Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 130 Powerful Instructions - Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 8 MIPS Throughput at 8 MHz
  - On-chip 2-cycle Multiplier
- Program and Data Memories
  - 16K Bytes of Nonvolatile In-System Programmable Flash
    Endurance: 1,000 Write/Erase Cycles
  - Optional Boot Code Memory with Independent Lock Bits
    Self-programming of Program and Data Memories
  - 512 Bytes Nonvolatile In-System Programmable EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 1K Bytes Internal SRAM
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and PWM
  - Expanded 16-bit Timer/Counter System with Separate Prescaler, Compare, Capture Modes and Dual 8-, 9- or 10-bit PWM
  - Dual Programmable Serial UARTs
  - Master/Slave SPI Serial Interface
  - Real Time Counter with Separate Oscillator
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - External and Internal Interrupt Sources
  - Three Sleep Modes: Idle, Power Save and Power-down
- I/O and Packages
  - 35 Programmable I/O Lines
  - 40-pin PDIP, 44-pin PLCC and TQFP
- Operating Voltages
  - 2.7V - 5.5V (ATmega161L), 4.0V - 5.5V (ATmega161)
- Speed Grades
  - 0 - 4 MHz (ATmega161L), 0 - 8 MHz (ATmega161)
- Commercial and Industrial Temperature Ranges

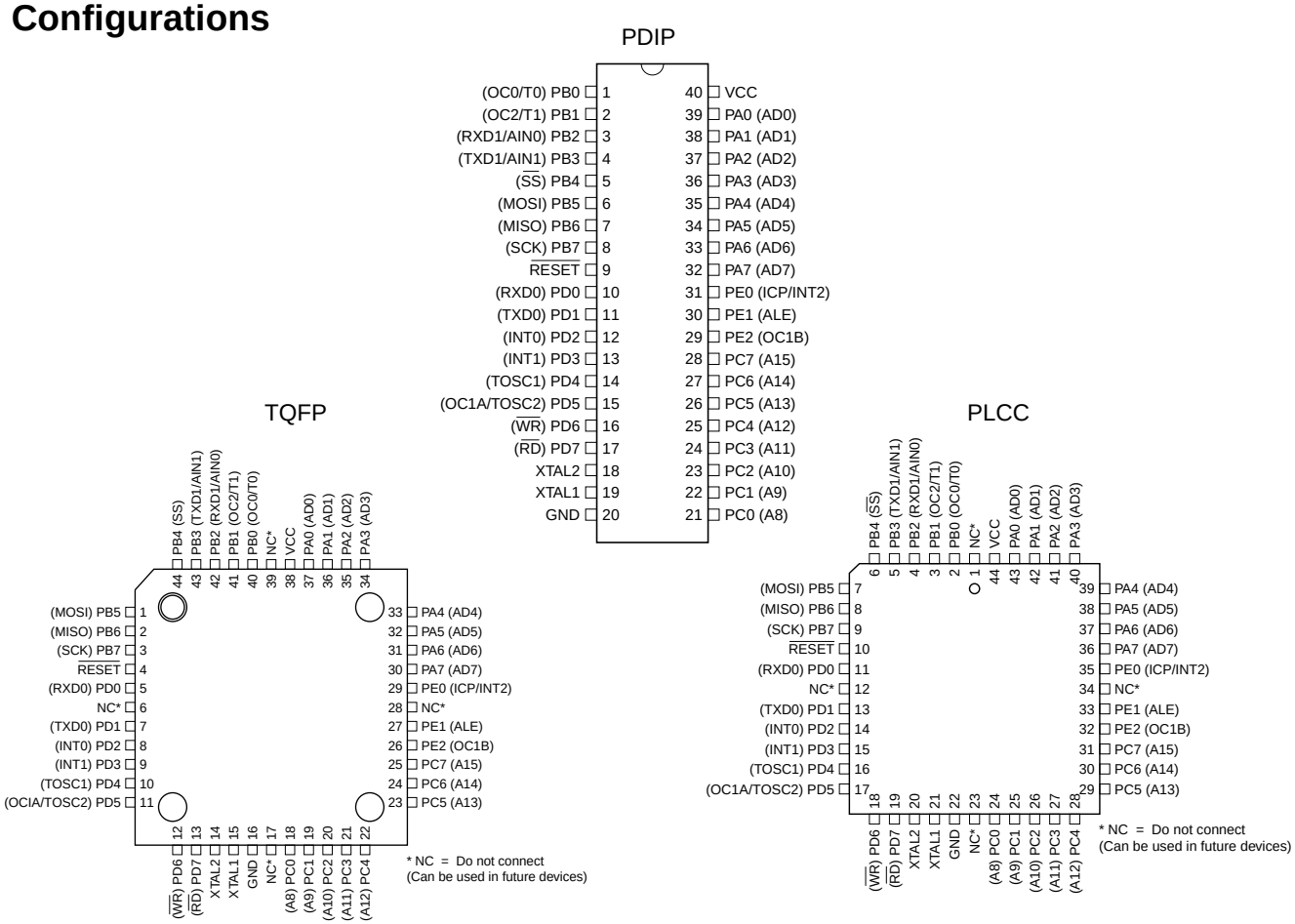
8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

