupt will not occur and the device will fail to wake up. If the user is in doubt whether the time before re-entering Power Save is sufficient, the following algorithm can be used to ensure that one TOSC1 cycle has elapsed:
  1. Write a value to TCCR0, TCNT0 or OCR0
  2. Wait until the corresponding Update Busy flag in ASSR returns to zero.
  3. Enter Power Save mode
- The 32 kHz oscillator for Timer/Counter0 is always running, except in power down mode. After a power up reset or wakeup from power down, the user should be aware of the fact that this oscillator might take as long as one second to stabilize. The user is advised to wait for at least one second before using Timer/Counter0 after power-up or wake-up from power down.
- Description of wake up from power save mode when the timer is clocked asynchronously: When the interrupt condition is met, the wake up process is started on the following cycle of the timer clock, that is, the timer is always advanced by at least one before the processor can read the counter value. The interrupt flags are updated 3 processor cycles after the processor clock has started. During these cycles, the processor executes instructions, but the interrupt condition is not readable, and the interrupt routine has not started yet.
- During asynchronous operation, the synchronization of the interrupt flags for the asynchronous timer takes 3 processor cycles plus one timer cycle. The timer is therefore advanced by at least one before the processor can read the timer value causing the setting of the interrupt flag. The output compare pin is changed on the timer clock, and is not synchronized to the processor clock.

Asynchronous Status Register - ASSR

Bit	7	6	5	4	3	2	1	0	
$3C ($5C)	-	-	-	-	AS0	TCN0UB	OCR0UB	TCR0UB	ASSR
Read/Write	R	R	R	R	R/W	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

Bit 7..4 - Res: Reserved Bits
These bits are reserved bits in the ATmega603/103 and always reads as zero.

- **Bit 3 - AS0: Asynchronous Timer/Counter0**

When set(one) Timer/Counter0 is clocked from the TOSC1 pin. When cleared (zero) Timer/Counter0 is clocked from the internal system clock, CK. When the value of this bit is changed the contents of TCNT0 might get corrrupted.

- **Bit 2 - TCN0UB: Timer/Counter0 Update Busy**

When Timer/Counter0 operates asynchronously and TCNT0 is written, this bit becomes set (one). When the value written to TCNT0 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that TCNT0 is ready to be updated with a new value.

- **Bit 1 - OCR0UB: Output Compare Register0 Update Busy**

When Timer/Counter0 operates asynchronously and OCR0 is written, this bit becomes set (one). When the value written to OCR0 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that OCR0 is ready to be updated with a new value.

- **Bit 0 - TCR0UB: Timer/Counter Control Register0 Update Busy**

When Timer/Counter0 operates asynchronously and TCCR0 is written, this bit becomes set (one). When the

value written to TCCR0 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that TCCR0 is ready to be updated with a new value.

If a write is performed to any of the three Timer/Counter0 registers while its update busy flag is set (one), the updated value might get corrupted and cause an unintentional interrupt to occur.

When reading TCNT0, OCR0 and TCCR0, there is a difference in result. When reading TCNT0, the actual timer value is read. When reading OCR0 or TCCR0, the value in the temporary storage register is read.

The 16-Bit Timer/Counter1

Figure 35 shows the block diagram for Timer/Counter1.

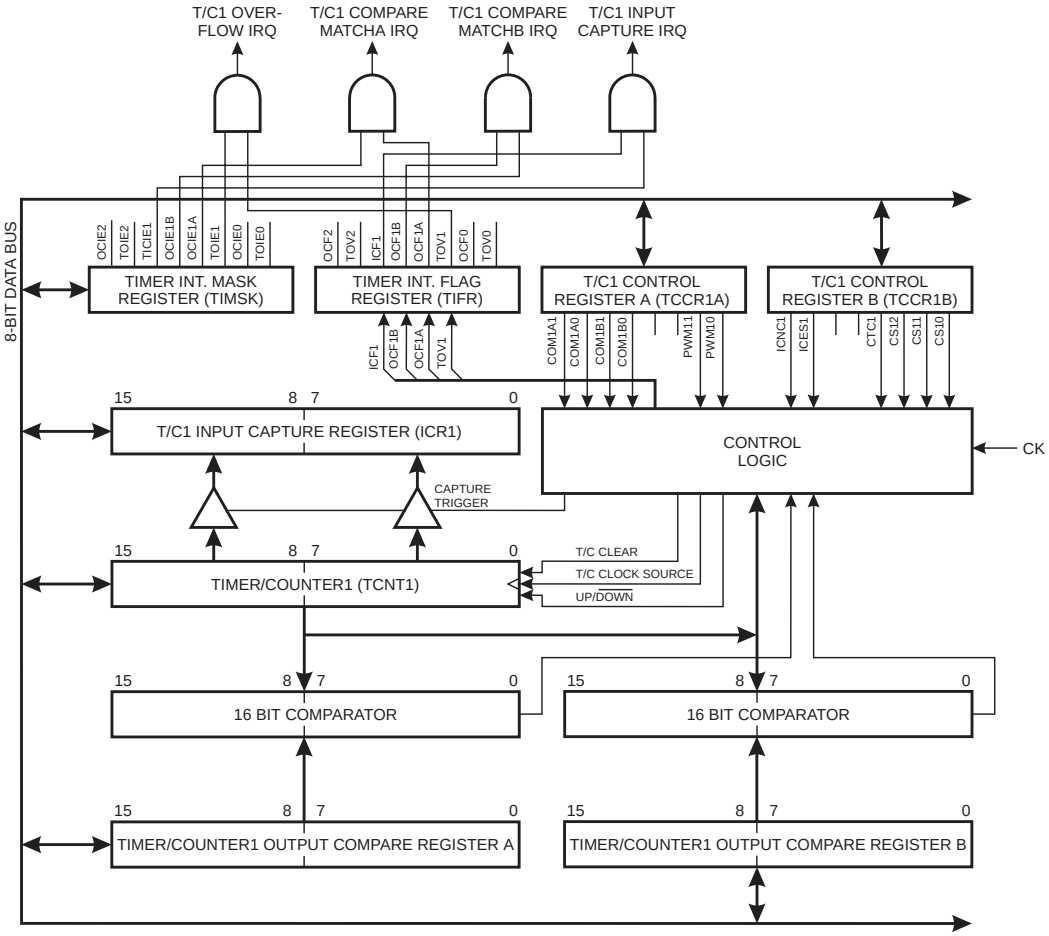
The 16-bit Timer/Counter1 can select clock source from CK, prescaled CK, or an external pin. In addition it can be stopped as described in the specification for the Timer/Counter1 Control Register - TCCR1B. The different status flags (overflow, compare match and capture event) and control signals are found in the Timer/Counter1 Control Registers - TCCR1A and TCCR1B. The interrupt enable/disable settings for Timer/Counter1 are found in the Timer/Counter Interrupt Mask Register - TIMSK.

When Timer/Counter1 is externally clocked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock.

The 16-bit Timer/Counter1 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities makes the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

The Timer/Counter1 supports two Output Compare functions using the Output Compare Register 1 A and B - OCR1A and OCR1B as the data sources to be compared to the Timer/Counter1 contents. The Output Compare functions include optional clearing of the counter on compareA match, and actions on the Output Compare pins on both compare matches.

Figure 35. Timer/Counter1 Block Diagram



Timer/Counter1 can also be used as a 8, 9 or 10-bit Pulse With Modulator. In this mode the counter and the OCR1A/OCR1B registers serve as a dual glitch-free standalone PWM with centered pulses. Refer to page 37 for a detailed description on this function.

The Input Capture function of Timer/Counter1 provides a capture of the Timer/Counter1 contents to the Input Capture Register - ICR1, triggered by an external event on the Input Capture Pin - PD4/(IC1). The actual capture event settings are defined by the Timer/Counter1 Control Register - TCCR1B. In addition, the Analog Comparator can be set to trigger the Input Capture. Refer to the paragraph, “The Analog Comparator”, for details on this.

If the noise canceler function is enabled, the actual trigger condition for the capture event is monitored over 4 samples before the capture is activated. The input pin signal is sampled at XTAL clock frequency.

The Timer/Counter1 Control Register A - TCCR1A

Bit	7	6	5	4	3	2	1	0	
$2F ($4F)	**COM1A1**	**COM1A0**	**COM1B1**	**COM1B0**	**-**	**-**	**PWM11**	**PWM10**	**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	

- **Bits 7,6 - COM1A1, COM1A0: Compare Output Mode1A, bits 1 and 0**

The COM1A1 and COM1A0 control bits determine any output pin action following a compare match in Timer/Counter1. Any output pin actions affect pin OC1A - Output CompareA pin 1. Since this is an alternative function to an I/O port, the corresponding direction control bit must be set (one) to control an output pin. The control configuration is shown in Table 16.

- **Bits 5,4 - COM1B1, COM1B0: Compare Output Mode1B, bits 1 and 0**

The COM1B1 and COM1B0 control bits determine any output pin action following a compare match in Timer/Counter1. Any output pin actions affect pin OC1B - Output CompareB. Since this is an alternative function to an I/O port, the corresponding direction control bit must be set (one) to control an output pin. The following control configuration is given:

Table 15. Compare 1 Mode Select

COM1X1	COM1X0	Description
0	0	Timer/Counter1 disconnected from output pin OC1X
0	1	Toggle the OC1X output line.
1	0	Clear the OC1X output line (to zero).
1	1	Set the OC1X output line (to one).

The COM1B1 and COM1B0 control bits determine any output pin action following a compare match in Timer/Counter1. Any output pin actions affect pin OC1B - Output CompareB. Since this is an alternative function to an I/O port, the corresponding direction control bit must be set (one) to control an output pin. The following control configuration is given: X = A or B

Table 16. Compare 1 Mode Select

COM1X1	COM1X0	Description
0	0	Timer/Counter1 disconnected from output pin OC1X
0	1	Toggle the OC1X output line.
1	0	Clear the OC1X output line (to zero).
1	1	Set the OC1X output line (to one).

In PWM mode, these bits have a different function. Refer to Table 17 for a detailed description.

When changing the COM1X1/COM1X0 bits, Output Compare Interrupts 1 must be disabled by clearing their Interrupt Enable bits in the TIMSK Register. Otherwise an interrupt can occur when the bits are changed.

- **Bits 3..2 - Res: Reserved bits**

These bits are reserved bits in the ATmega603/103 and always read zero.

- **Bits 1..0 - PWM11, PWM10: Pulse Width Modulator Select Bits**

These bits select PWM operation of Timer/Counter1 as specified in Table 19. This mode is described on page 37.

Table 17. PWM Mode Select

PWM11	PWM10	Description
0	0	PWM operation of Timer/Counter1 is disabled
0	1	Timer/Counter1 is an 8-bit PWM
1	0	Timer/Counter1 is a 9-bit PWM
1	1	Timer/Counter1 is a 10-bit PWM

The Timer/Counter1 Control Register B - TCCR1B

Bit	7	6	5	4	3	2	1	0	
$2E ($4E)	**ICNC1**	**ICES1**	**-**	**-**	**CTC1**	**CS12**	**CS11**	**CS10**	**TCCR1B**
Read/Write	R/W	R/W	R	R	R/w	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - ICNC1: Input Capture1 Noise Canceler (4 CKs)**

When the ICNC1 bit is cleared (zero), the input capture trigger noise canceler function is disabled. The input capture is triggered at the first rising/falling edge sampled on the input capture pin PD4(IC1) as specified. When the ICNC1 bit is set (one), four successive samples are measures on PD4(IC1), and all samples must be high/low according to the input capture trigger specification in the ICES1 bit. The actual sampling frequency is XTAL clock frequency.

- **Bit 6 - ICES1: Input Capture1 Edge Select**

While the ICES1 bit is cleared (zero), the Timer/Counter1 contents are transferred to the Input Capture Register - ICR1 - on the falling edge of the input capture pin - PD4(IC1). While the ICES1 bit is set (one), the Timer/Counter1 contents are transferred to the Input Capture Register - ICR1 - on the rising edge of the input capture pin - PD4(IC1).

- **Bits 5, 4 - Res: Reserved bits**

These bits are reserved bits in the ATmega603/103 and always read zero.

- **Bit 3 - CTC1: Clear Timer/Counter1 on Compare Match**

When the CTC1 control bit is set (one), the Timer/Counter1 is reset to $0000 in the clock cycle after a compareA match. If the CTC1 control bit is cleared, Timer/Counter1 continues counting and is unaffected by a compare match. Since the compare match is detected in the CPU clock cycle following the match, this function will behave differently when a prescaling higher than 1 is used for the timer. When a prescaling of 1 is used, and the compareA register is set to C, the timer will count as follows i CTC1 is set:

... | C-1 | C | C+1 | 0 | 1 | ...

When the prescaler is set to divide by 8, the timer will count like this:

... | C-1, C-1, C-1, C-1, C-1, C-1, C-1, C-1 | C, C, C, C, C, C, C, C | C+1, 0, 0, 0, 0, 0, 0, 0, 0 | ...

In PWM mode, this bit has no effect.

- **Bits 2,1,0 - CS12, CS11, CS10: Clock Select1, bit 2,1 and 0**

The lock Select1 bits 2,1 and 0 define the prescaling source of Timer/Counter1.

Table 18. Clock 1 Prescale Select

CS12	CS11	CS10	Description
0	0	0	Stop, the Timer/Counter1 is stopped.
0	0	1	CK
0	1	0	CK / 8
0	1	1	CK / 64
1	0	0	CK / 256
1	0	1	CK / 1024
1	1	0	External Pin T1, rising edge
1	1	1	External Pin T1, falling edge

The Stop condition provides a Timer Enable/Disable function. The CK down divided modes are scaled directly from the CK oscillator clock. If the external pin modes are used, the corresponding setup must be performed in the actual direction control register (cleared to zero gives an input pin).

The Timer/Counter1 - TCNT1H and TCNT1L

Bit	15	14	13	12	11	10	9	8	
$2D ($4D)	**MSB**								**TCNT1H**
$2C ($4C)								**LSB**	**TCNT1L**
	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	
	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	
	0	0	0	0	0	0	0	0	

This 16-bit register contains the prescaled value of the 16-bit Timer/Counter1. 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 register (TEMP). This temporary register is also used when accessing OCR1A, OCR1B and ICR1. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program.

- **TCNT1 Timer/Counter1 Write:**

  When the CPU writes to the high byte TCNT1H, the written data is placed in the TEMP register. Next, when the CPU writes the low byte TCNT1L, this byte of data is combined with the byte data in the TEMP register, and all 16 bits are written to the TCNT1 Timer/Counter1 register simultaneously. Consequently, the high byte TCNT1H must be accessed first for a full 16-bit register write operation. When using Timer/Counter1 as an 8-bit timer, it is sufficient to write the low byte only.

- TCNT1 Timer/Counter1 Read:

  When the CPU reads the low byte TCNT1L, the data of TCNT1L is sent to the CPU and the data of the high byte TCNT1H is placed in the TEMP register. When the CPU reads the data in the high byte TCNT1H, the CPU receives the data in the TEMP register. Consequently, the low byte TCNT1L must be accessed first for a full 16-bit register read operation. When using Timer/counter1 as an 8-bit timer, it is sufficient to read the low byte only.

The Timer/Counter1 is realized as an up or up/down (in PWM mode) counter with read and write access. If Timer/Counter1 is written to and a clock source is selected, the Timer/Counter1 continues counting in the clock cycle after it is preset with the written value.

Timer/Counter1 Output Compare Register - OCR1AH and OCR1AL

Bit	15	14	13	12	11	10	9	8	
$2B	**MSB**								**OCR1AH**
$2A								**LSB**	**OCR1AL**
	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	
	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	
	0	0	0	0	0	0	0	0	

Timer/Counter1 Output Compare Register - OCR1BH and OCR1BL

Bit	15	14	13	12	11	10	9	8	
$29	**MSB**								**OCR1BH**
$28								**LSB**	**OCR1BL**
	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	
	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	
	0	0	0	0	0	0	0	0	

The output compare registers are 16-bit read/write registers.

The Timer/Counter1 Output Compare Registers contain the data to be continuously compared with Timer/Counter1. Actions on compare matches are specified in the Timer/Counter1 Control and Status register. A compare match does only occur if Timer/Counter1 counts to the OCR value. A software write that sets TCNT1 and OCR1A or OCR1B to the same value does not generate a compare match.

A compare match will set the compare interrupt flag in the CPU clock cycle following the compare event.

Since the Output Compare Registers - OCR1A and OCR1B - are 16-bit registers, a temporary register TEMP is used when OCR1A/B are written to ensure that both bytes are updated simultaneously. When the CPU writes the high byte, OCR1AH or OCR1BH, the data is temporarily stored in the TEMP register. When the CPU writes the low byte, OCR1AH or OCR1BH, the TEMP register is simultaneously written to OCR1AL or OCR1BL. Consequently, the high byte OCR1AL or OCR1BL must be written first for a full 16-bit register write operation.

The TEMP register is also used when accessing TCNT1, and ICR1. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program.

The Timer/Counter1 Input Capture Register - ICR1H and ICR1L

Bit	15	14	13	12	11	10	9	8	
$27 ($37)	**MSB**								**ICR1H**
$26 ($36)								**LSB**	**ICR1L**
	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	

The input capture register is a 16-bit read-only register.

When the rising or falling edge (according to the input capture edge setting - ICES1) of the signal at the input capture pin - PD4(IC1) - is detected, the current value of the Timer/Counter1 is transferred to the Input Capture Register - ICR1. At the same time, the input capture flag - ICF1 - is set (one).

Since the Input Capture Register - ICR1 - is a 16-bit register, a temporary register TEMP is used when ICR1 is read to ensure that both bytes are read simultaneously. When the CPU reads the low byte ICR1L, the data is sent to the CPU and the data of the high byte ICR1H is placed in the TEMP register. When the CPU reads the data in the high byte ICR1H, the CPU receives the data in the TEMP register. Consequently, the low byte ICR1L must be accessed first for a full 16-bit register read operation.

The TEMP register is also used when accessing TCNT1, OCR1A and OCR1B. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program.

Timer/Counter1 in PWM mode

When the PWM mode is selected, Timer/Counter1 and the Output Compare Register1A - OCR1A and the Output Compare Register1B - OCR1B, form a dual 8, 9 or 10-bit, free-running, glitch-free and phase correct PWM with outputs on the PD5(OC1A) and OC1B pins. Timer/Counter1 acts as an up/down counter, counting up from $0000 to TOP (see Table 17) , when it turns and counts down again to zero before the cycle is repeated. When the counter value matches the contents of the 10 least significant bits of OCR1A or OCR1B, the PD5(OC1A)/OC1B pins are set or cleared according to the settings of the COM1A1/COM1A0 or COM1B1/COM1B0 bits in the Timer/Counter1 Control Register TCCR1A. Refer to Table 18 for details.

Table 19. Timer TOP Values and PWM Frequency

PWM Resolution	Timer TOP value	Frequency
8-bit	$00FF (255)	$f_{TC1}/510$
9-bit	$01FF (511)	$f_{TC1}/1022$
10-bit	$03FF(1023)	$f_{TC1}/2046$

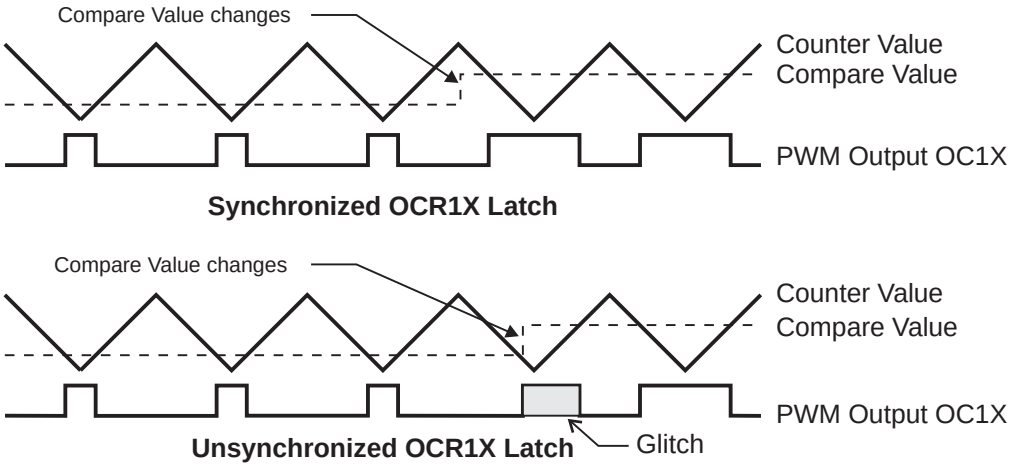
Table 20. Compare1 Mode Select in PWM Mode

COM1X1	COM1X0	Effect on OCX1
0	0	Not connected
0	1	Not connected
1	0	Cleared on compare match, upcounting. Set on compare match, downcounting (non-inverted PWM).
1	1	Cleared on compare match, downcounting. Set on compare match, upcounting (inverted PWM).

Note: X = A or B

Note that in the PWM mode, the 10 least significant OCR1A/OCR1B bits, when written, are transferred to a temporary location. They are latched when Timer/Counter1 reaches the value TOP. This prevents the occurrence of odd-length PWM pulses (glitches) in the event of an unsynchronized OCR1A/OCR1B write. See Figure 36 for an example.

Figure 36. Effects on Unsynchronized OCR1 Latching



Note: X = A or B

During the time between the write and the latch operation, a read from OCR1A or OCR1B will read the contents of the temporary location. This means that the most recently written value always will read out of OCR1A/B

When OCR1 contains $0000 or TOP, the output OC1A/OC1B is held low or high according to the settings of COM1A1/COM1A0 or COM1B1/COM1B0. This is shown in Table 21.

Table 21. PWM Outputs OCR1X = $0000 Or TOP

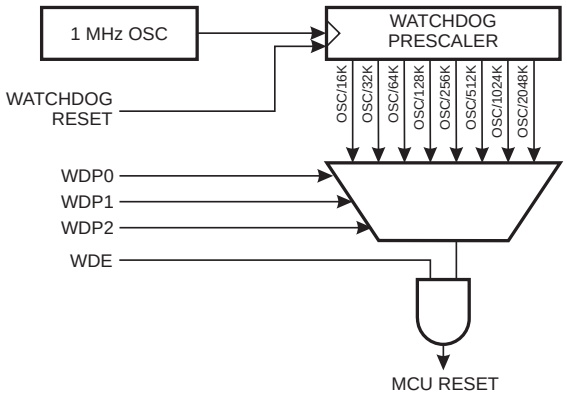
COM1X1	COM1X0	OCR1X	Output OC1X
1	0	$0000	L
1	0	TOP	H
1	1	$0000	H
1	1	TOP	L

Note: X = A or B

In PWM mode, the Timer Overflow Flag1, TOV1, is set when the counter changes direction at $0000. Timer Overflow Interrupt1 operates exactly as in normal Timer/Counter mode, i.e. it is executed when TOV1 is set provided that Timer Overflow Interrupt1 and global interrupts are enabled. This does also apply to the Timer Output Compare1 flags and interrupts.

The Watchdog Timer

Figure 37. Watchdog Timer



The Watchdog Timer is clocked from a separate on-chip oscillator which runs at 1MHz This is the typical value at V_{CC} = 5V. See characterization data for typical values at other V_{CC} levels. By controlling the Watchdog Timer prescaler, the Watchdog reset interval can be adjusted from 16K to 2,048K cycles (nominally 16 - 2048 ms). The WDR - Watchdog Reset - instruction resets the Watchdog Timer. From the Watchdog is reset, eight different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog reset, the ATmega603/103 resets and executes from the reset vector. For timing details on the Watchdog reset, refer to page 21.

When enabling the watchdog, the status of the watchdog timer is unknown. The user is advised to execute a WDR instruction before enabling the watchdog. Otherwise, the device might get reset before the first WDR after enabling is reached.

To prevent unintentional disabling of the watchdog, a special turn-off procedure must be followed when the watchdog is disabled. Refer to the description of the Watchdog Timer Control Register for details.

The Watchdog Timer Control Register - WDTCR

Bit	7	6	5	4	3	2	1	0	
$21 ($41)	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0	WDTCR
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..5 - Res: Reserved bits**

These bits are reserved bits in the ATmega603/103 and will always read as zero.

- **Bit 4 - WDTOE: Watch Dog Turn Off Enable**

This bit must be set (one) when the WDE bit is cleared, Otherwise, the watchdog will not be disabled. Once set, hardware will clear this bit to zero after four clock cycles. Refer to the description of the WDE bit for a watchdog disable procedure.

- **Bit 3 - WDE: Watch Dog Enable**

When the WDE is set (one) the Watchdog Timer is enabled, and if the WDE is cleared (zero) the Watchdog Timer function is disabled. WDE can only be cleared if the WDTOE bit is set (one). To disable an enabled watchdog timer, the following procedure must be followed:

1. In the same operation, write a logical one to WDTOE and WDE. A logical 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 logical 0 to WDE. This disables the watchdog.

- **Bits 2..0 - WDP2, WDP1, WDP0: Watch Dog Timer Prescaler 2, 1 and 0**

The WDP2, WDP1 and WDP0 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 22.

Table 22. Watch Dog Timer Prescale Select

WDP2	WDP1	WDP0	Timeout Period
0	0	0	16K cycles
0	0	1	32K cycles
0	1	0	64K cycles
0	1	1	128K cycles
1	0	0	256K cycles
1	0	1	512K cycles
1	1	0	1,024K cycles
1	1	1	2,048K cycles

EEPROM Read/Write Access

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

The write access time is in the range of 2.5 - 4ms, depending on the V_{CC} voltages. A self-timing function, however, lets the user software detect when the next byte can be written. A special EEPROM Write Complete interrupt can be set to trigger when the EEPROM is ready to accept new data. In order to prevent unintentional EEPROM writes, a specific procedure must be followed. Refer to the description of the EEPROM control register for details on this.

When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed. When it is read, the CPU is halted for 4 clock cycles.

The EEPROM Address Register - EEARH, EEARL

Bit	15	14	13	12	11	10	9	8	
$1F ($3F)	-	-	-	-	EEAR11	EEAR10	EEAR9	EEAR8	EEARH
$1E ($3E)	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	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	
	0	0	0	0	0	0	0	0	

The EEPROM Address Registers - EEARH and EEARL specify the EEPROM address in the 2K/4K-byte EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 2047/4095. The ATmega603 has an EEPROM address space of 2K, and the EEAR11 I/O bit is read-only with initial value of 0.

The EEPROM Data Register - EEDR

Bit	7	6	5	4	3	2	1	0	
$1D ($3D)	MSB							LSB	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.

The EEPROM Control Register - EECR

Bit	7	6	5	4	3	2	1	0	
$1C ($3C)	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
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..4 - Res: Reserved bits**
These bits are reserved bits in the ATmega603/103 and will always be read as zero.

• **Bit 3 - EERIE: EEPROM Ready Interrupt Enable**
When the I bit in SREG and EERIE are set (one), the EEPROM Ready Interrupt is enabled. When cleared (zero), the interrupt is disabled. The EEPROM Ready interrupt constantly generates an interrupt request when EEWE is cleared (zero).

• **Bit 2 - EEMWE: EEPROM Master Write Enable**
The EEMWE bit determines whether setting EEWE to one causes the EEPROM to be written. When EEMWE is set(one) setting EEWE will write data to the EEPROM at the selected address If EEMWE is zero, setting EEWE will have no effect. When EEMWE has been set (one) by software, hardware clears the bit to zero after four clock cycles. See the description of the EEWE bit for a EEPROM write procedure.

• **Bit 1 - EEWE: EEPROM Write Enable**
The EEPROM Write Enable Signal EEWE is the write strobe to the EEPROM. When address and data are correctly set up, the EEWE bit must be set to write the value into the EEPROM. The EEMWE bit must be set when the logical one is written to EEWE, otherwise no EEPROM write takes place. The following procedure should be followed when writing the EEPROM (the order of steps 2 and 3 is unessential):

1. Wait until EEWE becomes zero.
2. Write new EEPROM address to EEAR (optional)
3. Write new EEPROM data to EEDR (optional)
4. Write a logical one to the EEMWE bit in EECR
5. Within four clock cycles after setting EEMWE, write a logical one to EEWE.

When the write access time (typically 2.5 ms at V_{CC} = 5V or 4 ms at V_{CC} = 2.7V) has elapsed, the EEWE bit is cleared (zero) by hardware. The user software can poll this bit and wait for a zero before writing the next byte. When EEWE 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 set. When the EERE bit is cleared (zero) by hardware, requested data is found in the EEDR register. The EEPROM read access takes one instruction there is no need to poll the EERE bit. When EERE has been set, the CPU is halted for four cycles before the next instruction is executed.

The user should poll the EEWE bit before starting the read operation. If a write operation is in progress when new data or address is written to the EEPROM I/O registers, the write operation will be interrupted, and the result is undefined.

Prevent 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 the 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 for executing instructions is too low.

EEPROM data corruption can easily be avoided by following these design recommendations (one is sufficient):

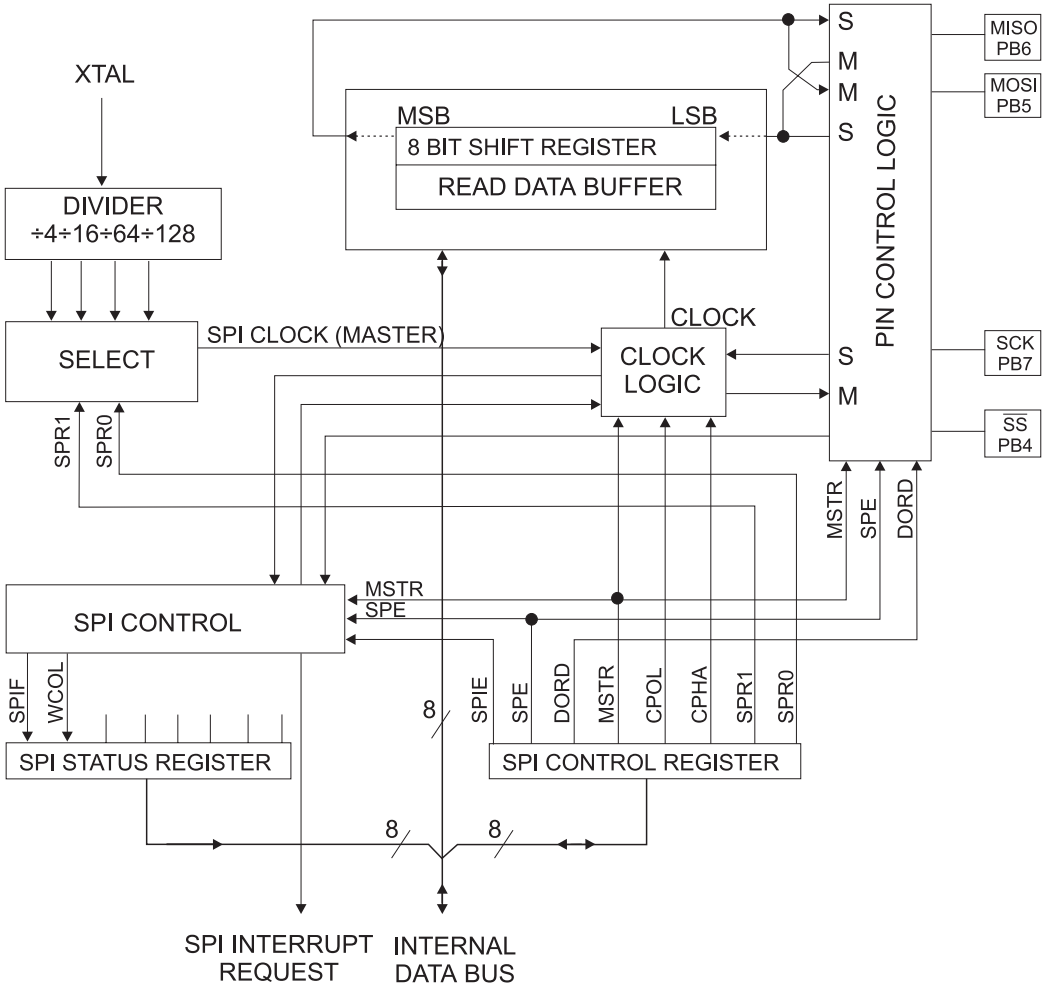
1. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This is best done by an external low V_{CC} Reset Protection circuit, often referred to as a Brown-Out Detector (BOD). Please refer to application notes AVR 190 and AVR 180 for design considerations regarding power-on reset and low voltage detection.
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 EEPROM registers from unintentional writes.
3. Store constants in Flash memory if the ability to change memory contents from software is not required. Flash memory can not be updated by the CPU, and will not be subject to corruption.

The Serial Peripheral Interface - SPI

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the ATmega603/103 and peripheral devices or between several ATmega603/103 devices. The ATmega603/103 SPI features include the following:

- Full-Duplex, 3-Wire Synchronous Data Transfer
- Master or Slave Operation
- LSB First or MSB First Data Transfer
- Four Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wakeup from Idle Mode (Slave Mode Only)

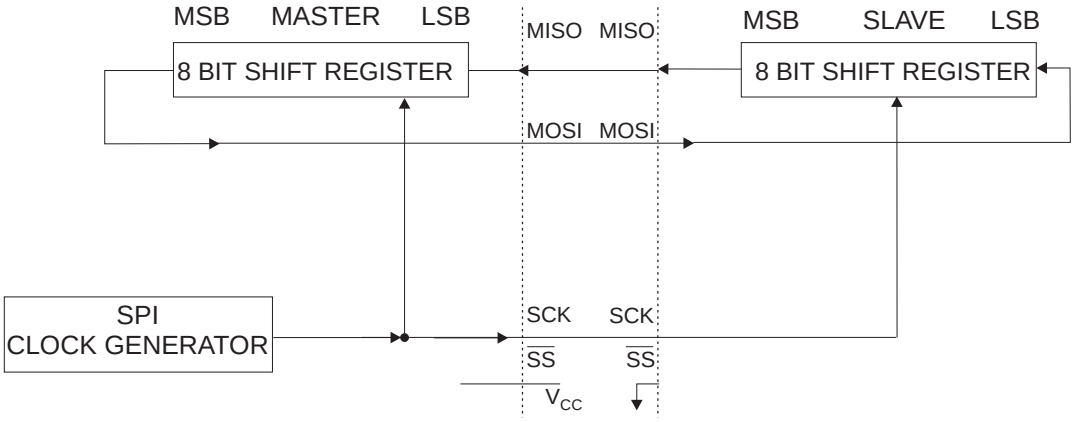
Figure 38. SPI Block Diagram



The interconnection between master and slave CPUs with SPI is shown in Figure 39. The PB1(SCK) pin is the clock output in the master mode and is the clock input in the slave mode. Writing to the SPI data register of the master CPU starts the SPI clock generator, and the data written shifts out of the PB2(MOSI) pin and into the PB2 (MOSI) pin of the slave CPU. After shifting one byte, the SPI clock generator stops, setting the end of transmission flag (SPIF). If the SPI interrupt enable bit (SPIE) in the SPCR register is set, an interrupt is requested. The Slave Select input, PB0($\overline{SS}$), is set low to select an individual SPI device as a slave. When PB0($\overline{SS}$) is set high, the SPI port is deactivated and the PB2(MOSI) pin can be used as an input. Slave/Master mode can also be selected in software by clearing or by setting the MSTR bit in the SPI Control Register.

The two shift registers in the Master and the Slave can be considered as one distributed 16-bit circular shift register. This is shown in Figure 39. When data is shifted from the master to the slave, data is also shifted in the opposite direction, simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

Figure 39. SPI Master-Slave Interconnection



The system is single buffered in the transmit direction and double buffered in the receive direction. This means that characters to be transmitted cannot be written to the SPI Data Register before the entire shift cycle is completed. When receiving data, however, a received character must be read from the SPI Data Register before the next character has been completely shifted in. Otherwise, the first character is lost.

When the SPI is enabled, the data direction of the MOSI, MISO, SCK and $\overline{SS}$ pins is overridden according to the following table:

Table 23. SPI Pin Overrides

PIN	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
SS	User Defined	Input

$\overline{SS}$ Pin Functionality

When the SPI is configured as a master (MSTR in SPCR is set), the user can determine the direction of the $\overline{SS}$ pin. If $\overline{SS}$ is configured as an output, the pin is a general output pin which does not affect the SPI system. If $\overline{SS}$ is configured as an input, it must be hold high to ensure Master SPI operation. If, in master mode, the $\overline{SS}$ pin is input, and is driven low by peripheral circuitry, the SPI system interprets this as another master selecting the SPI as a slave and starting to send data to it. To avoid bus contention, the SPI system takes the following actions:

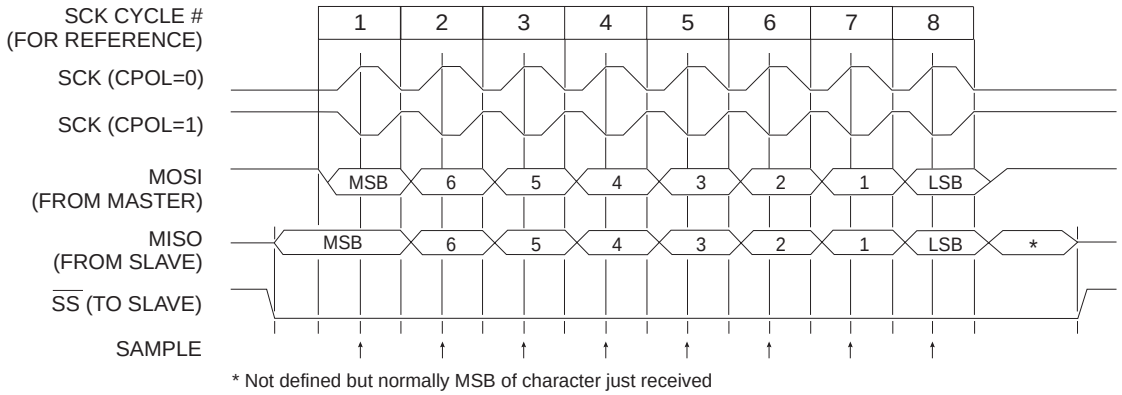
1. The MSTR bit in SPCR is cleared and the SPI system becomes a slave. As a result of the SPI becoming a slave, the MOSI and SCK pins become inputs.
2. The SPIF flag in SPSR is set, and if the SPI interrupt is enabled, the interrupt routine will be executed.

Thus, when interrupt-driven SPI transmittal is used in master mode, and there exists a possibility that $\overline{SS}$ is driven low, the interrupt should always check that the MSTR bit is still set. Once the MSTR bit has been cleared by a slave select, it must be set by the user.

When the SPI is configured as a slave, the $\overline{SS}$ pin is always input. When $\overline{SS}$ is held low, the SPI is activated and MISO becomes an output if configured so by the user. All other pins are inputs. When $\overline{SS}$ is driven high, all pins are inputs, and the SPI is passive, which means that it will not receive incoming data.

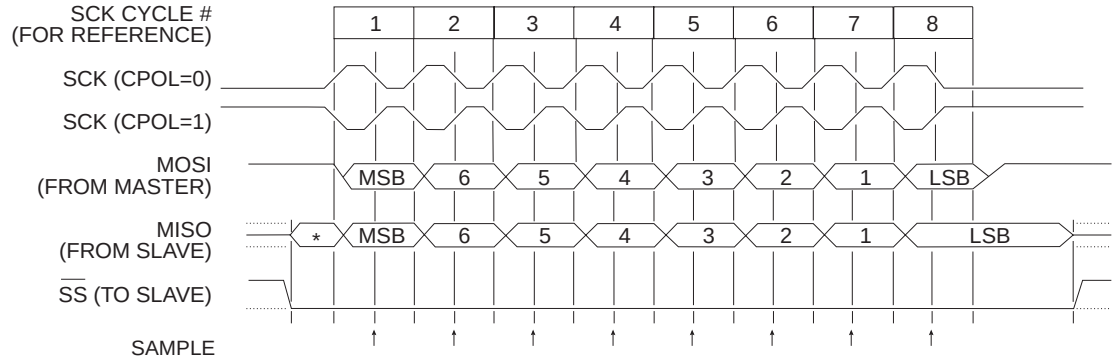
There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 40 and Figure 41.

Figure 40. SPI Transfer Format with CPHA = 0



* Not defined but normally MSB of character just received

Figure 41. SPI Transfer Format with CPHA = 1



* Not defined but normally LSB of previously transmitted character.

The SPI Control Register - SPCR

Bit	7	6	5	4	3	2	1	0	
$0D ($2D)	**SPIE**	**SPE**	**DORD**	**MSTR**	**CPOL**	**CPHA**	**SPR1**	**SPR0**	**SPCR**
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 - SPIE: SPI Interrupt Enable**

This bit causes setting of the SPIF bit in the SPSR register to execute the SPI interrupt provided that global interrupts are enabled.

- **Bit 6 - SPE: SPI Enable**

When the SPE bit is set (one), the SPI is enabled and $\overline{SS}$, MOSI, MISO and SCK are connected to pins PB4, PB5, PB6 and PB7.

- **Bit 5 - DORD: Data Order**

When the DORD bit is set (one), the LSB of the data word is transmitted first.

When the DORD bit is cleared (zero), the MSB of the data word is transmitted first.