ATmega161
ATmega161L

Advance Information

Rev. 1228A–08/99

Pin Configurations



Description

The ATmega161 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega161 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.The AVR core 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.

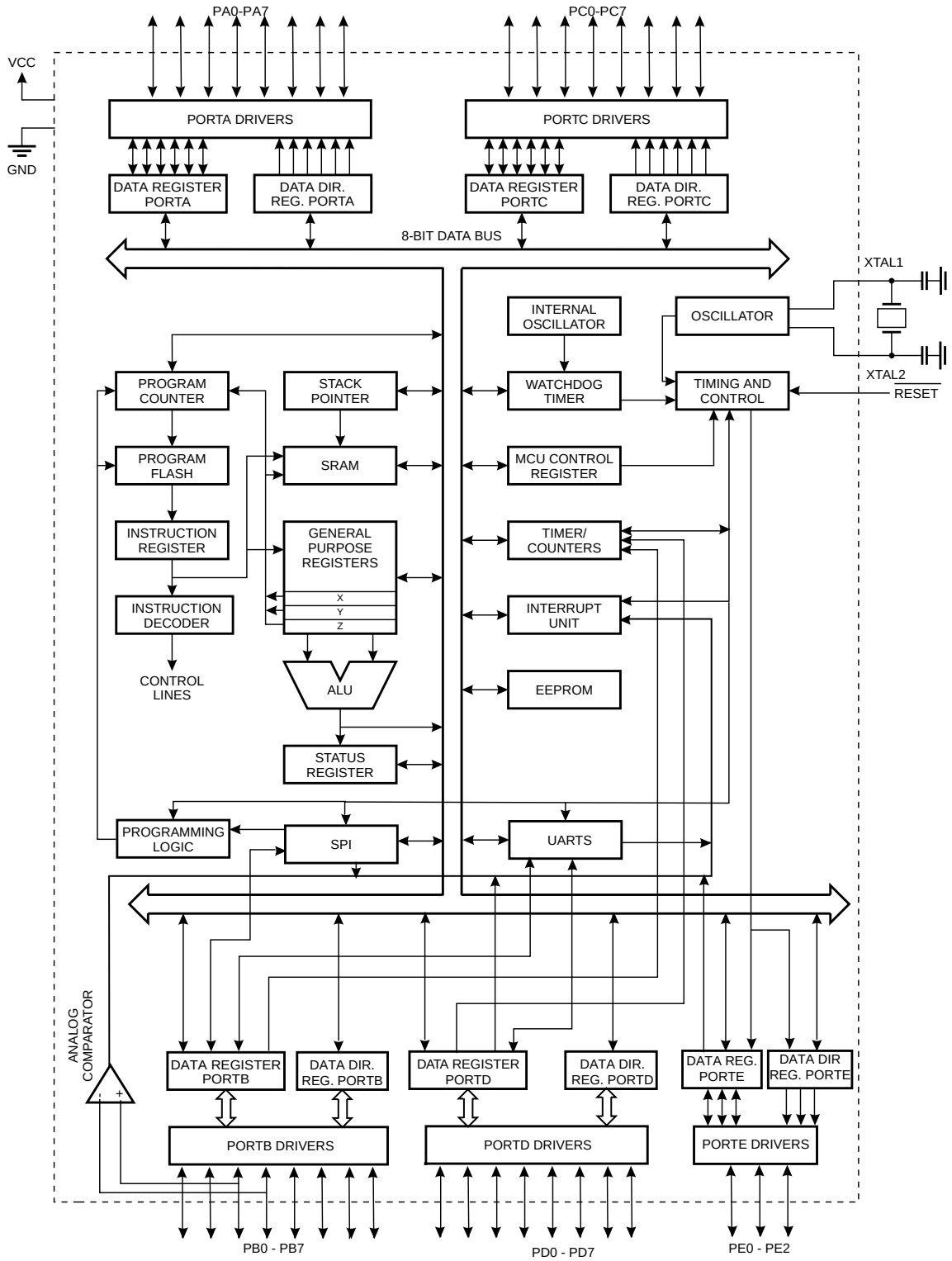
The ATmega161 provides the following features: 16K bytes of In-System- or Self-programmable Flash, 512 bytes EEPROM, 1K bytes SRAM, 35 general purpose I/O lines, 32 general purpose working registers, Real Time Counter, three flexible timer/counters with compare modes, internal and external interrupts, two programmable serial UARTs, 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 and SRAM contents but freezes the oscillator, disabling all other chip functions until the next external 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 nonvolatile memory technology. The on-chip Flash program memory can be reprogrammed using the self-programming capability through the bootblock, using an ISP through the SPI-port, or by using a conventional nonvolatile memory programmer. By combining an enhanced RISC 8-bit CPU with In-System Programmable Flash on a monolithic chip, the Atmel ATmega161 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega161 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.

Block Diagram

Figure 1. The ATmega161 Block Diagram



Pin Descriptions

VCC

Supply voltage

GND

Ground

Port A (PA7..PA0)

Port A is an 8-bit bidirectional 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. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A serves as Multiplexed Address/Data port when using external memory interface.

Port B (PB7..PB0)

Port B is an 8-bit bidirectional I/O port 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. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega161 as listed on page 80.

Port C (PC7..PC0)

Port C is an 8-bit bidirectional I/O port with internal pull-up resistors. The Port C output buffers can sink 20 mA. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves as Address high output when using external memory interface.

Port D (PD7..PD0)

Port D is an 8-bit bidirectional 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. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega161 as listed on page 87.

Port E (PE2..PE0)

Port E is a 3-bit bidirectional 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. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E also serves the functions of various special features of the ATmega161 as listed on page 93.

$\overline{\text{RESET}}$

Reset input. A low level on this pin for more than 500 ns will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

XTAL1

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

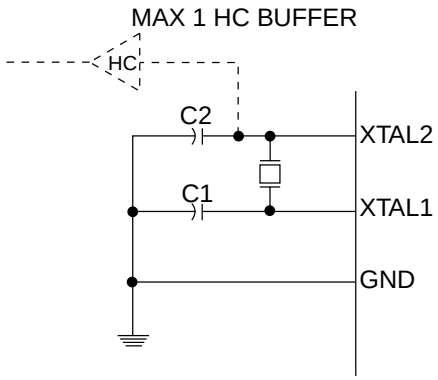
XTAL2

Output from the inverting oscillator amplifier

Crystal Oscillator

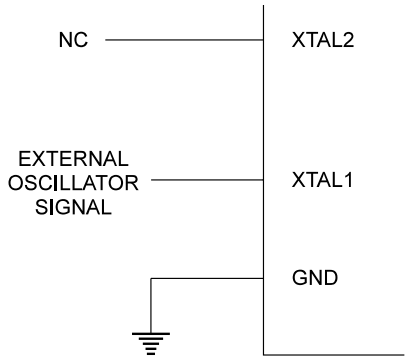
XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or a ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 3.

Figure 2. Oscillator Connections



Note: When using the MCU Oscillator as a clock for an external device, an HC buffer should be connected as indicated in the figure.