- **Bit 4 - MSTR: Master/Slave Select**

This bit selects Master SPI mode when set (one), and Slave SPI mode when cleared (zero). If $\overline{SS}$ is configured as an input and is driven low while MSTR is set, MSTR will be cleared, and SPIF in SPSR will become set. The user will then have to set MSTR to re-enable SPI master mode.

- **Bit 3 - CPOL: Clock Polarity**

When this bit is set (one), SCK is high when idle. When CPOL is cleared (zero), SCK is low when idle. Refer to Figure 40 and Figure 41 for additional information.

- **Bit 2 - CPHA : Clock Phase**

Refer to Figure 40 or Figure 41 for the functionality of this bit.

- **Bits 1,0 - SPR1, SPR0: SPI Clock Rate Select 1 and 0**

These two bits control the SCK rate of the device configured as a master. SPR1 and SPR0 have no effect on the slave. The relationship between SCK and the Oscillator Clock frequency f_{cl} is shown in the following table:

Table 24. Relationship Between SCK and the Oscillator Frequency

SPR1	SPR0	SCK Frequency
0	0	f_{cl} / 4
0	1	f_{cl} / 16
1	0	f_{cl} / 64
1	1	f_{cl} / 128

The SPI Status Register - SPSR

Bit	7	6	5	4	3	2	1	0	
$0E	**SPIF**	**WCOL**	**-**	**-**	**-**	**-**	**-**	**-**	**SPSR**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - SPIF: SPI Interrupt Flag**

When a serial transfer is complete, the SPIF bit is set (one) and an interrupt is generated if SPIE in SPCR is set (one) and global interrupts are enabled. SPIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, the SPIF bit is cleared by first reading the SPI status register with SPIF set (one), then accessing the SPI Data Register (SPDR).

- **Bit 6 - WCOL: Write Collision flag**

The WCOL bit is set if the SPI data register (SPDR) is written during a data transfer. During data transfer, the result of reading the SPDR register may be incorrect, and writing to it will have no effect. The WCOL bit (and the SPIF bit) are cleared (zero) by first reading the SPI Status Register with WCOL set (one), and then accessing the SPI Data Register.

- **Bit 5..0 - Res: Reserved bits**

These bits are reserved bits in the ATmega603/103 and will always read as zero.

The SPI Data Register - SPDR

Bit	7	6	5	4	3	2	1	0	
$0F ($2F)	**MSB**							**LSB**	**SPDR**
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 SPI Data Register is a read/write register used for data transfer between the register file and the SPI Shift register. Writing to the register initiates data transmission. Reading the register causes the Shift Register Receive buffer to be read.

The UART

The ATmega603/103 features a full duplex Universal Asynchronous Receiver and Transmitter (UART). The main features are:

- Baud rate generator generates any baud rate
- High baud rates at low XTAL frequencies
- 8 or 9 bits data
- Noise filtering
- Overrun detection
- Framing Error detection
- False Start Bit detection
- Three separate interrupts on TX Complete, TX Data Register Empty and RX Complete

Data Transmission

A block schematic of the UART transmitter is shown in Figure 42.

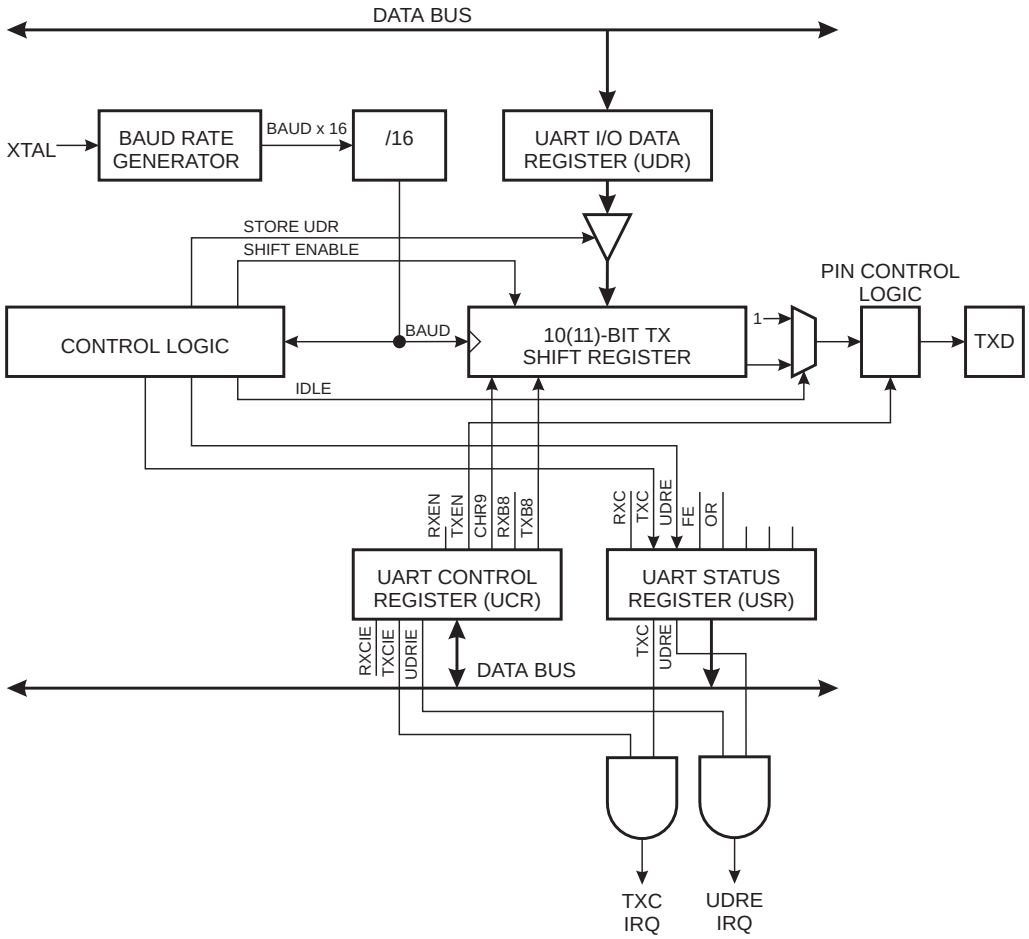
Data transmission is initiated by writing the data to be transmitted to the UART I/O Data Register, UDR. Data is transferred from UDR to the Transmit shift register when:

- A new character has been written to UDR after the stop bit from the previous character has been shifted out. The shift register is loaded immediately.
- A new character has been written to UDR before the stop bit from the previous character has been shifted out. The shift register is loaded when the stop bit of the character currently being transmitted has been shifted out.

If the 10(11)-bit Transmitter shift register is empty, data is transferred from UDR to the shift register. At this time the UDRE (UART Data Register Empty) bit in the UART Status Register, USR, is set. When this bit is set (one), the UART is ready to receive the next character. Writing to UDR clears UDRE. At the same time as the data is transferred from UDR to the 10(11)-bit shift register, bit 0 of the shift

register is cleared (start bit) and bit 9 or 10 is set (stop bit). If 9 bit data word is selected (the CHR9 bit in the UART Control Register, UCR is set), the TXB8 bit in UCR is transferred to bit 9 in the Transmit shift register.

Figure 42. UART Transmitter

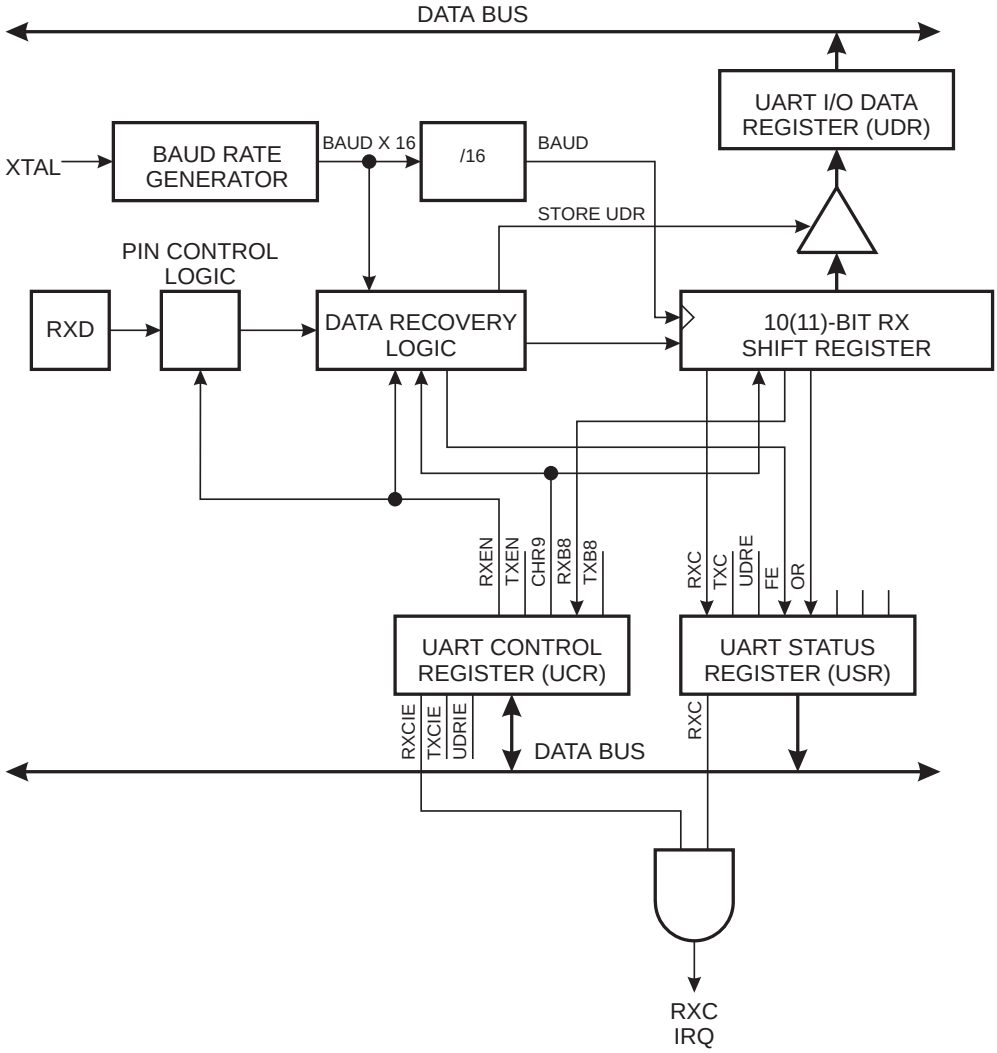


On the Baud Rate clock following the transfer operation to the shift register, the start bit is shifted out on the TXD pin. Then follows the data, LSB first. When the stop bit has been shifted out, the shift register is loaded if any new data has been written to the UDR during the transmission. During loading, UDRE is set. If there is no new data in the UDR register to send when the stop bit is shifted out, the UDRE flag will remain set. In this case, after the stop bit has been present on TXD for one bit length, the TX Complete Flag, TXC, in USR is set.

The TXEN bit in UCR enables the UART transmitter when set (one). When this bit is cleared (zero), the PE1 pin can be used for general I/O. When TXEN is set, the UART Transmitter will be connected to PE1, which is forced to be an output pin regardless of the setting of the DDE1 bit in DDRE.

Data Reception

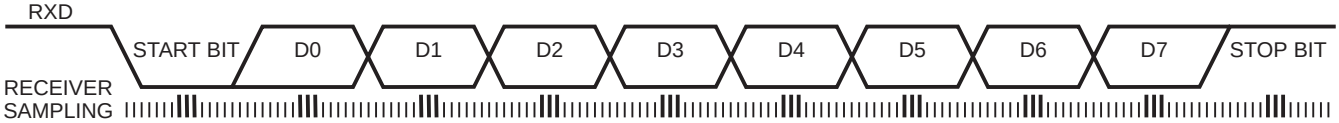
Figure 43. UART Receiver



The receiver front-end logic samples the signal on the RXD pin at a frequency 16 times the baud rate. While the line is idle, one single sample of logical zero will be interpreted as the falling edge of a start bit, and the start bit detection sequence is initiated. Let sample 1 denote the first zero-sample. Following the 1 to 0-transition, the receiver samples the RXD pin at sample 8, 9 and 10. If two or more of these three samples are found to be logical ones, the start bit is rejected as a noise spike and the receiver starts looking for the next 1 to 0-transition.

If however, a valid start bit is detected, sampling of the data bits following the start bit is performed. These bits are also sampled at samples 8, 9 and 10. The logical value found in at least two of the three samples is taken as the bit value. All bits are shifted into the transmitter shift register as they are sampled. Sampling of an incoming character is shown in Figure 44.

Figure 44. Sampling Received Data



When the stop bit enters the receiver, the majority of the three samples must be one to accept the stop bit. If two or more samples are logical zeros, the Framing Error (FE) flag in the UART Status Register (USR) is set when the

received byte is transferred to UDR. Before reading the UDR register, the user should always check the FE bit to detect Framing Errors. FE is cleared when UDR is read.

Whether or not a valid stop bit is detected at the end of a character reception cycle, the data is transferred to UDR and the RXC flag in USR is set. UDR is in fact two physically separate registers, one for transmitted data and one for received data. When UDR is read, the Receive Data register is accessed, and when UDR is written, the Transmit Data register is accessed. If 9 bit data word is selected (the CHR9 bit in the UART Control Register, UCR is set), the RXB8 bit in UCR is loaded with bit 9 in the Transmit shift register when data is transferred to UDR.

If, after having received a character, the UDR register has not been accessed since the last receive, the OverRun (OR) flag in UCR is set. This means that the new data transferred to the shift register could not be transferred to UDR and is lost. The OR bit is buffered, and is available when the valid data byte in UDR has been read. The user should always check the OR after reading from the UDR register in order to detect any overruns.

When the RXEN bit in the UCR register is cleared (zero), the receiver is disabled. This means that the PE0 pin can be used as a general I/O pin. When RXEN is set, the UART Receiver will be connected to PE0, which is forced to be an input pin regardless of the setting of the DDE0 bit in DDRE. When PE0 is forced to input by the UART, the PORTE0 bit can still be used to control the pull-up resistor on the pin.

UART Control

The UART I/O Data Register - UDR

Bit	7	6	5	4	3	2	1	0	
$0C ($2C)	**MSB**							**LSB**	**UDR**
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 UDR register is actually two physically separate registers sharing the same I/O address. When writing to the register, the UART Transmit Data register is written. When reading from UDR, the UART Receive Data register is read.

The UART Status Register - USR

Bit	7	6	5	4	3	2	1	0	
$0B ($2B)	**RXC**	**TXC**	**UDRE**	**FE**	**DOR**	**-**	**-**	**-**	**USR**
Read/Write	R	R/W	R	R	R	R	R	R	
Initial value	0	0	1	0	0	0	0	0	

The USR register is a read-only register providing information on the UART Status.

- **Bit 7 - RXC: UART Receive Complete**

This bit is set (one) when a received character is transferred from the Receiver Shift register to UDR. The bit is set regardless of any detected framing errors. When the RXCIE bit in UCR is set, the UART Receive Complete interrupt will be executed when RXC is set(one). RXC is cleared by reading UDR. When interrupt-driven data reception is used, the UART Receive Complete Interrupt routine must read UDR in order to clear RXC, otherwise a new interrupt will occur once the interrupt routine terminates.

- **Bit 6 - TXC: UART Transmit Complete**

This bit is set (one) when the entire character (including the stop bit) in the Transmit Shift register has been shifted out and no new data has been written to the UDR. This flag is especially useful in half-duplex communications interfaces, where a transmitting application must enter receive mode and free the communications bus immediately after completing the transmission.

When the TXCIE bit in UCR is set, setting of TXC causes the UART Transmit Complete interrupt to be executed. TXC is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, the TXC bit is cleared (zero) by writing a logical one to the bit.

- **Bit 5 - UDRE: UART Data Register Empty**

This bit is set (one) when a character written to UDR is transferred to the Transmit shift register. Setting of this bit indicates that the transmitter is ready to receive a new character for transmission.

When the UDRIE bit in UCR is set, the UART Transmit Complete interrupt to be executed as long as UDRE is set. UDRE is cleared by writing UDR. When interrupt-driven data transmittal is used, the UART Data Register Empty Interrupt routine must write UDR in order to clear UDRE, otherwise a new interrupt will occur once the interrupt routine terminates.

UDRE is set (one) during reset to indicate that the transmitter is ready.

• **Bit 4 - FE: Framing Error**
This bit is set if a Framing Error condition is detected, i.e. when the stop bit of an incoming character is zero.

The FE bit is cleared when the stop bit of received data is one.

• **Bit 3 - DOR: Data OverRun**
This bit is set if an Overrun condition is detected, i.e. when a character already present in the UDR register is not read before the next character is transferred from the Receiver Shift register. The DOR bit is buffered, which means that it is will be set once the valid data still in UDRE is read.

The DOR bit is cleared (zero) when data is received and transferred to UDR.

• **Bits 2..0 - Res: Reserved bits**
These bits are reserved bits in the ATmega603/103 and will always read as zero.

The UART Control Register - UCR

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	**RXCIE**	**TXCIE**	**UDRIE**	**RXEN**	**TXEN**	**CHR9**	**RXB8**	**TXB8**	**UCR**
Read/Write	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 - RXCIE: RX Complete Interrupt Enable**
When this bit is set (one), a setting of the RXC bit in USR will cause the Receive Complete interrupt routine to be executed provided that global interrupts are enabled.

• **Bit 6 - TXCIE: TX Complete Interrupt Enable**
When this bit is set (one), a setting of the TXC bit in USR will cause the Transmit Complete interrupt routine to be executed provided that global interrupts are enabled.

• **Bit 5 - UDRIE: UART Data Register Empty Interrupt Enable**
When this bit is set (one), a setting of the UDRE bit in USR will cause the UART Data Register Empty interrupt routine to be executed provided that global interrupts are enabled.

• **Bit 4 - RXEN: Receiver Enable**
This bit enables the UART receiver when set (one). When the receiver is disabled, the TXC, DOR and FE status flags cannot become set. If these flags are set, turning off RXEN does not cause them to be cleared.

• **Bit 3 - TXEN: Transmitter Enable**
This bit enables the UART transmitter when set (one). When disabling the transmitter while transmitting a character, the transmitter is not disabled before the character in the shift register plus any following character in UDR has been completely transmitted.

• **Bit 2 - CHR9: 9 Bit Characters**
When this bit is set (one) transmitted and received characters are 9 bit long plus start and stop bits. The 9th bit is read and written by using the RXB8 and TXB8 bits in UCR, respectively. The 9th data bit can be used as an extra stop bit or a parity bit.

• **Bit 1 - RXB8: Receive Data Bit 8**
When CHR9 is set (one), RXB8 is the 9th data bit of the received character.

• **Bit 0 - TXB8: Transmit Data Bit 8**
When CHR9 is set (one), TXB8 is the 9th data bit in the character to be transmitted.

The Baud Rate Generator
The baud rate generator is a frequency divider which generates baud-rates according to the following equation:

$$BAUD = \frac{f_{CK}}{16(UBRR + 1)}$$

- BAUD = Baud-Rate
- f_{CK} = Crystal Clock frequency
- UBRR= Contents of the UART Baud Rate register, UBRR (0-255)

For standard crystal frequencies, the most commonly used baud rates can be generated by using the UBRR settings in Table 25. UBRR values which yield an actual baud rate differing less than 2% from the target baud rate, are bolded in the table.