Figure 3. External Clock Drive Configuration

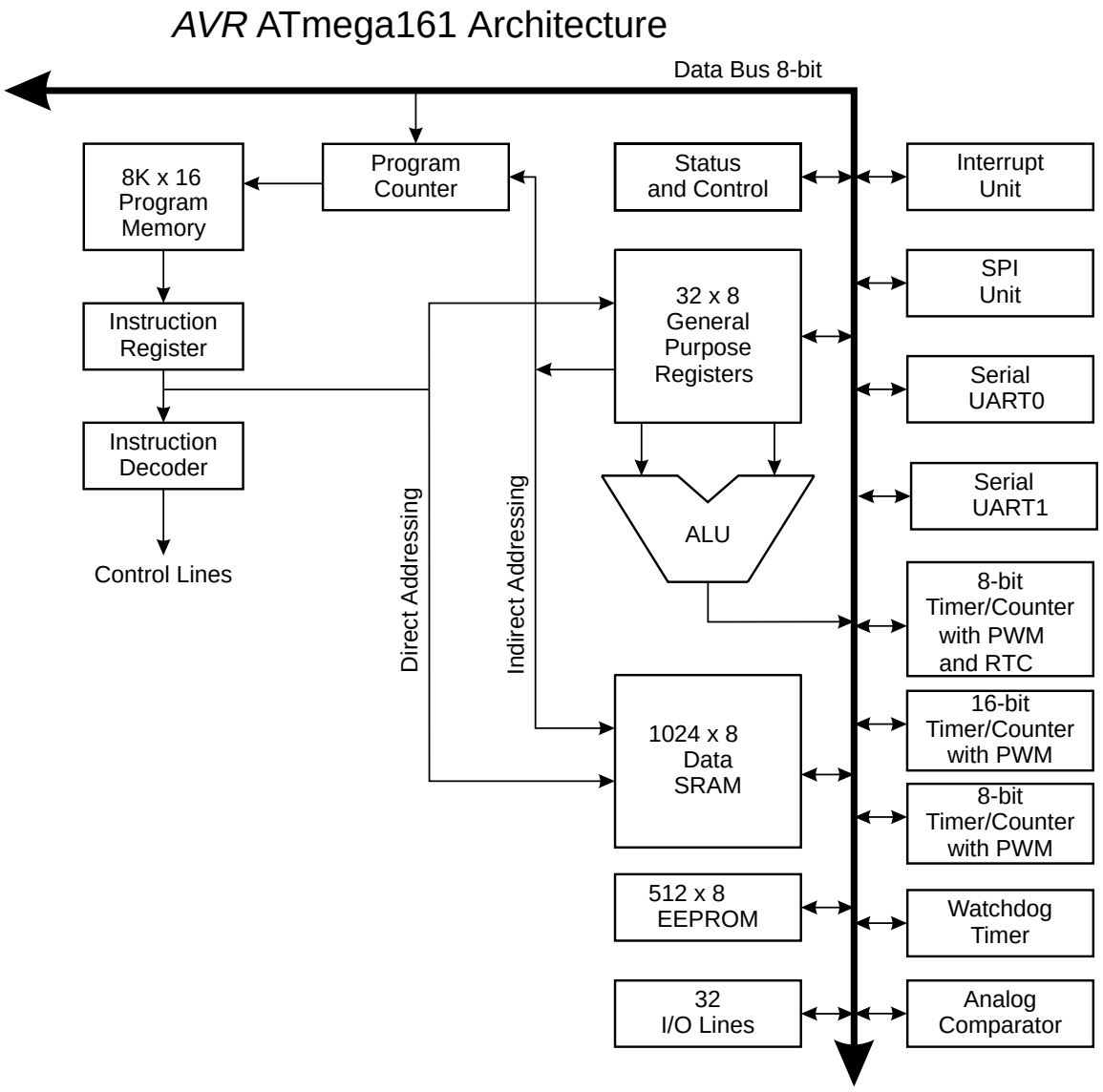


Architectural Overview

The fast-access register file concept 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 Arithmetic Logic Unit (ALU) 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-bits 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-bits X-register, Y-register and Z-register.

Figure 4. The ATmega161 AVR RISC Architecture



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 ATmega161 AVR 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 ($00 - $1F), 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, 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.

The AVR uses a Harvard architecture concept – with separate memories and buses for program and data. The program memory is executed with a two stage pipeline. 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 Self-programmable Flash memory.