Table 25. UBRR Settings at Various Crystal Frequencies

Baud Rate	1 MHz		%Error	1.8432 MHz		%Error	2 MHz		%Error	2.4576 MHz		%Error
2400	UBRR=	**25**	**0.2**	UBRR=	**47**	**0.0**	UBRR=	**51**	**0.2**	UBRR=	**63**	**0.0**
4800	UBRR=	**12**	**0.2**	UBRR=	**23**	**0.0**	UBRR=	**25**	**0.2**	UBRR=	**31**	**0.0**
9600	UBRR=	6	7.5	UBRR=	**11**	**0.0**	UBRR=	**12**	**0.2**	UBRR=	**15**	**0.0**
14400	UBRR=	3	7.8	UBRR=	**7**	**0.0**	UBRR=	8	3.7	UBRR=	10	3.1
19200	UBRR=	2	7.8	UBRR=	**5**	**0.0**	UBRR=	6	7.5	UBRR=	**7**	**0.0**
28800	UBRR=	1	7.8	UBRR=	**3**	**0.0**	UBRR=	3	7.8	UBRR=	4	6.3
38400	UBRR=	1	22.9	UBRR=	**2**	**0.0**	UBRR=	2	7.8	UBRR=	**3**	**0.0**
57600	UBRR=	0	7.8	UBRR=	**1**	**0.0**	UBRR=	1	7.8	UBRR=	2	12.5
76800	UBRR=	0	22.9	UBRR=	1	33.3	UBRR=	1	22.9	UBRR=	**1**	**0.0**
115200	UBRR=	0	84.3	UBRR=	**0**	**0.0**	UBRR=	0	7.8	UBRR=	0	25.0

Baud Rate	3.2768 MHz		%Error	3.6864 MHz		%Error	4 MHz		%Error	4.608 MHz		%Error
2400	UBRR=	**84**	**0.4**	UBRR=	**95**	**0.0**	UBRR=	**103**	**0.2**	UBRR=	**119**	**0.0**
4800	UBRR=	**42**	**0.8**	UBRR=	**47**	**0.0**	UBRR=	**51**	**0.2**	UBRR=	**59**	**0.0**
9600	UBRR=	**20**	**1.6**	UBRR=	**23**	**0.0**	UBRR=	**25**	**0.2**	UBRR=	**29**	**0.0**
14400	UBRR=	**13**	**1.6**	UBRR=	**15**	**0.0**	UBRR=	16	2.1	UBRR=	**19**	**0.0**
19200	UBRR=	10	3.1	UBRR=	**11**	**0.0**	UBRR=	**12**	**0.2**	UBRR=	**14**	**0.0**
28800	UBRR=	**6**	**1.6**	UBRR=	**7**	**0.0**	UBRR=	8	3.7	UBRR=	**9**	**0.0**
38400	UBRR=	4	6.3	UBRR=	**5**	**0.0**	UBRR=	6	7.5	UBRR=	7	6.7
57600	UBRR=	3	12.5	UBRR=	**3**	**0.0**	UBRR=	3	7.8	UBRR=	**4**	**0.0**
76800	UBRR=	2	12.5	UBRR=	**2**	**0.0**	UBRR=	2	7.8	UBRR=	3	6.7
115200	UBRR=	1	12.5	UBRR=	**1**	**0.0**	UBRR=	1	7.8	UBRR=	2	20.0

Baud Rate	7.3728 MHz		%Error	8 MHz		%Error	9.216 MHz		%Error	11.059 MHz		%Error
2400	UBRR=	**191**	**0.0**	UBRR=	**207**	**0.2**	UBRR=	**239**	**0.0**	UBRR=	287	-
4800	UBRR=	**95**	**0.0**	UBRR=	**103**	**0.2**	UBRR=	**119**	**0.0**	UBRR=	**143**	**0.0**
9600	UBRR=	**47**	**0.0**	UBRR=	**51**	**0.2**	UBRR=	**59**	**0.0**	UBRR=	**71**	**0.0**
14400	UBRR=	**31**	**0.0**	UBRR=	**34**	**0.8**	UBRR=	**39**	**0.0**	UBRR=	**47**	**0.0**
19200	UBRR=	**23**	**0.0**	UBRR=	**25**	**0.2**	UBRR=	**29**	**0.0**	UBRR=	**35**	**0.0**
28800	UBRR=	**15**	**0.0**	UBRR=	16	2.1	UBRR=	**19**	**0.0**	UBRR=	**23**	**0.0**
38400	UBRR=	**11**	**0.0**	UBRR=	**12**	**0.2**	UBRR=	**14**	**0.0**	UBRR=	**17**	**0.0**
57600	UBRR=	**7**	**0.0**	UBRR=	8	3.7	UBRR=	**9**	**0.0**	UBRR=	**11**	**0.0**
76800	UBRR=	**5**	**0.0**	UBRR=	6	7.5	UBRR=	7	6.7	UBRR=	**8**	**0.0**
115200	UBRR=	**3**	**0.0**	UBRR=	3	7.8	UBRR=	**4**	**0.0**	UBRR=	**5**	**0.0**

The UART Baud Rate Register - UBRR

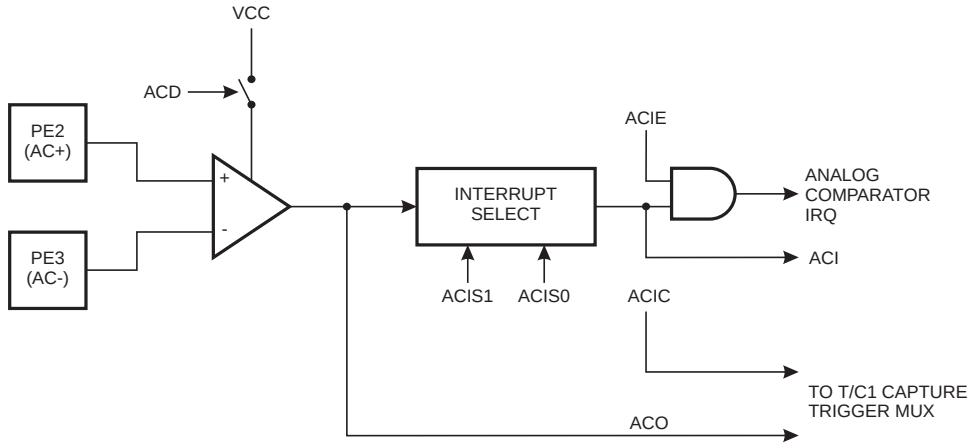
Bit	7	6	5	4	3	2	1	0	
$09 ($29)	**MSB**							**LSB**	**UBRR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

The UBRR register is an 8-bit read/write register which specifies the UART Baud Rate according to the description on the previous page.

The Analog Comparator

The analog comparator compares the input values on the positive pin PE2 (AC+) and negative pin PE3 (AC-). When the voltage on the positive pin PE2 (AC+) is higher than the voltage on the negative pin PE3 (AC-), the Analog Comparator Output, ACO is set (one). The comparator's output can be set to trigger the Timer/Counter1 Input Capture function. In addition, 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 45.

Figure 45. Analog Comparator Block Diagram



The Analog Comparator Control and Status Register - ACSR

Bit	7	6	5	4	3	2	1	0	
$08 ($28)	**ACD**	**-**	**ACO**	**ACI**	**ACIE**	**ACIC**	**ACIS1**	**ACIS0**	**ACSR**
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

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

When this bit is set(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 - Res: Reserved bit**

This bit is a reserved bit in the ATmega603/103 and will always read as zero.

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

ACO is directly connected to the comparator output.

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

This bit is set (one) when a comparator output event triggers the interrupt mode defined by ACI1 and ACI0. The Analog Comparator Interrupt routine is executed if the ACIE bit is set (one) and the I-bit in SREG is set (one). ACI is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ACI is cleared by writing a logic one to the flag. Observe however, that if another bit in this register is modified using the SBI or CBI instruction, ACI will be cleared if it has become set before the operation.

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

When the ACIE bit is set (one) and the I-bit in the Status Register is set (one), the analog comparator interrupt is activated. When cleared (zero), the interrupt is disabled.

- **Bit 2 - ACIC: Analog Comparator Input Capture enable**

When set (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 cleared (zero), no connection between the analog comparator and the Input Capture function is given. To make the comparator trigger the Timer/Counter1 Input Capture interrupt, the TICIE1 bit in the Timer Interrupt Mask Register (TIMSK) must be set (one).

- **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 26.

Table 26. 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

Note: 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.

The Analog to Digital Converter

Feature list:

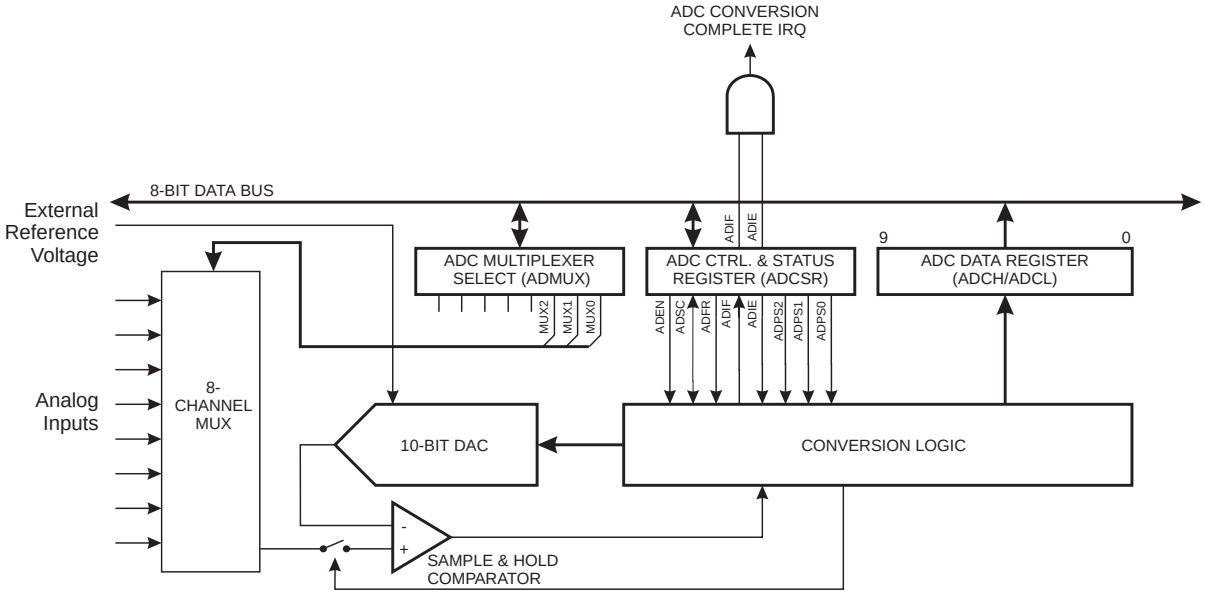
- 10-bit Resolution
- ±½ LSB accuracy
- 70 - 280 µs conversion time
- 8 Multiplexed Input Channels
- Free Run or Single Conversion Mode
- Interrupt on ADC conversion complete.
- Sleep Mode Noise Canceler

The ATmega603/103 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows each pin of Port F to be used as an input for the ADC. The ADC contains a Sample and Hold Amplifier 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 46.

The ADC has two separate analog supply voltage pins, AV_{CC} and AGND. AGND must be connected to GND, and the voltage on AVCC must not differ more than ± 0.4 V from V_{CC}. See the section ADC Noise Canceling Techniques on how to connect these pins.

An external reference voltage must be applied to the AREF pin. This voltage must be in the range 2.7V - AV_{CC}.

Figure 46. Analog to Digital Converter Block Schematic



Operation

The ADC can operate in two modes - Single Conversion and Free Running Mode. In Single Conversion Mode, each conversion will have to be initiated by the user. In Free Running Mode the ADC is constantly sampling and updating the ADC Data Register. The ADFR bit in ADCSR selects between the two available modes.

The ADC is enabled by writing a logical one to the ADC Enable bit, ADEN in ADCSR. The first conversion that is started after enabling the ADC, will be preceded by a dummy conversion to initialize the ADC. To the user, the only difference will be that this conversion takes 27 clock pulses instead of the normal 14.

A conversion is started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit will stay high as long as the conversion is in progress and be set to zero 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.

As the ADC generates a 10-bit result, two data registers, ADCH and ADCL, must be read to get the result when the conversion is complete. Special data protection logic is used to ensure that the contents of the data registers belong to the same result when they are read. This mechanism works as follows:

When reading data, ADCL must be read first. 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, none of the registers are 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 ADCL and ADCH, the interrupt will trigger even if the result gets lost.

Prescaling

The ADC accepts input clock frequencies in the range 50 - 200 kHz. In free running mode, the ADC needs 14 clock pulses to perform a conversion, which means that the conversion time range is 70 - 280 µs. In single conversion mode, the conversion time is 15 clock cycles. The output of the ADC is not guaranteed to be correct if the input clock is out of range. The ADPS0 - ADPS2 bits are used to generate a proper ADC clock input frequency from any XTAL frequency above 100 kHz.

ADC Noise Canceler Function

The ADC features a noise canceler that enables conversion during idle mode to reduce noise induced from the CPU core. To make use of this feature, the following procedure should be used:

1. Turn of the ADC by clearing ADEN.
2. Turn on the ADC and simultaneously start a conversion by setting ADEN and ADSC. This starts a dummy conversion that will be followed by a valid conversion.
3. Within 14 ADC clock cycles, enter idle mode.
4. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the MCU and execute the ADC conversion complete interrupt routine.

The ADC Multiplexer Select Register - ADMUX

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	-	-	-	-	-	MUX2	MUX1	MUX0	ADMUX
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 ATmega603/103 and always read as zero.

- **Bits 2..0 - MUX2..MUX0: Analog Channel Select Bits 2-0**

The value of these three bits selects which analog input 7-0 is connected to the ADC.

The ADC Control and Status Register - ADCSR

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSR
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 a logical ‘1’ to this bit enables the ADC. By clearing this bit 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, a logical ‘1’ must be written to this bit to start each conversion. In free run mode, a logical ‘1’ must be written to this bit to start the first conversion. The first time ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, a dummy conversion will precede the initiated conversion. This dummy 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. When a dummy conversion precedes a real conversion, ADSC will stay high until the real conversion completes.

Writing a 0 to this bit has no effect.

• **Bit 5 - ADFR: ADC Free Run Select**

When this bit is set (one) the ADC operates in Free Running mode. In this mode, the ADC samples and updates the data registers continuously. Clearing this bit (zero) will terminate Free Running mode.

• **Bit 4 -ADIF: ADC Interrupt Flag**

This bit is set (one) 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 (one). 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 ADCSR, a pending interrupt can be disabled. This also applies if the SBI and CBI instructions are used.

• **Bit 3 - ADIE: ADC Interrupt Enable**

When this bit is set (one) and the I-bit in SREG is set (one), the ADC Conversion Complete Interrupt is activated.

• **Bits 2..0 - ADPS2..ADPS0: ADC Prescaler Select Bits**

These bits determine the division factor between the XTAL frequency and the input clock to the ADC.

Table 27. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	Invalid
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

The ADC Data Register - ADCL and ADCH

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	-	-	-	-	-	-	ADC9	ADC8	ADCH
$04 ($24)	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	

When an ADC conversion is complete, the result is found in these two registers. In free-run mode, it is essential that both registers are read, and that ADCL is read before ADCH.

Scanning Multiple Channels

Since change of analog channel always is delayed until a conversion is finished, the free running mode can be used to scan multiple channels without interrupting the converter. Typically, the ADC Conversion Complete interrupt will be used to perform the channel shift. However, the user should take the following fact into consideration:

The interrupt triggers once the result is ready to be read. In free running mode, the next conversion will start one ADC clock cycle after the interrupt triggers. If the ADMUX contents are changed within one ADC clock period after the interrupt triggered, the ADMUX setting will apply to the conversion about to start. If ADMUX is changed later than one ADC clock cycle after the interrupt triggered, the next conversion has already started, and the old setting is used.

ADC Noise Canceling Techniques

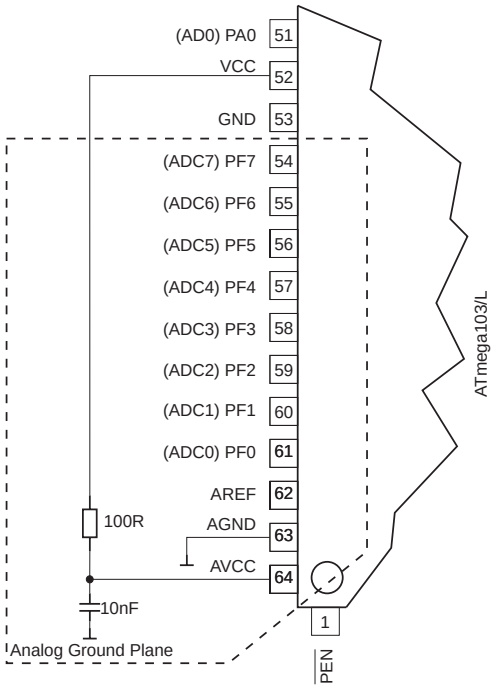
Digital circuitry inside and outside the ATmega603/103 generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

1. The analog part of the ATmega603/103 and all analog components in the application should have a separate analog ground plane on the PCB. This ground plane is connected to the digital ground plane via a single point on the PCB.
2. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground

plane, and keep them well away from high-speed switching digital tracks.

3. The AV_{CC} pin on the ATmega603/103 should be connected to the digital V_{CC} supply voltage via a RC network as shown in Figure 47.
4. Use the ADC noise canceler function to reduce induced noise from the CPU.
5. If some PORTF pins are used as digital inputs, it is essential that these *do not switch* while a conversion is in progress.

Figure 47. ADC Power Connections



ADC DC Characteristics

TA=-40°C to 85°C

Symbol	Parameter	Condition	Min	Typ	Max	Units
	Resolution			10		Bits
	Integral Non-Linearity	VREF > 2V		0.2	0.5	LSB
	Differential Non-Linearity	VREF > 2V		0.2	0.5	LSB
	Zero Error (Offset)			1		LSB
	Conversion Time		70		280	µs
	Clock Frequency		50		200	kHz
AV_{CC}	Analog Supply Voltage		V_{CC}–0.3(1)		V_{CC}+0.3(2)	V
V_{REF}	Reference Voltage		AGND		AV_{CC}	V
R_{REF}	Reference Input Resistance		6	10	13	kΩ
R_{AIN}	Analog Input Resistance			100		MΩ

Notes: 1. Minimum for AV_{CC} is 2.7V.
2. Maximum for AV_{CC} is 6.0V.

I/O-Ports

Port A

PORT A is an 8-bit bi-directional I/O port with internal pull-ups.

Three data memory address locations are allocated for Port A, one each for the Data Register - PORTA, $1B($3B), Data Direction Register - DDRA, $1A($3A) and the Port A Input Pins - PINA, $19($39). The Port A Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The PORT A output buffers can sink 40mA and thus drive LED displays directly. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

The PORT A pins have alternate functions related to the optional external data SRAM. PORT A can be configured to be the multiplexed low-order address/data bus during accesses to the external data memory.

When PORT A is set to the alternate function by the SRE - External SRAM Enable - bit in the MCUCR - MCU Control Register, the alternate settings override the data direction register.

The PORT A Data Register - PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	**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	

The PORT A Data Direction Register - DDRA

Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	**DDA7**	**DDA6**	**DDA5**	**DDA4**	**DDA3**	**DDA2**	**DDA1**	**DDA0**	**DDRA**
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 PORT A Input Pins Address - PINA

Bit	7	6	5	4	3	2	1	0	
$19 ($39)	**PINA7**	**PINA6**	**PINA5**	**PINA4**	**PINA3**	**PINA2**	**PINA1**	**PINA0**	**PINA**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

The Port A Input Pins address - PINA - is not a register, and this address enables access to the physical value on each Port A pin. When reading PORTA the PORTA Data Latch is read, and when reading PINA, the logical values present on the pins are read.

PORTA as General Digital I/O

All 8 bits in PORT A are equal when used as digital I/O pins.

PAn, General I/O pin: The DDAn bit in the DDRA register selects the direction of this pin, if DDAn is set (one), PAn is configured as an output pin. If DDAn is cleared (zero), PAn is configured as an input pin. If PORTAn is set (one) when the pin configured as an input pin, the MOS pull up resistor is activated. To switch the pull up resistor off, PORTAn has to be cleared (zero) or the pin has to be configured as an output pin.

n: 7,6...0, pin number.

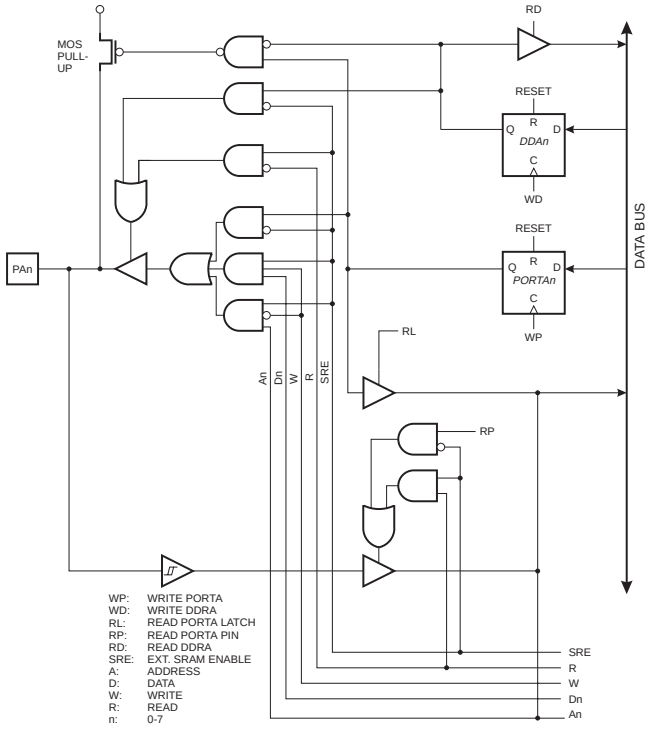
Table 28. DDAn Effects on PORT A Pins

DDAn	PORTAn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PAn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

PORT A Schematics

Note that all port pins are synchronized. The synchronization latch is however, not shown in the figure.

Figure 48. PORTA Schematic Diagrams (Pins PA0 - PA7)



Port B

Port B is an 8-bit bi-directional I/O port with internal pull-ups.

Three data memory address locations are allocated for Port B, one each for the Data Register - PORTB, $18($38), Data Direction Register - DDRB, $17($37) and the Port B Input Pins - PINB, $16($36). The Port B Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The Port B output buffers can sink 40mA and thus drive LED displays directly. When pins PB0 to PB7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

The Port B pins with alternate functions are shown in the following table:

Table 29. Port B Pins Alternate Functions

Port Pin	Alternate Functions
PB0	$\overline{SS}$ (SPI Slave Select input)
PB1	SCK (SPI Bus Serial Clock)
PB2	MOSI (SPI Bus Master Output/Slave Input)
PB3	MISO (SPI Bus Master Input/Slave Output)
PB4	OC0A/PWM0A (Output Compare and PWM Output for Timer/Counter0)
PB5	OC1A/PWM1A (Output Compare and PWM Output A for Timer/Counter1)
PB6	OC1B/PWM1B (Output Compare and PWM Output B for Timer/Counter1)
PB7	OC2/PWM2 (Output Compare and PWM Output for Timer/Counter2

When the pins are used for the alternate function the DDRB and PORTB register have to be set according to the alternate function description.

The Port B Data Register - PORTB

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	**PORTB7**	**PORTB6**	**PORTB5**	**PORTB4**	**PORTB3**	**PORTB2**	**PORTB1**	**PORTB0**	**PORTB**
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 Port B Data Direction Register - DDRB

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	**DDB7**	**DDB6**	**DDB5**	**DDB4**	**DDB3**	**DDB2**	**DDB1**	**DDB0**	**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	

The Port B Input Pins Address - PINB

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	**PINB7**	**PINB6**	**PINB5**	**PINB4**	**PINB3**	**PINB2**	**PINB1**	**PINB0**	**PINB**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

The Port B Input Pins address - PINB - is not a register, and this address enables access to the physical value on each Port B pin. When reading PORTB, the PORTB Data Latch is read, and when reading PINB, the logical values present on the pins are read.

PORTB as General Digital I/O

All 8 bits in port B are equal when used as digital I/O pins.

PBn, General I/O pin: The DDBn bit in the DDRB register selects the direction of this pin, if DDBn is set (one), PBn is configured as an output pin. If DDBn is cleared (zero), PBn is configured as an input pin. If PORTBn is set (one) when the pin configured as an input pin, the MOS pull up resistor is activated. To switch the pull up resistor off, the PORTBn has to be cleared (zero) or the pin has to be configured as an output pin.

Table 30. DDBn Effects on Port B Pins

DDBn	PORTBn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PBn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

n: 7,6...0, pin number.

Alternate Functions OF PORTB

The alternate pin configuration is as follows:

- **OC2/PWM2, Bit 7**

OC2/PWM2, Output Compare output for Timer/Counter2 or PWM output when Timer/Counter2 is in PWM Mode. The pin has to be configured as an output to serve this function.

- **OC1B/PWM1B, Bit 6**

OC1B/PWM1B, Output Compare output B for Timer/Counter1 or PWM output B when Timer/Counter1 is in PWM Mode. The pin has to be configured as an output to serve this function.

- **OC1A/PWM1A, Bit 5**

OC1A/PWM1A, Output Compare output A for Timer/Counter1 or PWM output A when Timer/Counter1 is in PWM Mode. The pin has to be configured as an output to serve this function.

- **OC0/PWM0, Bit 4**

OC0/PWM0, Output Compare output for Timer/Counter0 or PWM output when Timer/Counter0 is in PWM Mode. The pin has to be configured as an output to serve this function.

- **MISO - PORTB, Bit 3**

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 DDB3. When the SPI is enabled as a slave, the data direction of this pin is controlled by DDB3. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB3 bit. See the description of the SPI port for further detatils.

- **MOSI - PORTB, Bit 2**

MOSI: SPI 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 DDB2. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB2. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB2 bit. See the description of the SPI port for further detatils.

- **SCK - PORTB, Bit 1**

SCK: Master clock output, slave clock input pin for SPI channel. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB1. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB1. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB1 bit. See the description of the SPI port for further detatils.

- **$\overline{SS}$ - PORTB, Bit 0**

$\overline{SS}$: Slave port select input. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB0. As a slave, the SPI is activated when this pin is driven low. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB0. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB0 bit. See the description of the SPI port for further detatils.

PORT B Schematics

Note that all port pins are synchronized. The synchronization latches are however, not shown in the figures.

Figure 49. PORTB Schematic Diagram (Pin PB0)

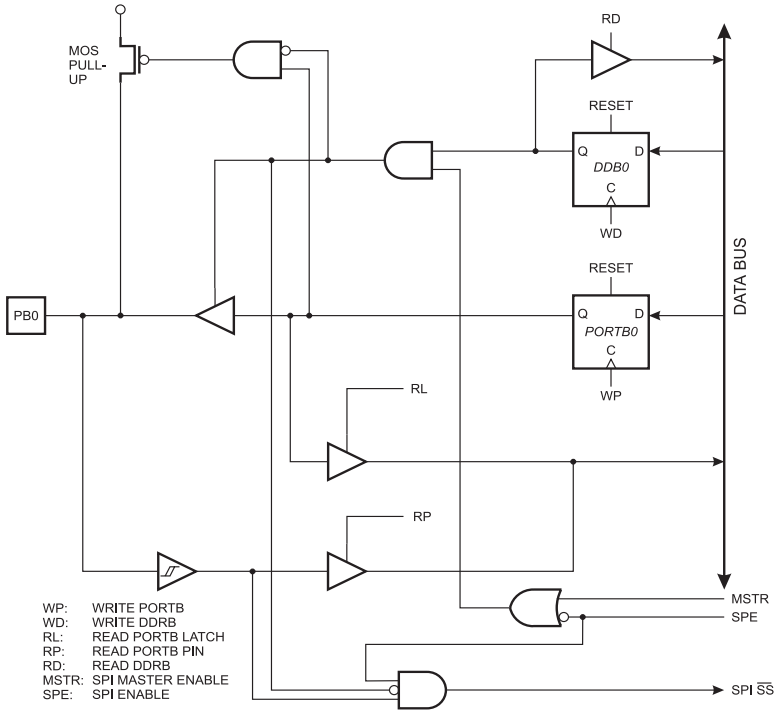


Figure 50. PORTB Schematic Diagram (Pin PB1)

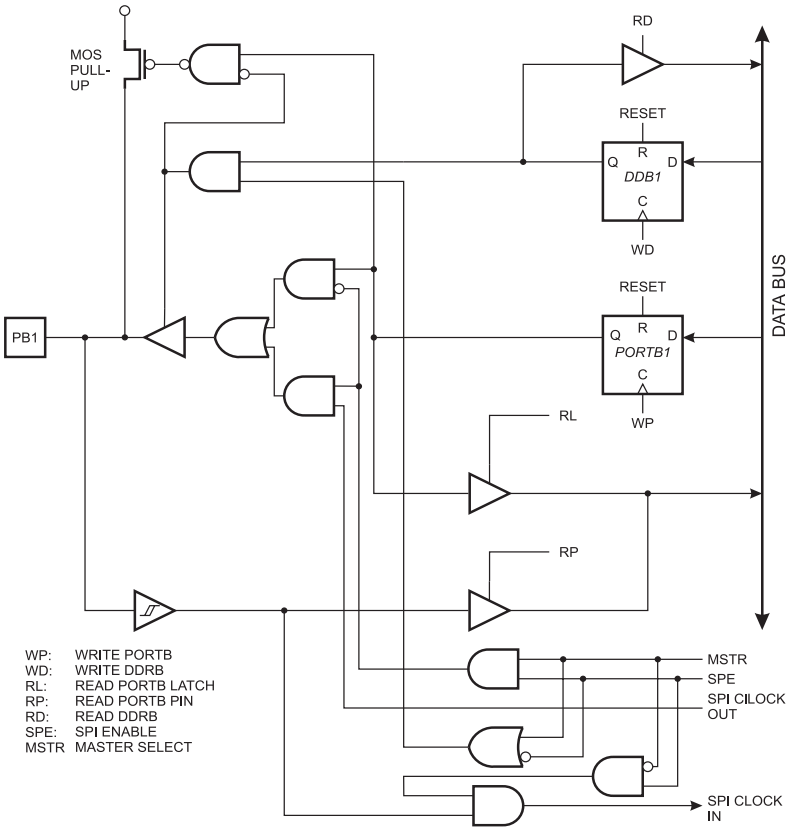


Figure 51. PORTB Schematic Diagram (Pin PB2)

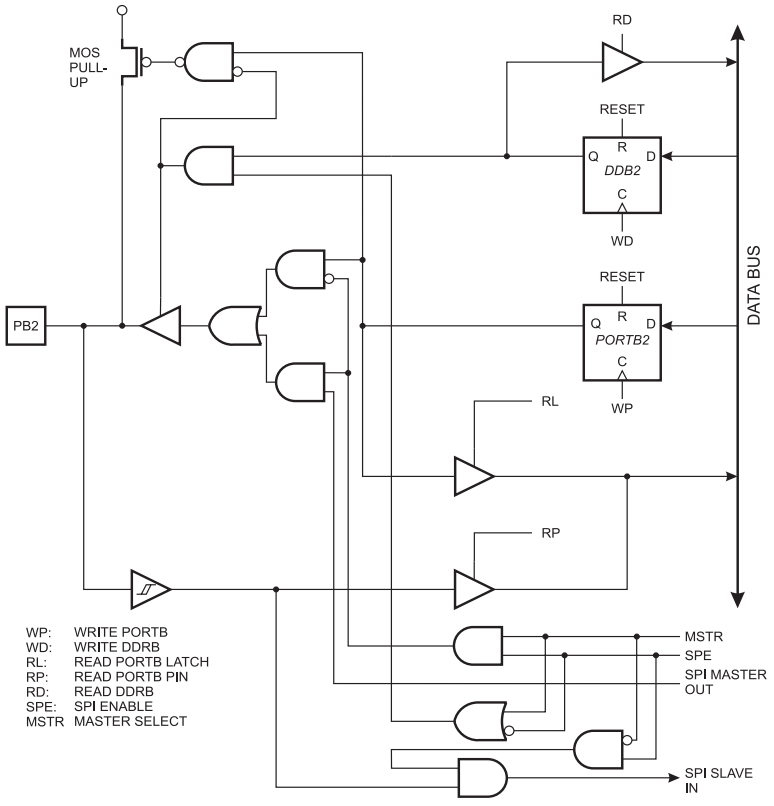


Figure 52. PORTB Schematic Diagram (Pin PB3)

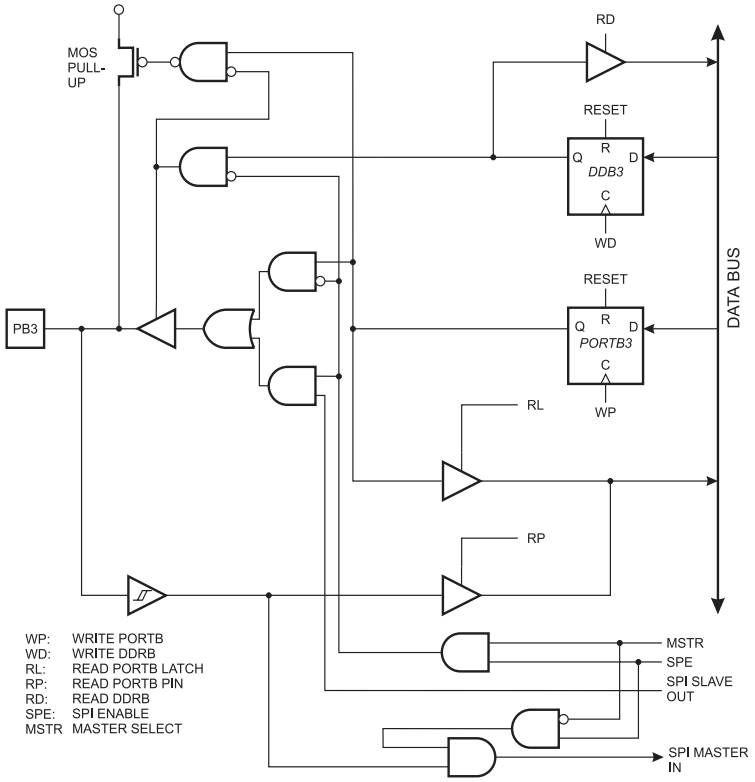


Figure 53. PORTB Schematic Diagram (Pin PB4)

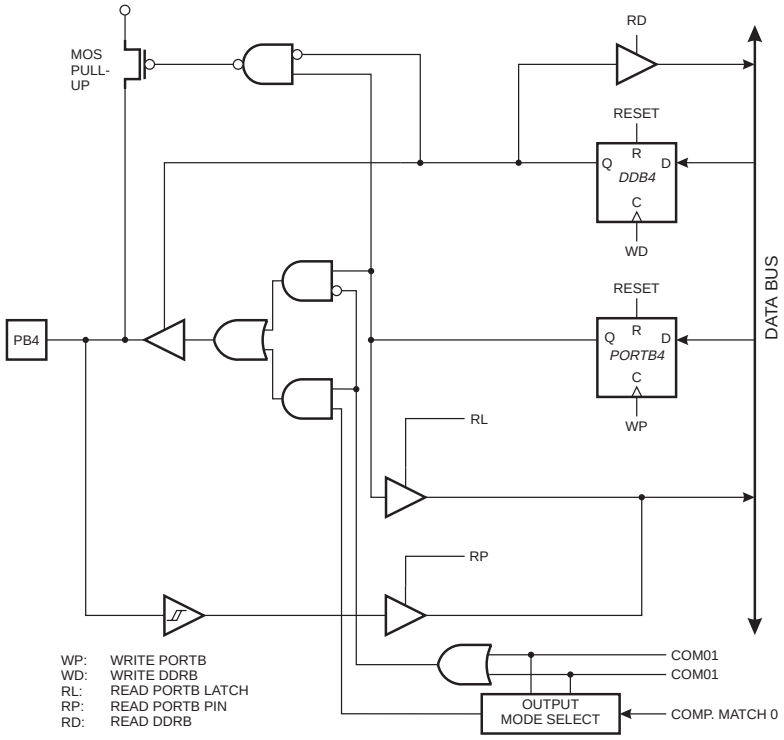


Figure 54. PORTB Schematic Diagram (Pins PB5 and PB6)

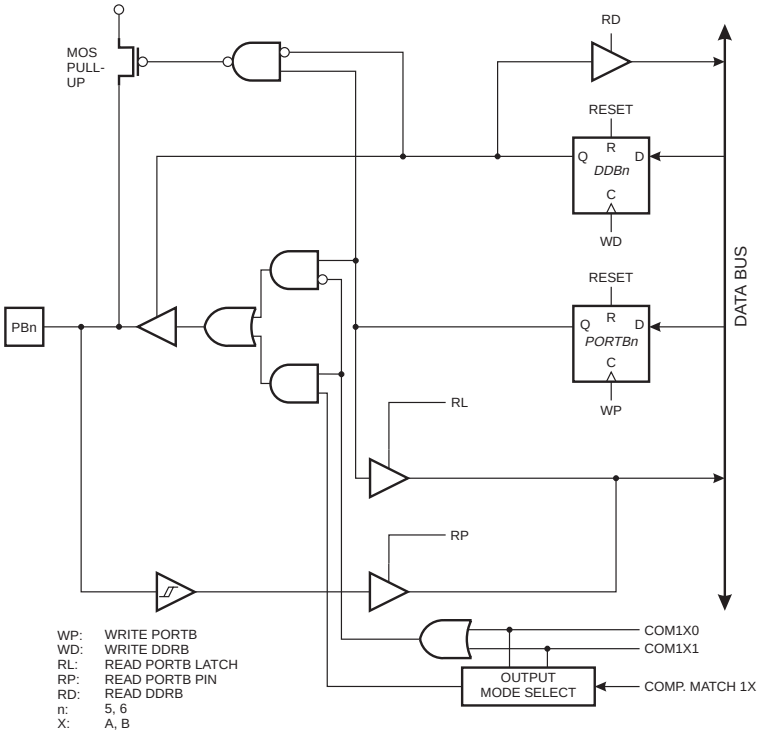
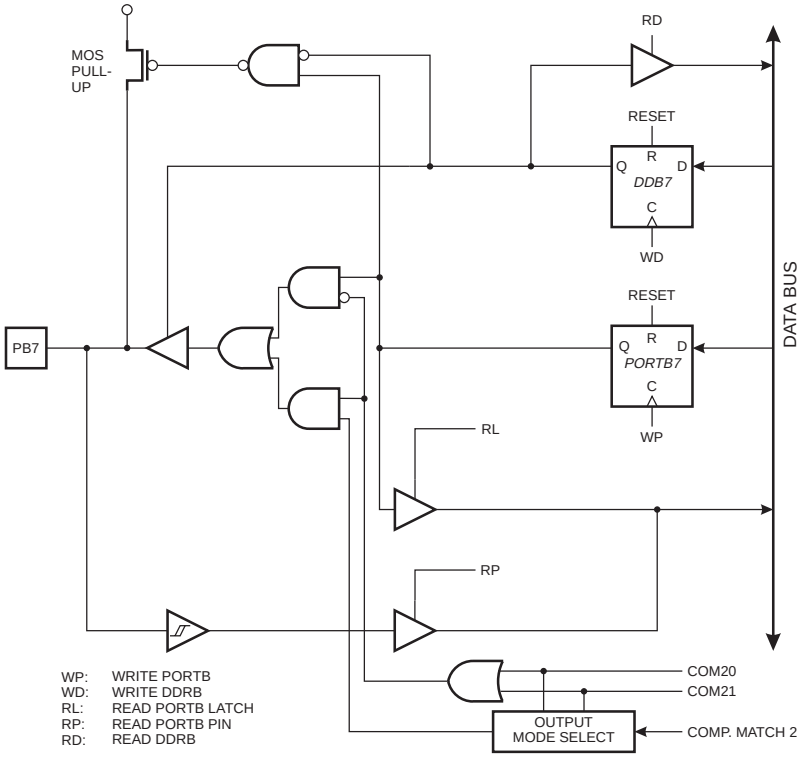


Figure 55. PORTB Schematic Diagram (Pin PB7)



Port C

PORT C is an 8-bit Output port.

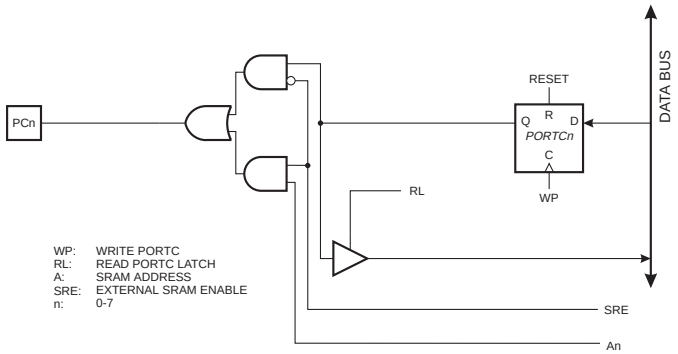
The PORT C pins have alternate functions related to the optional external data SRAM. When using the device with external SRAM, PORT C outputs the high-order address byte during accesses to external data memory.

The PORT C Data Register - PORTC

Bit	7	6	5	4	3	2	1	0	
$15 ($35)	**PORTC7**	**PORTC6**	**PORTC5**	**PORTC4**	**PORTC3**	**PORTC2**	**PORTC1**	**PORTC0**	**PORTC**
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	

PORT C Schematics

Figure 56. PORTC Schematic Diagram (Pins PC0 - PC7)



Port D

Port D is an 8 bit bi-directional I/O port with internal pull-up resistors.

Three data memory address locations are allocated for the Port D, one each for the Data Register - PORTD, $12($32), Data Direction Register - DDRD, $11($31) and the Port D Input Pins - PIND, $10($30). The Port D Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

The Port D output buffers can sink 40 mA. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated.

Some Port D pins have alternate functions as shown in the following table:

Table 31. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD0	$\overline{\text{INT0}}$ (External Interrupt0 Input)
PD1	$\overline{\text{INT1}}$ (External Interrupt1 Input)
PD2	$\overline{\text{INT2}}$ (External Interrupt2 Input)
PD3	$\overline{\text{INT3}}$ (External Interrupt3 Input)
PD4	IC1 (Timer/Counter1 Input Capture Trigger)
PD6	T1 (Timer/Counter1 Clock Input)
PD7	T2 (Timer/Counter2 Clock Input)

When the pins are used for the alternate function the DDRD and PORTD register has to be set according to the alternate function description.