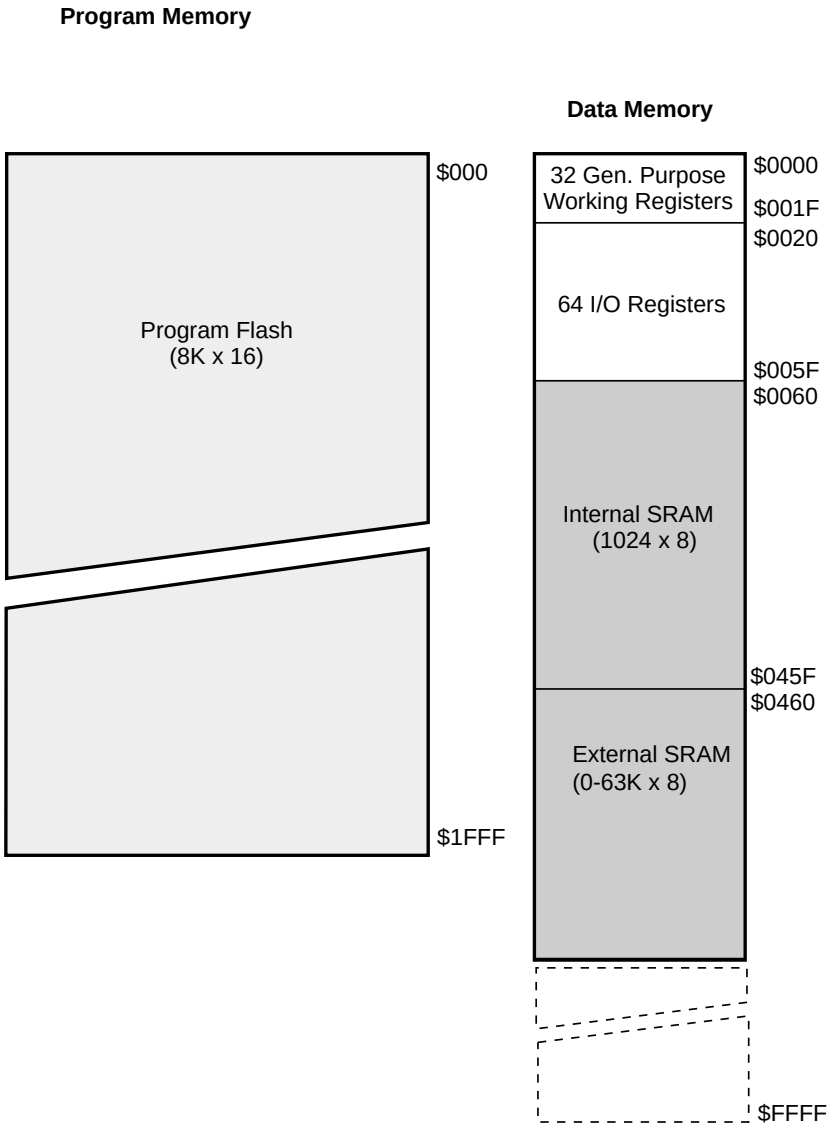
With the jump and call instructions, the whole 8K word address space is directly accessed. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

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 (Stack Pointer) in the reset routine (before subroutines or interrupts are executed). The 16-bit stack pointer is read/write accessible in the I/O space.

The 1K bytes data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.

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

Figure 5. Memory Maps

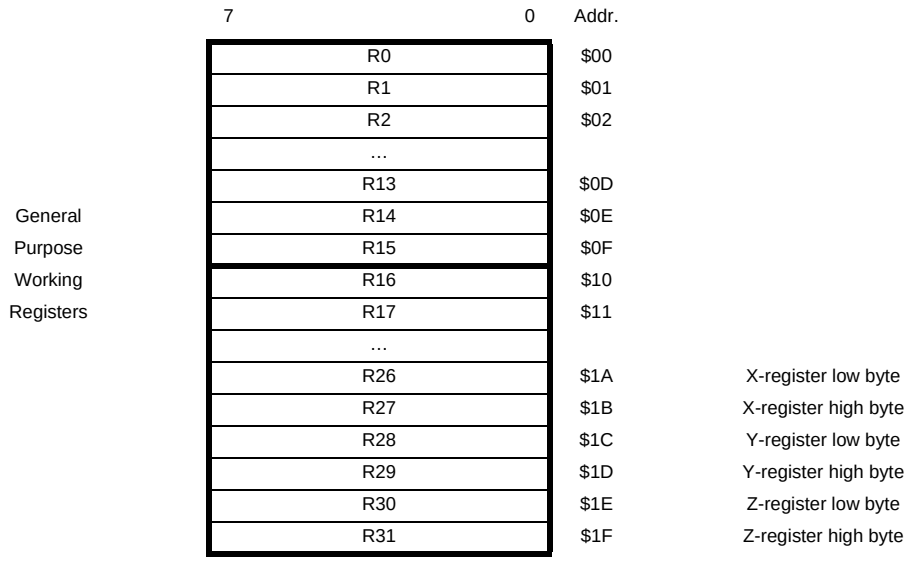


A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All the different interrupts have a separate interrupt vector in the interrupt vector table at the beginning of the program memory. The different interrupts have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

General Purpose Register File

Figure 6 shows the structure of the 32 general purpose working registers in the CPU.

Figure 6. AVR CPU General Purpose Working Registers



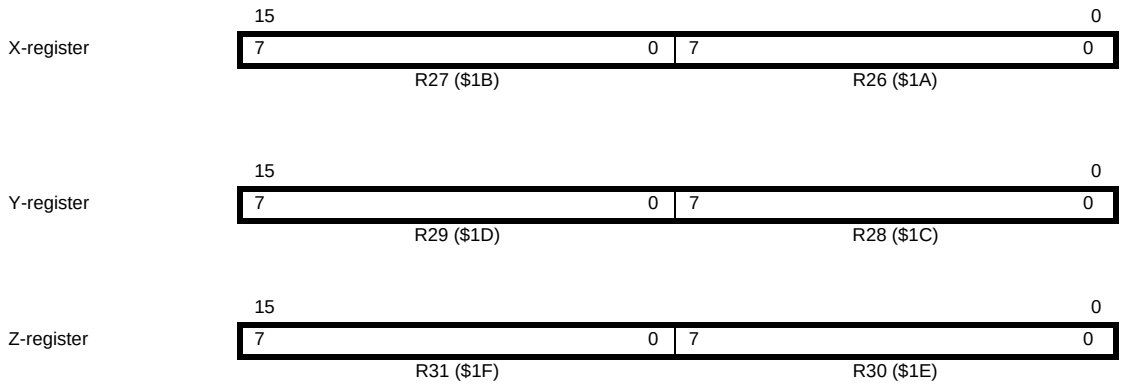
All the register operating instructions in the instruction set have direct and single cycle access to all registers. The only exceptions are the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI, and ORI between a constant and a register, and the LDI instruction for load immediate 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 6, 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.

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 Data Space. The three indirect address registers X, Y and Z are defined as:

Figure 7. 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).

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. ATmega161 does also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See Instruction Set section for a detailed description.

Self-programmable Flash Program Memory

The ATmega161 contains 16K bytes on-chip Self- and In-System programmable Flash memory for program storage. Since all instructions are 16- or 32-bit words, the Flash is organized as 8K x 16. The Flash memory has an endurance of at least 1000 write/erase cycles. The ATmega161 Program Counter (PC) is 13 bits wide, thus addressing the 8192 program memory locations.

See page 95 for a detailed description on Flash data downloading.

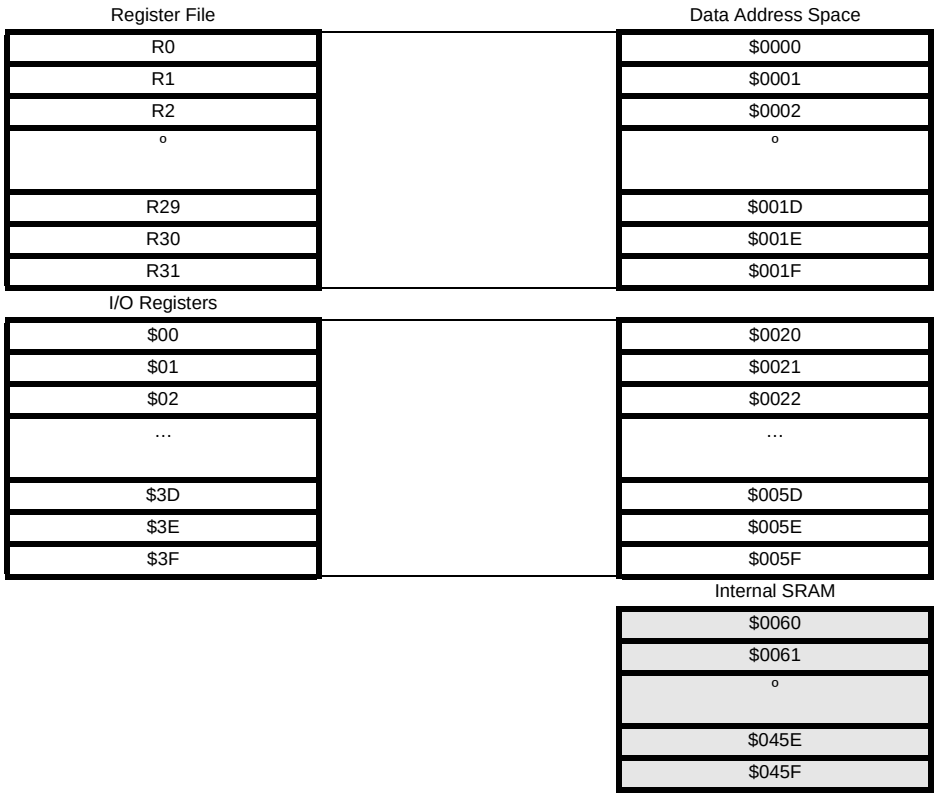
See page 11 for the different program memory addressing modes.