The Port D Data Register - PORTD

Bit	7	6	5	4	3	2	1	0	
$12	**PORTD7**	**PORTD6**	**PORTD5**	**PORTD4**	**PORTD3**	**PORTD2**	**PORTD1**	**PORTD0**	**PORTD**
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 Port D Data Direction Register - DDRD

Bit	7	6	5	4	3	2	1	0	
$11	**DDD7**	**DDD6**	**DDD5**	**DDD4**	**DDD3**	**DDD2**	**DDD1**	**DDD0**	**DDRD**
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 PORT D Input Pins Address - PIND

Bit	7	6	5	4	3	2	1	0	
$10	**PIND7**	**PIND6**	**PIND5**	**PIND4**	**PIND3**	**PIND2**	**PIND1**	**PIND0**	**PIND**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

The Port D Input Pins address - PIND - is not a register, and this address enables access to the physical value on each Port D pin. When reading PORTD, the PORTD Data Latch is read, and when reading PIND, the logical values present on the pins are read.

PORTD as general digital I/O

PDn, General I/O pin: The DDDn bit in the DDRD register selects the direction of this pin. If DDDn is set (one), PDn is configured as an output pin. If DDDn is cleared (zero), PDn is configured as an input pin. If PDn is set (one) when configured as an input pin the MOS pull up resistor is activated. To switch the pull up resistor off the PDn has to be cleared (zero) or the pin has to be configured as an output pin.

Table 32. DDDn Bits on Port D Pins

DDDn	PORTDn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PDn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

n: 7,6...0, pin number.

Alternate Functions of PORTD

$\overline{\text{INT0}}$ .. $\overline{\text{INT3}}$ - PORTD, Bits 0..3

External Interrupt sources 0 - 3. The PD0 - PD3 pins can serve as external active low interrupt sources to the MCU. The internal pull up MOS resistors can be activated as described above. See the interrupt description for further details, and how to enable the sources.

IC1 - PORTD, Bit 4

IC1 - Input Capture pin for Timer/Counter1. When a positive or negative (selectable) edge is applied to this pin, the contents of Timer/Counter1 is transferred to the Timer/Counter1 Input Capture Register. The pin has to be configured as an input (DDD4 is cleared(zero)) to serve this function. See the Timer/Counter1 description on how to operate this function. The internal pull up MOS resistor can be activated as described above.

T1 - PORTD, Bit 6

T1, Timer/Counter1 counter source. See the timer description for further details.

T2 - PORTD, Bit 7

T2, Timer/Counter2 counter source. See the timer description for further details.

PORTD Schematics

Note that all port pins are synchronized. The synchronization latches are however, not shown in the figures.

Figure 57. PORTD Schematic Diagram (Pins PD0, PD1, PD2 and PD3)

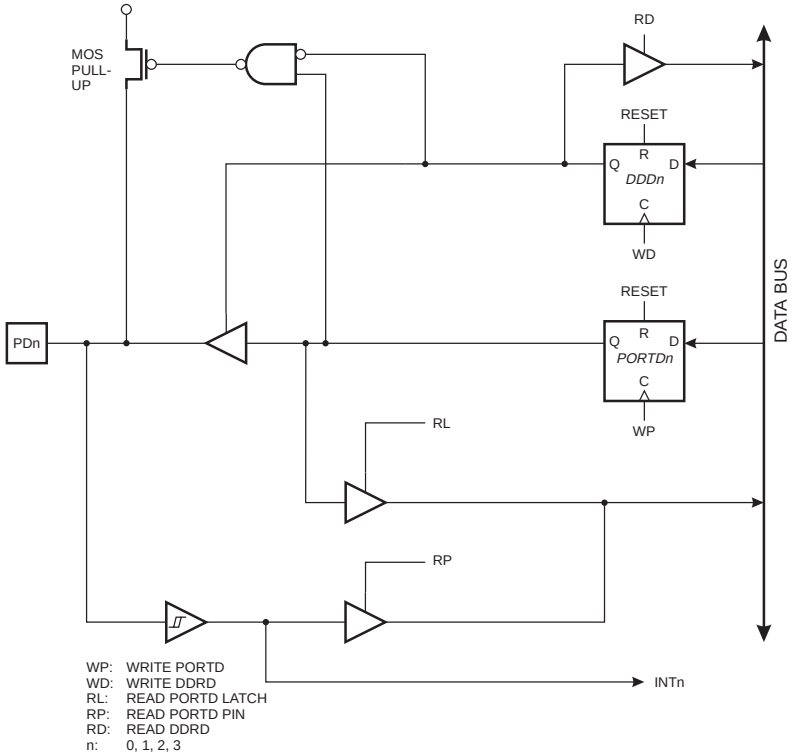


Figure 58. PORTD Schematic Diagram (Pin PD4)

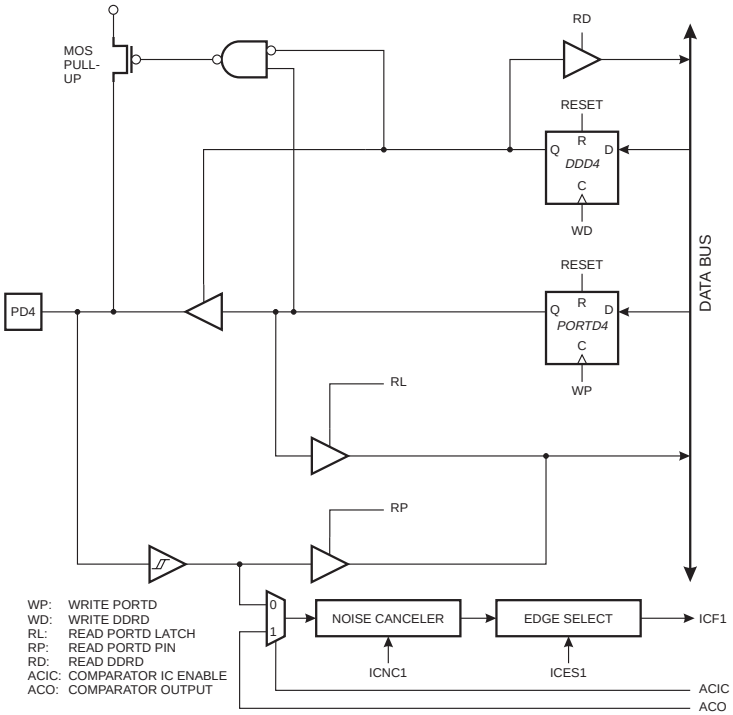


Figure 59. PORTD Schematic Diagram (Pin PD5)

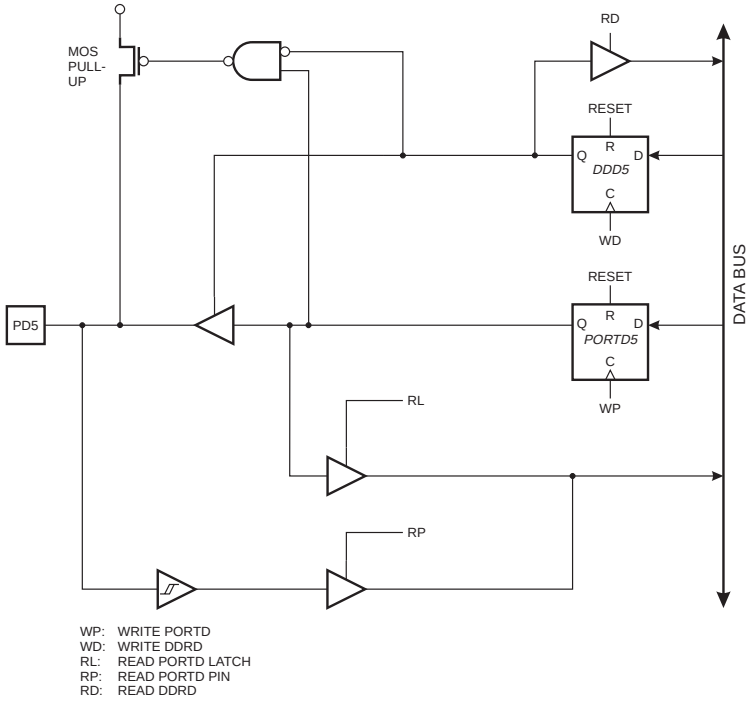
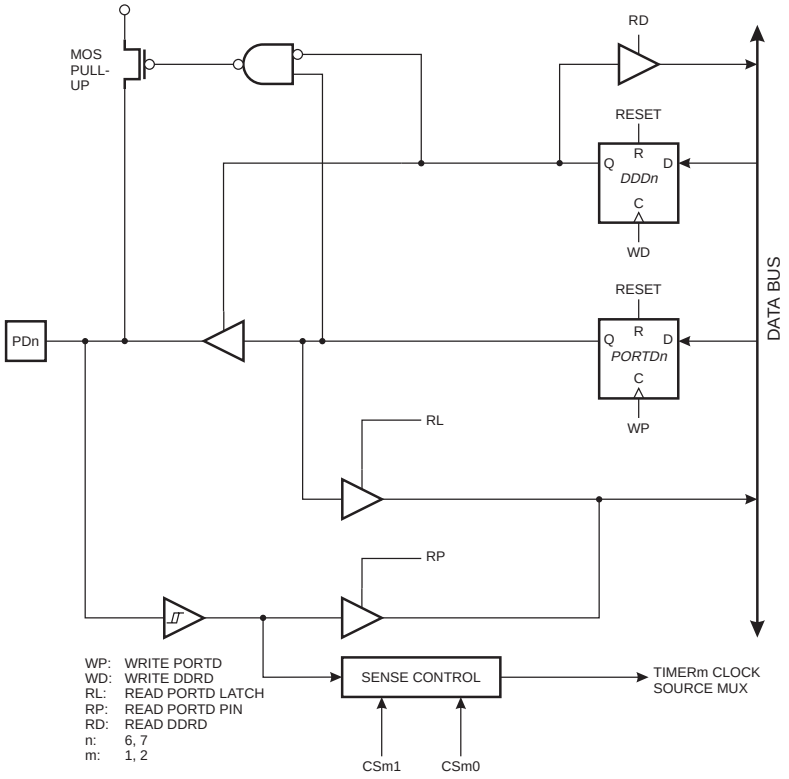


Figure 60. PORTD Schematic Diagram (Pins PD6 and PD7)



Port E

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors.

Three data memory address locations are allocated for the Port E, one each for the Data Register - PORTE, $03($23), Data Direction Register - DDRE, $02($22) and the Port E Input Pins - PINE, $01($21). The Port E Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

The Port E output buffers can sink 40 mA. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated.

All Port E pins have alternate functions as shown in the following table:

Table 33. Port E Pins Alternate Functions

Port Pin	Alternate Function
PE0	PDI/RXD (Programming Data Input or UART Receive Pin)
PE1	PDO/TXD (Programming Data Output or UART Transmit Pin)
PE2	AC+ (Analog Comparator Positive Input)
PE3	AC- (Analog Comparator Negative Input)
PE4	INT4 (External Interrupt4 Input)
PE5	INT5 (External Interrupt5 Input)
PE6	INT6 (External Interrupt6 Input)
PE7	INT7 (External Interrupt7 Input)

When the pins are used for the alternate function the DDRE and PORTE register has to be set according to the alternate function description.

The Port E Data Register - PORTE

Bit	7	6	5	4	3	2	1	0	
$03 ($23)	**PORTE7**	**PORTE6**	**PORTE5**	**PORTE4**	**PORTE3**	**PORTE2**	**PORTE1**	**PORTE0**	**PORTE**
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 Port E Data Direction Register - DDRE

Bit	7	6	5	4	3	2	1	0	
$02 ($22)	**DDE7**	**DDE6**	**DDE5**	**DDE4**	**DDE3**	**DDE2**	**DDE1**	**DDE0**	**DDRE**
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 PORT F Input Pins Address - PINE

Bit	7	6	5	4	3	2	1	0	
$01 ($21)	**PINE7**	**PINE6**	**PINE5**	**PINE4**	**PINE3**	**PINE2**	**PINE1**	**PINE0**	**PINE**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

The Port E Input Pins address - PINE - is not a register, and this address enables access to the physical value on each Port E pin. When reading PORTE, the PORTE Data Latch is read, and when reading PINE, the logical values present on the pins are read.

PORTE as general digital I/O

PDn, General I/O pin: The DDEn bit in the DDRD register selects the direction of this pin. If DDEn is set (one), PEn is configured as an output pin. If DDEn is cleared (zero), PEn is configured as an input pin. If PEn is set (one) when configured as an input pin, the MOS pull up resistor is activated. To switch the pull up resistor off the PEn has to be cleared (zero) or the pin has to be configured as an output pin.

Table 34. DDEn Bits on Port E Pins

DDEn	PORTEn	I/O	Pull up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PDn will source current if ext. pulled low.
1	0	Output	No	Push-Pull Zero Output
1	1	Output	No	Push-Pull One Output

n: 7,6...0, pin number.

Alternate Functions OF PORT E

PDI/RXD - PORTE, Bit 0

PDI, Serial Programming Data Input. During Serial Program Downloading, this pin is used as data input line for the ATmega603/103.

RXD, UART Receive Pin.

PDO/TXD - PORTE, Bit 1

PDO, Serial Programming Data Output. During Serial Program Downloading, this pin is used as data output line for the ATmega603/103.

TXD, UART Transmit Pin.

AC+ - PORTE, Bit 5

AC+ - Analog Comparator Positive Input. This pin is directly connected to the positive input of the analog comparator.

AC- - PORTE, Bit 6

AC- - Analog Comparator Negative Input. This pin is directly connected to the negative input of the analog comparator.

INT4 .. INT7 - PORTD, Bit 7

INT4 .. INT7 - External Interrupt sources 4 - 7: The PE4 - PE7 pins can serve as external interrupt sources to the MCU. Interrupts can be triggered by low level or positive or negative edge on these pins. The internal pull up MOS resistors can be activated as described above. See the interrupt description for further details, and how to enable the sources.

PORTE Schematics

Note that all port pins are synchronized. The synchronization latches are however, not shown in the figures.

Figure 61. PORTE Schematic Diagram, Pin PE0

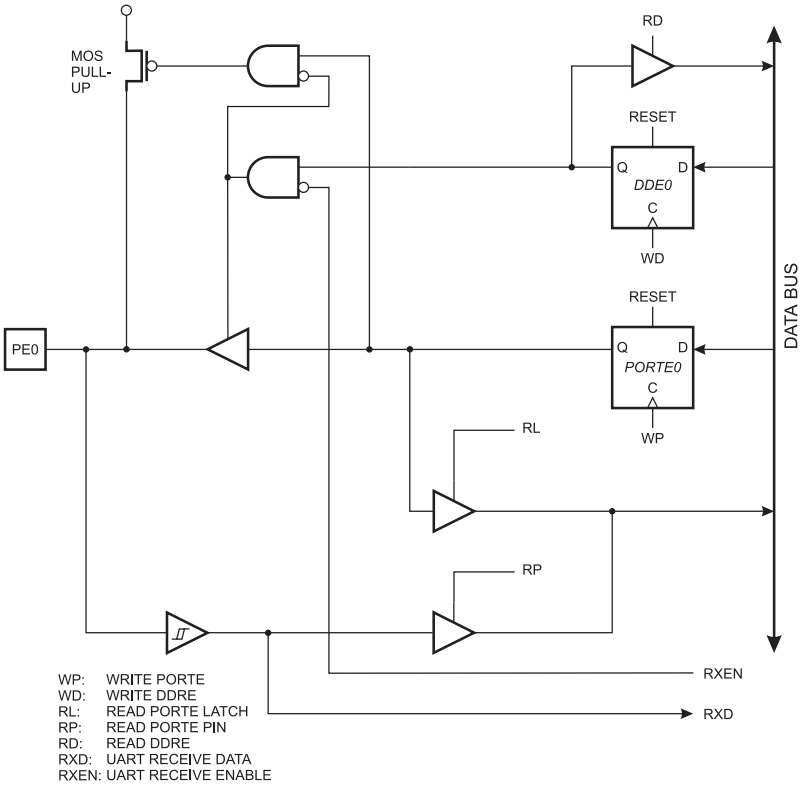


Figure 62. PORTE Schematic Diagram (Pin PE1)

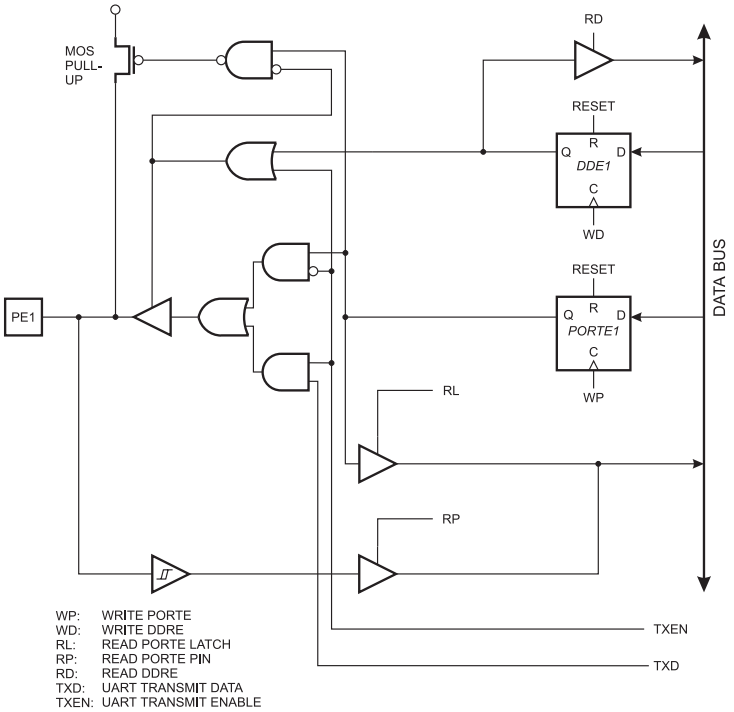
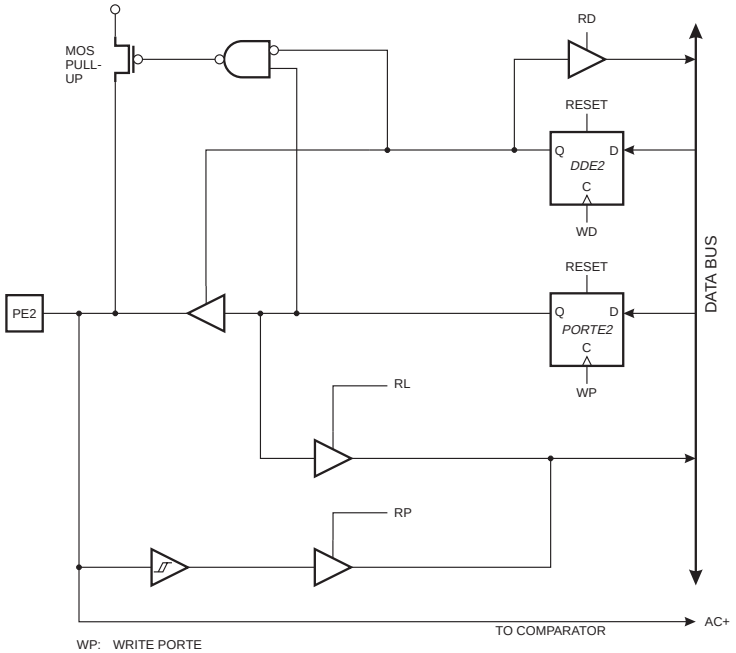
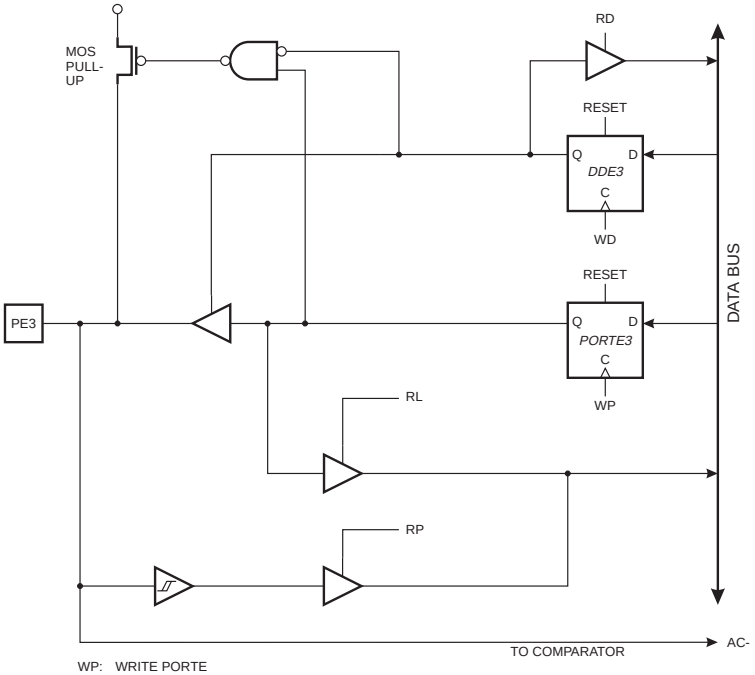


Figure 63. PORTE Schematic Diagram (Pin PE2)



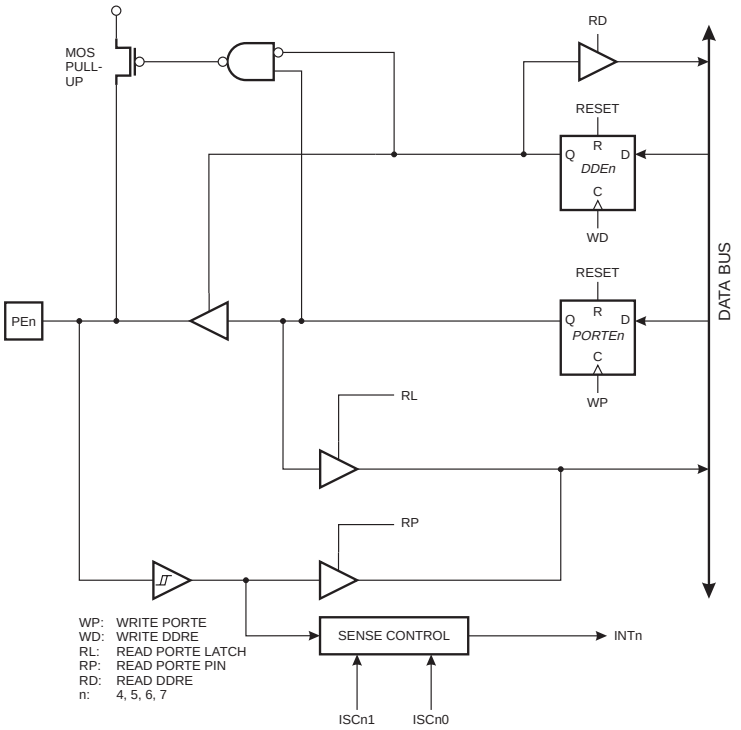
WP: WRITE PORTE
WD: WRITE DDRE
RL: READ PORTE LATCH
RP: READ PORTE PIN
RD: READ DDRE

Figure 64. PORTE Schematic Diagram (Pin PE3)



WP: WRITE PORTE
WD: WRITE DDRE
RL: READ PORTE LATCH
RP: READ PORTE PIN
RD: READ DDRE

Figure 65. PORTE Schematic Diagram (Pins PD4, PD5, PD6 and PD7)



Port F

Port F is an 8-bit input port.

One I/O memory location is allocated for Port F, the Port F Input Pins - PINF, $00($20).

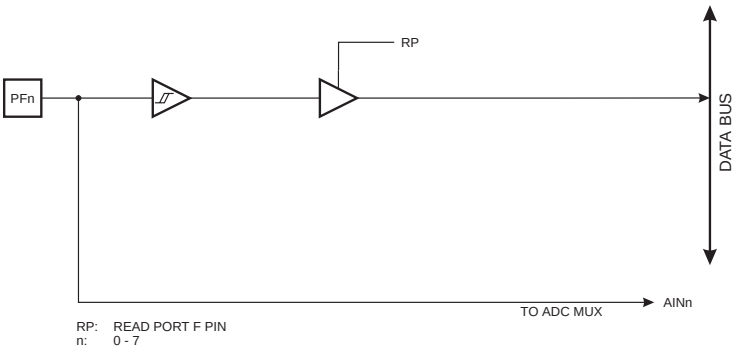
All Port F pins are connected to the analog multiplexer which further is connected to the A/D converter. The digital input function of Port F can be used together with the A/D converter, allowing the user to use some pins of Port F and digital inputs and other as analog inputs, at the same time.

The PORT F Input Pins Address - PINF

Bit	7	6	5	4	3	2	1	0	
$00 ($20)	**PINF7**	**PINF6**	**PINF5**	**PINF4**	**PINF3**	**PINF2**	**PINF1**	**PINF0**	**PINF**
Read/Write	R	R	R	R	R	R	R	R	
Initial value	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	

The Port F Input Pins address - PINF - is not a register, and this address enables access to the physical value on each Port F pin.

Figure 66. PORTF Schematic Diagram (Pins PF7 - PF0)



Memory Programming

Program Memory Lock Bits

The ATmega603/103 MCU provides two lock bits which can be left unprogrammed (‘1’) or can be programmed (‘0’) to obtain the additional features listed in Table 35.

Table 35. Lock Bit Protection Modes

Program Lock Bits			Protection Type
Mode	LB1	LB2	
1	1	1	No program lock features
2	0	1	Further programming of the Flash and EEPROM is disabled
3	0	0	Same as mode 2, but verify is also disabled.

Note: The Lock Bits can only be erased to ‘1’ with the Chip Erase operation.

Fuse Bits

The ATmega603/103 has four fuse bits, SPIEN, SUT1, SUT0 and EESAVE. When SPIEN is programmed (‘0’), Serial Program Downloading is enabled. Default value is programmed (‘0’). This bit is not accessible in Serial Programming Mode and is not changed by a chip erase.

The SUT1 and SUT0 fuses determine the MCU start-up time. See Table 5 for further details. Default value is unprogrammed (‘11’), which gives a nominal start up time of 16 ms.

When EESAVE is programmed, EEPROM memory is preserved through the Chip Erase cycle. Default value is unprogrammed (‘1’). The EESAVE Fuse bit cannot be programmed if any of the Lock bits are programmed.

Signature Bytes

All Atmel microcontrollers have a three-byte signature code which identifies the device. This code can be read in both serial and parallel mode. The three bytes reside in a separate address space. For the ATmega603 they are:

1. $00: $1E (indicates manufactured by Atmel)
2. $01: $06 (indicates 64 kB Flash memory)
3. $02: $01 (indicates ATmega603 when $01 is $06)

For the ATmega103 they are:

1. $00: $1E (indicates manufactured by Atmel)
2. $01: $01 (indicates 128 kB Flash memory)
3. $02: $01 (indicates ATmega103 when $01 is $01)

Programming the Flash and EEPROM

Atmel’s ATmega603/103 offers 64K/128K bytes of in-system reprogrammable Flash memory and 2K/4K bytes of EEPROM Data memory. The ATmega603/103 is normally shipped with the on-chip Flash Program and EEPROM Data memory arrays in the erased state (i.e. contents = $FF) and ready to be programmed. This device supports a High-Voltage (12V) Parallel programming mode and a Low-Voltage Serial programming mode. The +12V supplied is used for programming enable only, and no current of significance is drawn by this pin. The serial programming mode provides a convenient way to download the Program and Data into the ATmega603/103 inside the user’s system. The Program memory array on the ATmega603/103 is organized as 256/512 pages of 256 bytes each. When programming the Flash, the program data is latched into a page buffer. This allows one page of program data to be programmedsimultaneously in either programming mode. The EEPROM Data memory array on the ATmega603/103 is programmed byte-by-byte in either programming mode. An auto-erase cycle is provided with the self-timed EEPROM programming operation in the serial programming mode.

Parallel Programming

This section describes how to parallel program and verify Flash Program memory, EEPROM Data memory, Program Memory Lock bits, and Fuse bits in the ATmega603/103. Pulses are assumed to be at least 500 ns unless otherwise noted.

Signal Names

In this section, some pins of the ATmega603/103 are referenced by signal names describing their functionality during parallel programming rather than their pin names. Pins not described in the following table are referenced by pin names.

Table 36. Pin Name Mapping

Signal Name in Programming Mode	Pin Name	I/O	Function
RDY / $\overline{\text{BSY}}$	PD1	O	0: Device is busy programming, 1: Device is ready for new command
$\overline{\text{OE}}$	PD2	I	Output Enable (Active Low)
$\overline{\text{WR}}$	PD3	I	Write Pulse (Active Low)
BS1	PD4	I	Byte Select Bit 0
XA0	PD5	I	XTAL Action Bit 0
XA1	PD6	I	XTAL Action Bit 1
BS2	PD7	I	Byte Select Bit 2 (Always Low)
PAGEL	PA0	I	Program Memory Page Load

The XA1/XA0 bits determine the action taken when the XTAL1 pin is given a positive pulse. The bit settings are shown in the following table:

Table 37. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash or EEPROM Address (High or Low address byte for Flash determined by BS1)
0	1	Load Data (High or Low data byte for Flash determined by BS1)
1	0	Load Command
1	1	No Action, Idle

When pulsing $\overline{\text{WR}}$ or $\overline{\text{OE}}$, the command loaded determines the action on input or output. The command is a byte where the different bits are assigned functions as shown in the following table:

Table 38. Command Byte Bit Coding

Bit#	Meaning when Set
7	Chip Erase
6	Write Fuse Bits. Located in the data byte at the following bit positions: D5: SPIEN Fuse, D3: EESAVE Fuse, D1: SUT1 Fuse, D0: SUT0 Fuse (Note: Write ‘0’ to program, ‘1’ to erase)
5	Write Lock Bits. Located in the data byte at the following bit positions: D2: LB2, D1: LB1 (Note: write ‘0’ to program)
4	Write Flash or EEPROM (determined by bit 0)
3	Read signature row
2	Read Lock and Fuse Bits. Located in the data byte at the following bits positions: D5: SPIEN Fuse, D3: EESAVE Fuse, D2: LB2, D1: SUT1/LB1, D0: SUT0 (Note: ‘0’ means programmed)
1	Read from Flash or EEPROM (determined by bit 0)
0	0 : Flash Access, 1 : EEPROM Access

Enter Programming Mode

The following algorithm puts the device in parallel programming mode:

1. Apply 4.5 - 5.5V between VCC and GND.
2. Set $\overline{\text{RESET}}$ and BS pins to '0' and wait at least 100 ns.
3. Apply 11.5 - 12.5V to $\overline{\text{RESET}}$. Any activity on BS1 within 100 ns after +12V has been applied to $\overline{\text{RESET}}$, will cause the device to fail entering programming mode.

Chip Erase

The chip erase will erase the Flash and EEPROM memories and 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 is programmed.

Load Command "Chip Erase"

1. Set XA1, XA0 to '10'. This enables command loading.
2. Set BS1 to '0'.
3. Set PB(7:0) to '1000 0000'. This is the command for Chip erase.
4. Give XTAL1 a positive pulse. This loads the command, and starts the erase of the Flash and EEPROM arrays. After pulsing XTAL1, give $\overline{\text{WR}}$ a negative pulse to enable lock bit erase at the end of the erase cycle, then wait for at least 10 ms. Chip erase does not generate any activity on the RDY/$\overline{\text{BSY}}$ pin.

Programming The Flash

The Flash is organized as 256/512 pages of 256 bytes each. 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:

A: Load Command "Program Flash"

1. Set XA1, XA0 to '10'. This enables command loading.
2. Set BS1 to '0'
3. Set PB(7:0) to '0001 0000'. This is the command for Flash programming.
4. Give XTAL1 a positive pulse. This loads the command.

B: Load Address Low byte

1. Set XA1, XA0 to '00'. This enables address loading.
2. Set BS1 to '0'. This selects Low address.
3. Set PB(7:0) = Address Low byte ($00 - $FF)
4. Give XTAL1 a positive pulse. This loads the Address Low byte.

C: Load Data byte

1. Set XA1, XA0 to '01'. This enables data loading.
2. Set PB(7:0) = Data Low byte ($00 - $FF)
3. Give XTAL1 a positive pulse. This loads the data byte.

D: Latch Data Low byte

1. Set BS1 to ('0'). This selects low data byte.
2. Give PAGEL a positive pulse, This latches the data byte.

E: Load Data byte

1. Set XA1, XA0 to '01'. This enables data loading.
2. Set PB(7:0) = Data High byte ($00 - $FF)
3. Give XTAL1 a positive pulse. This loads the Data byte.

F: Latch Data High byte

1. Set BS1 to '1'. This selects high data byte.
2. Give PAGEL a positive pulse. This latches the data byte.

G: Repeat B through F 128 times to fill the page buffer

H: Load Adddress High byte

1. Set XA1, XA0 to '00'. This enables address loading.
2. Set BS1 to '1'. This selects high address.
3. Set PB(7:0) = Address high byte (Atmega603: $7F, Atmega103: $FF).
4. Give XTAL1 a positive pulse. This loads the Address High byte.

I: Program Page

1. Give $\overline{\text{WR}}$ a negative pulse. This starts programming of the entire page of data. RDY/$\overline{\text{BSY}}$ goes low.
2. Wait until RDY/$\overline{\text{BSY}}$ goes high.

J: End Page Programming

1. Set XA1, XA0 to '10'. This enables command loading.
2. Set PB(7:0) = '0000 0000'. This is the command for No Operation.
3. Give XTAL1 a positive pulse. This loads the command and the internal write signals are reset.

K: Repeat A through J 256/512 times or until all data have been programmed

Programming The Eeprom

The programming algorithm for the EEPROM data memory is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0001 0001'.
2. Load Low EEPROM Address ($00 - $FF)

3. Load High EEPROM Address (ATmega603: $07, ATmega103: $0F)
4. Load Low EEPROM Data ($00 - $FF)
5. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

The Command needs only be loaded before programming the first byte.

Reading The Flash

The algorithm for reading the Flash memory is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0000 0010'.
2. Load Low Address ($00 - $FF)
3. Load High Address (Atmega603: $7F, Atmega103: $FF)
4. Set $\overline{OE}$ to '0', and BS1 to '0'. The Low Data byte can now be read at PB(7:0)
5. Set BS to '1'. The High Data byte can now be read from PB(7:0)
6. Set $\overline{OE}$ to '1'.

The Command needs only be loaded before reading the first byte.

Reading The EEPROM

The algorithm for reading the EEPROM memory is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0000 0011'.
2. Load Low EEPROM Address ($00 - $FF)
3. Load High EEPROM Address (ATmega603: $07, ATmega103: $0F)
4. Set $\overline{OE}$ to '0', and BS1 to '0'. The EEPROM Data byte can now be read at PB(7:0)
5. Set $\overline{OE}$ to '1'.

The Command needs only be loaded before reading the first byte.

Programming The Fuse Bits

The algorithm for programming the Fuse bits is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0100 0000'.
2. Load Data.
   Bit 5 = '0' programs the SPIEN Fuse bit. Bit 5 = '1' erases the SPIEN Fuse bit.
   Bit 3 = '0' programs the EESAVE Fuse bit. Bit 3 = '1' erases the EESAVE Fuse bit.
   Bit 2 = always '1'
   Bit 1 = '0' programs the SUT1 Fuse bit. Bit 1 = '1' erases the SUT1 Fuse bit.
   Bit 0 = '0' programs the SUT0 Fuse bit. Bit 5 = '1' erases the SUT0 fuse bit.
3. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

Programming The Lock Bits

The algorithm for programming the Lock bits is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0010 0000'.
2. Load Data.
   Bit 2 = '0' programs Lock Bit2
   Bit 1 = '0' programs Lock Bit1
3. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

The lock bits can only be cleared by executing a chip erase.

Reading The Fuse And Lock Bits

The algorithm for reading the Fuse and Lock bits is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0000 0100'.
2. Set $\overline{OE}$ to '0', and BS1 to '0' or '1'. The Status of Fuse and Lock bits can now be read at PB(7:0)
   BS1 = '0':
   Bit 5: SPIEN Fuse ('0' means programmed, '1' means erased)
   Bit 3: EESAVE Fuse ('0' means programmed, '1' means erased)
   Bit 1: SUT1 Fuse ('0' means programmed, '1' means erased)
   Bit 0: SUT0 Fuse ('0' means programmed, '1' means erased)
   BS1 = '1':
   Bit 2: LB1 ('0' means programmed, '1' means erased)
   Bit 1: LB0 ('0' means programmed, '1' means erased)
3. Set $\overline{OE}$ to '1'.

Reading The Signature Bytes

The algorithm for reading the Signature Bytes bits is as follows (refer to Flash Programming for details on Command, Address and Data loading):

1. Load Command '0000 1000'.
2. Load Low address ($00 - $02)
3. Set $\overline{OE}$ to '0', and BS to '0'. The Selected Signature byte can now be read at PB(7:0)
4. Set $\overline{OE}$ to '1'.

The command needs only be programmed before reading the first byte.

Serial Downloading

Both the Flash and EEPROM memory arrays can be programmed using the serial SPI bus while $\overline{\text{RESET}}$ is pulled to GND. The serial interface consists of pins SCK, RXD/PDI (input) and TXD/PDO (output). After $\overline{\text{RESET}}$ is set low, the Programming Enable instruction needs to be executed first before program/erase operations can be executed.

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 $FF.

The Program and EEPROM memory arrays have separate address spaces:

ATmega603: $0000 to $7FFF for program memory and $0000 to $07FF for EEPROM memory.

ATmega103: $0000 to $FFFF for program memory and $0000 to $0FFF for EEPROM memory.

Either an external system clock is supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2.The minimum low and high periods for the serial clock (SCK) input are defined as follows:

Low: > 2 XTAL1 clock cycles

High: > 2 XTAL1 clock cycles

SERIAL PROGRAMMING ALGORITHM

To program and verify the ATmega103/L in the serial programming mode, the following sequence is recommended (See 4-byte instruction formats in Table 39.):

1. *Power-up sequence:*

   Apply power between VCC and GND while $\overline{\text{RESET}}$ and SCK are set to ‘0’. If a crystal is not connected across pins XTAL1 and XTAL2, apply a clock signal to the XTAL1 pin. 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 of at least two XTAL1 cycles duration after SCK has been set to ‘0’.
2. Wait for at least 20 ms and enable serial programming by sending the Programming Enable serial instruction to pin PE0(PDI/RXD).
3. When issuing the third byte in Programming Enable, the value sent as byte number two ($53), will echo back during transmission of byte number three. In any case, all four bytes in programming enable must be transmitted. If the $53 did not echo back, give SCK a positive pulse and issue a new Programming Enable command. If the $53 is not seen within 32 attempts, there is no functional device connected.
4. If a chip erase is performed (must be done to erase the Flash), wait 10 ms, give $\overline{\text{RESET}}$ a positive pulse, and start over from Step 2.
5. The Flash is programmed one page at a time. The memory page is loaded one byte at a time by supplying the 7 LSB of the address and data together with the Load Program Memory Page instruction. The Program Memory Page is stored by loading the Write Program Memory Page instruction with the 9 MSB of the address.
6. 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. The next byte can be written after 4 ms (2.7V) 1 ms (5.0V).
7. Any memory location can be verified by using the Read instruction which returns the content at the selected address at serial output PE1(PDO/TXD).
8. At the end of the programming session, $\overline{\text{RESET}}$ can be set high to commence normal operation.
9. Power-off sequence (if needed):

   Set XTAL1 to ‘0’ (if a crystal is not used).

   Set $\overline{\text{RESET}}$ to ‘1’.

   Turn V_{CC} power off

Table 39. Serial Programming Instruction Set

Instruction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte 4	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx	Enable Serial Programming after RESET goes low.
Chip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip erase EEPROM and Flash
Read Program Memory	0010 **H**000	**aaaa aaaa**	**bbbb bbbb**	**oooo oooo**	Read **H**(high or low) data **o** from Program memory at word address **a:b**
Load Program Memory Page	0100 **H**000	xxxx xxxx	**xbbb bbbb**	**iiii iiii**	Write **H**(high or low) data **i** to Program memory at word address **b**
Write Program Memory Page	0100 1100	aaaa aaaa	bxxx xxxx	xxxx xxxx	Write Program Memory Page at address a:b.
Read EEPROM Memory	1010 0000	xxxx **aaaa**	**bbbb bbbb**	**oooo oooo**	Read data **o** from EEPROM memory at address **a:b**
Write EEPROM Memory	1100 0000	xxxx **aaaa**	**bbbb bbbb**	**iiii iiii**	Write data **i** to EEPROM memory at address **a:b**
Read Lock Bits	0101 1000	xxxx xxxx	xxxx xxxx	xxxx x***21***x	Read lock bits. '0': Programmed, '1': Unprogrammed
Write Lock Bits	1010 1100	111x x***21***x	xxxx xxxx	xxxx xxxx	Write lock bits. Set bits ***1,2*** = '0' to program lock bits.
Read Fuse Bits	0101 0000	xxxx xxxx	xxxx xxxx	xx***5***x ***6***1***43***	Read fuse bits. '0': Programmed, '1': Unprogrammed
Write Fuse Bits	1010 1100	101x ***6***1***43***	xxxx xxxx	xxxx xxxx	Write fuse bits. Set bit ***6***,***4,3*** = '0' to program fuse, '1' to unprogram
Read Signature Byte	0011 0000	xxxx xxxx	xxxx xx**bb**	**oooo oooo**	Read Signatrue Byte **o** at address **b**

Notes: a = address high bits

b = address low bits

H = 0 - Low byte, 1 - High Byte

o = data out

i = data in

x = don't care

1 = lock bit 1

2 = lock bit 2

3 = SUT0 fuse

4 = SUT1 fuse

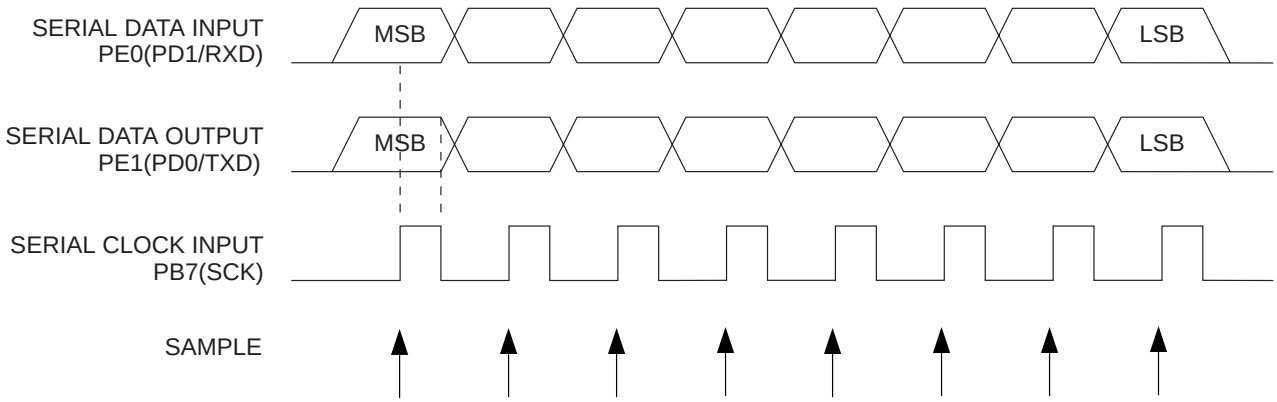
5 = SPIEN fuse

6 = EESAVE fuse

When writing serial data to the ATmega603/103, data is sampled by the ATmega 103 on the rising edge of SCK.

When reading data from the ATmega603/103, data is clocked on the falling edge of SCK. See Figure 67. for an explanation.

Figure 67. Serial Programming Waveforms



Absolute Maximum Ratings*

Operating Temperature	-40°C to +105°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin with respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current.	300 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.

DC Characteristics

T_A = -40°C to 85°C V_{CC} = 2.7V to 6.0V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage		-0.5		0.3 V_{CC}	V
V_{IL1}	Input Low Voltage	XTAL	-0.5		0.2 V_{CC}	V
V_{IH}	Input High Voltage	Except (XTAL, RESET)	0.6 V_{CC}		V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	XTAL	0.8 V_{CC}		V_{CC} + 0.5	V
V_{IH2}	Input High Voltage	RESET	V_{CC}		V_{CC} + 0.5	V
V_{OL}	Output Low Voltage(1) Ports A, B, C, D	I_{OL} = 20 mA, V_{CC} = 5V I_{OL} = 10 mA, V_{CC} = 3V			0.6 0.5	V V
V_{OH}	Output High Voltage Ports A, B, C, D	I_{OH} = -3 mA, V_{CC} = 5V I_{OH} = -1.5 mA, V_{CC} = 3V	4.2 2.3			V V
I_{IL}	Input Leakage Current I/O Pin	V_{CC} = 6V, Pin Low	-8.0		8.0	µA
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 6V, Pin High	-8.0		8.0	µA
RRST	Reset Pullup		100		500	kΩ
$R_{I/O}$	I/O Pin Pullup		35		120	kΩ
I_{CC}	Power Supply Current	Active 4 MHz, 3 V_{CC}(2)			3.0	mA
		Idle 4 MHz, 3 V_{CC}		1.0	1.2	mA
		Power Down 4 MHz, 3 V_{CC} WDT Enabled		8.5	15.0	µA
		Power Down 4 MHz, 3 V_{CC} WDT Disabled		<1	2.0	µA
V_{ACIO}	Analog Comp Input Offset V	V_{CC} = 5V			40	mV
I_{ACLK}	Analog Comp Input Leakage A	V_{CC} = 5V V_{IN} = V_{CC}/2	-50		50	nA
t_{ACPD}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

Notes:
1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 10 mA
   Maximum total I_{OL} for all output pins: 300 mA
   Port A: 26 mA
   Ports A, B, D: 15 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification.
   Pins are not guaranteed to sink current greater than the listed test conditions.
2. With 4 MHz XTAL, Bus frequency is also 4 MHz.
3. Minimum V_{CC} for Power Down is 2V.

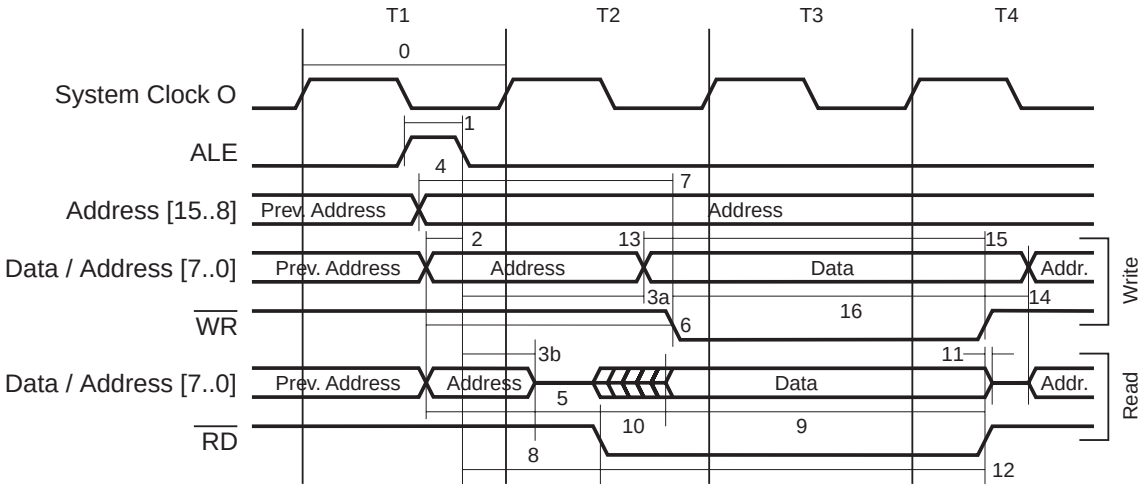
Table 40. External Data Memory Characteristics, 4.0 - 6.0 Volts, No Wait State

			8 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	8.0	MHz
1	t_{LHLL}	ALE Pulse Width	32.5		$0.5t_{CLCL}$-30.0		ns
2	t_{AVLL}	Address Valid A to ALE Low	22.5		$0.5t_{CLCL}$-40.0		ns
3a	t_{LLAX_ST}	Address Hold After ALE Low, ST/STD/STS Instructions	67.5		$0.5t_{CLCL}$-5.0		ns
3b	t_{LLAX_LD}	Address Hold after ALE Low, LD/LDD/LDS Instructions	15.0		15.0		ns
4	t_{AVLLC}	Address Valid C to ALE Low	22.5		$0.5t_{CLCL}$-40.0		ns
5	t_{AVRL}	Address Valid to RD Low	95.0		$1.0t_{CLCL}$-30.0		ns
6	t_{AVWL}	Address Valid to WR Low	157.5		$1.5t_{CLCL}$-30.0		ns
7	t_{LLWL}	ALE Low to WR Low	105.0	145	$1.0t_{CLCL}$-20.0	$1.0t_{CLCL}$+20.0	ns
8	t_{LLRL}	ALE Low to RD Low	42.5	82.5	$0.5t_{CLCL}$-20.0	$0.5t_{CLCL}$+20.0	ns
9	t_{DVRH}	Data Setup to RD High	60.0		60.0		ns
10	t_{RLDV}	Read Low to Data Valid		70.0		1.0_{CLCL}-55.0	ns
11	t_{RHDX}	Data Hold After RD High	0.0		0.0		ns
12	t_{RLRH}	RD Pulse Width	105.0		$1.0t_{CLCL}$-20.0		ns
13	t_{DVWL}	Data Setup to WR Low	27.5		$0.5t_{CLCL}$-35.0		ns
14	t_{WHDX}	Data Hold After WR High	0.0		0.0		ns
15	t_{DVWH}	Data Valid to WR High	95.0		$1.0t_{CLCL}$-30.0		ns
16	t_{WLWH}	WR Pulse Width	42.5		$0.5t_{CLCL}$-20.0		ns

Table 41. External Data Memory Characteristics, 4.0 - 6.0 Volts, 1 Cycle Wait State

			8 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	8.0	MHz
10	t_{RLDV}	Read Low to Data Valid		195.0		$2.0t_{CLCL}$-55.0	ns
12	t_{RLRH}	RD Pulse Width	230.0		$2.0t_{CLCL}$-20.0		ns
15	t_{DVWH}	Data Valid to WR High	220.0		$2.0t_{CLCL}$-30.0		ns
16	t_{WLWH}	WR Pulse Width	167.5		$1.5t_{CLCL}$-20.0		ns

Figure 68. External RAM Timing



Note: Clock cycle T3 is only present when external SRAM waitstate is enabled

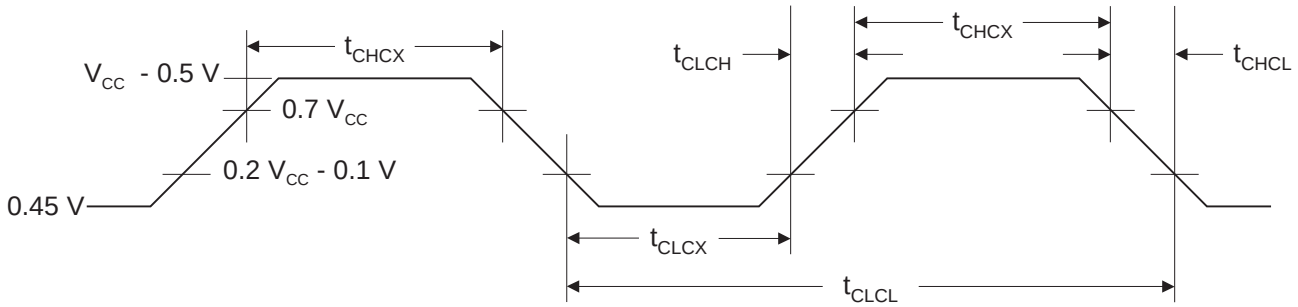
Table 42. External Data Memory Characteristics, 2.7 - 6.0 Volts, No Wait State

			8 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	4.0	MHz
1	t_{LHLL}	ALE Pulse Width	70.0		$0.5t_{CLCL}$-55.0		ns
2	t_{AVLL}	Address Valid A to ALE Low	60.0		$0.5t_{CLCL}$-65.0		ns
3a	t_{LLAX_ST}	Address Hold After ALE Low, ST/STD/STS Instructions	130.0		$0.5t_{CLCL}$-5.0		ns
3b	t_{LLAX_LD}	Address Hold after ALE Low, LD/LDD/LDS Instructions	15.0		15.0		ns
4	t_{AVLLC}	Address Valid C to ALE Low	60.0		$0.5t_{CLCL}$-65.0		ns
5	t_{AVRL}	Address Valid to RD Low	200.0		$1.0t_{CLCL}$-50.0		ns
6	t_{AVWL}	Address Valid to WR Low	325.0		$1.5t_{CLCL}$-50.0		ns
7	t_{LLWL}	ALE Low to WR Low	230.0	270.0	$1.0t_{CLCL}$-20.0	$1.0t_{CLCL}$+20.0	ns
8	t_{LLRL}	ALE Low to RD Low	105.0	145.0	$0.5t_{CLCL}$-20.0	$0.5t_{CLCL}$+20.0	ns
9	t_{DVRH}	Data Setup to RD High	95.0		95.0		ns
10	t_{RLDV}	Read Low to Data Valid		170.0		1.0_{CLCL}-80.0	ns
11	t_{RHDX}	Data Hold After RD High	0.0		0.0		ns
12	t_{RLRH}	RD Pulse Width	230.0		$1.0t_{CLCL}$-20.0		ns
13	t_{DVWL}	Data Setup to WR Low	70.0		$0.5t_{CLCL}$-55.0		ns
14	t_{WHDX}	Data Hold After WR High	0.0		0.0		ns
15	t_{DVWH}	Data Valid to WR High	210.0		$1.0t_{CLCL}$-40.0		ns
16	t_{WLWH}	WR Pulse Width	105.0		$0.5t_{CLCL}$-20.0		ns

Table 43. External Data Memory Characteristics, 2.7 - 6.0 Volts, No Wait State

			4 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	8.0	MHz
10	t_{RLDV}	Read Low to Data Valid		420.0		$2.0t_{CLCL}$-80.0	ns
12	t_{RLRH}	RD Pulse Width	480.0		$2.0t_{CLCL}$-20.0		ns
15	t_{DVWH}	Data Valid to WR High	460.0		$2.0t_{CLCL}$-40.0		ns
16	t_{WLWH}	WR Pulse Width	355.0		$1.5t_{CLCL}$-20.0		ns

External Clock Drive Waveforms



External Clock Drive

Symbol	**Parameter**	**VCC = 2.7V to 6.0V**		**VCC = 4.0V to 6.0V**		**Units**
$1/t_{CLCL}$	Oscillator Frequency	0	4	0	6	MHz
t_{CLCL}	Clock Period	250		167		ns
t_{CHCX}	High Time	0		0		ns
t_{CLCX}	Low Time	0		0		ns
t_{CLCH}	Rise Time		1.6		0.5	µs
t_{CHCL}	Fall Time		1.6		0.5	µs

Not in ATmega603.

ATmega603/103 Register Summary

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Page
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	page 14
$3E ($5E)	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	page 14
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	page 14
$3C ($5C)	XDIV	XDIVEN	XDIV6	XDIV5	XDIV4	XDIV3	XDIV2	XDIV1	XDIV0	page 16
$3B ($5B)	RAMPZ	-	-	-	-	-	-	-	RAMPZ0	page 15
$3A ($5A)	EICR	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	page 23
$39 ($59)	EIMSK	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0	page 22
$38 ($58)	EIFR	INTF7	INTF6	INTF5	INTF4	-	-	-	-	page 22
$37 ($57)	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	page 23
$36 ($56)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	page 24
$35 ($55)	MCUCR	SRE	SRW	SE	SM1	SM0	-	-	-	page 15
$34 ($54)	MCUSR	-	-	-	-	-	-	EXTRF	PORF	page 21
$33 ($53)	TCCR0	-	PWM0	COM01	COM00	CTC0	CS02	CS01	CS00	page 28
$32 ($52)	TCNT0	Timer/Counter0 (8 Bit)								page 30
$31 ($51)	OCR0	Timer/Counter0 Output Compare Register								page 30
$30 ($50)	ASSR	-	-	-	-	AS0	TCN0UB	OCR0UB	TCR0UB	page 32
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	-	-	PWM11	PWM10	page 34
$2E ($4E)	TCCR1B	ICNC1	ICES1	-	-	CTC1	CS12	CS11	CS10	page 36
$2D ($4D)	TCNT1H	Timer/Counter1 - Counter Register High Byte								page 36
$2C ($4C)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								page 36
$2B ($4B)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								page 36
$2A ($4A)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								page 36
$29 ($49)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								page 36
$28 ($48)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								page 36
$27 ($47)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								page 37
$26 ($46)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								page 37
$25 ($45)	TCCR2	-	PWM2	COM21	COM20	CTC2	CS22	CS21	CS20	page 28
$24 ($44)	TCNT2	Timer/Counter2 (8 Bit)								page 30
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register								page 30
$21 ($47)	WDTCR	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0	page 39
$1F ($3F)	EEARH	-	-	-	-	EEAR11	EEAR10	EEAR9	EEAR8	page 40
$1E ($3E)	EEARL	EEPROM Address Register L								page 40
$1D ($3D)	EEDR	EEPROM Data Register								page 40
$1C ($3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE	page 40
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	page 56
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	page 56
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	page 56
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	page 58
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	page 58
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	page 58
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	page 63
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	page 64
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	page 64
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	page 64
$0F ($2F)	SPDR	SPI Data Register								page 45
$0E ($2E)	SPSR	SPIF	WCOL	-	-	-	-	-	-	page 45
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	page 44
$0C ($2C)	UDR	UART I/O Data Register								page 48
$0B ($2B)	USR	RXC	TXC	UDRE	FE	OR	-	-	-	page 48
$0A ($2A)	UCR	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8	page 49
$09 ($29)	UBRR	UART Baud Rate Register								page 50
$08 ($28)	ACSR	ACD	-	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	page 51
$07 ($27)	ADMUX	-	-	-	-	-	MUX2	MUX1	MUX0	page 53
$06 ($26)	ADCSR	ADES	ABSY	ADRF	ADIF	ADIE	ADPS2	ADPS1	ADPS0	page 53
$05 ($25)	ADCH	-	-	-	-	-	-	ADC9	ADC8	page 54
$04 ($24)	ADCL	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	page 54
$03 ($23)	PORTE	PORTE7	PORTE6	PORTE5	PORTE4	PORTE3	PORTE2	PORTE1	PORTE0	page 67
$02 ($22)	DDRE	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0	page 67
$01 ($21)	PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0	page 67
$00 ($20)	PINF	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0	page 71

ATmega603/103 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 ← $FF - Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← $00 - Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • ($FF - 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 ← $FF	None	1
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
JMP	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3
ICALL		Indirect Call to (Z)	PC ← Z	None	3
CALL	k	Direct Subroutine Call	PC ← k	None	4
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
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
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1 / 2
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1 / 2
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1 / 2
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

ATmega603/103 Instruction Set Summary (Continued)

DATA TRANSFER INSTRUCTIONS					
ELPM(0)		Extended Load Program Memory	R0 ← (Z+RAMPZ)	None	3
MOV	Rd, Rr	Move Between Registers	Rd ← 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
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
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
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
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	3
WDR		Watchdog Reset	(see specific descr. for WD timer)	None	1

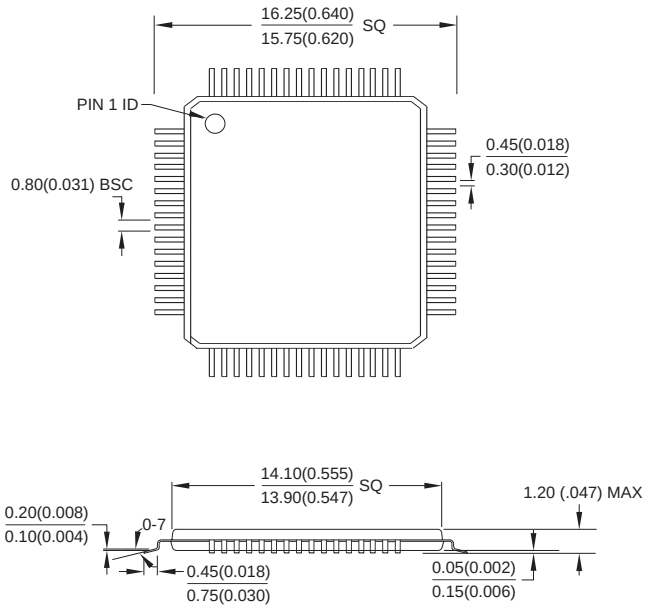
Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
4	2.7 - 6.0V	ATmega603L-4AC	64A	Commercial (0°C to 70°C)
		ATmega603L-4AI	64A	Industrial (-40°C to 85°C)
6	4.0 - 6.0V	ATmega603-6AC	64A	Commercial (0°C to 70°C)
		ATmega603-6AI	64A	Industrial (-40°C to 85°C)
4	2.7 - 6.0V	ATmega103L-4AC	64A	Commercial (0°C to 70°C)
		ATmega103L-4AI	64A	Industrial (-40°C to 85°C)
6	4.0 - 6.0V	ATmega103-6AC	64A	Commercial (0°C to 70°C)
		ATmega103-6AI	64A	Industrial (-40°C to 85°C)

Package Type	
64A	64-Lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)

Packaging Information

64A, 64-Lead, Very Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (VQFP)
Dimensions in Millimeters and (Inches)*



*Controlling dimension: millimeters