EEPROM Data Memory

The ATmega161 contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles per location. The interface between the EEPROM and the CPU is described on page 54 specifying the EEPROM address registers, the EEPROM data register, and the EEPROM control register.

For the SPI data downloading, see page 109 for a detailed description.

Figure 8. SRAM Organization



SRAM Data Memory

Figure 8 shows how the ATmega161 SRAM Memory is organized.

The lower 1120 Data Memory locations address 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 1K locations address the internal data SRAM. An optional external data memory device can be placed in the same SRAM memory space. This memory device will occupy the locations following the internal SRAM and up to as much as 64K - 1, depending on external memory size.

When the addresses accessing the data memory space exceeds the internal data SRAM locations, the memory device is accessed using the same instructions as for the internal data SRAM access. When the internal data space is accessed, the read and write strobe pins ($\overline{RD}$ and $\overline{WR}$) are inactive during the whole access cycle. External memory operation is enabled by setting the SRE bit in the MCUCR register. See “Interface to external memory” on page 72 for details.

Accessing external memory 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 memory, interrupts, subroutine calls and returns take two clock cycles extra because the two-byte program counter is pushed and popped. When external memory 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 direct addressing reaches the entire data space.

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 32 general purpose working registers, 64 I/O registers and the 1K bytes of internal data SRAM in the ATmega161 are all accessible through all these addressing modes.

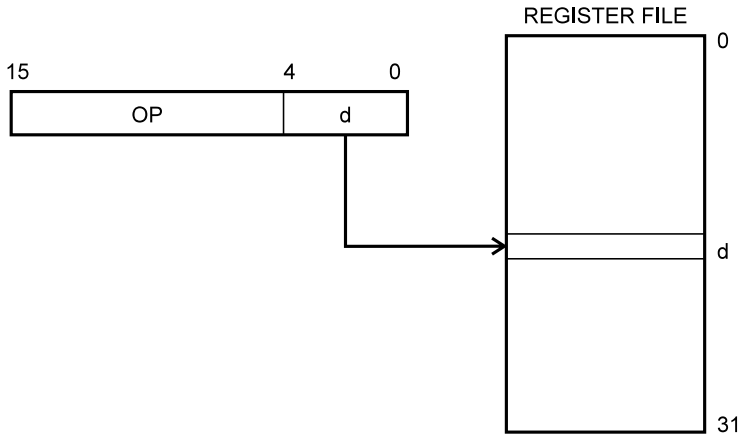
See the next section for a detailed description of the different addressing modes.

Program and Data Addressing Modes

The ATmega161 AVR 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

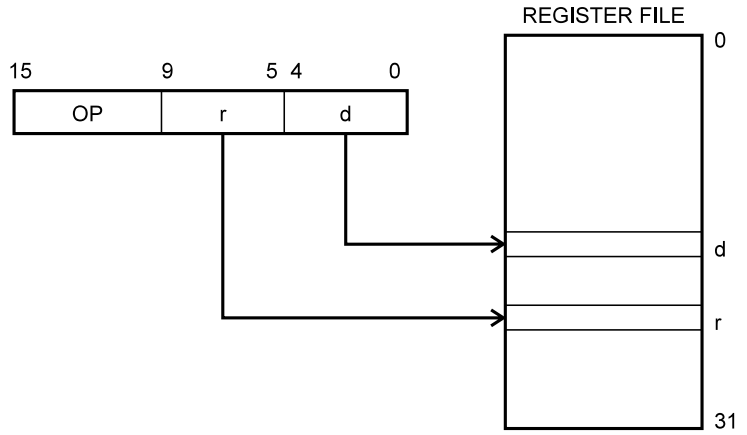
Figure 9. Direct Single Register Addressing



The operand is contained in register d (Rd).

Register Direct, Two Registers Rd and Rr

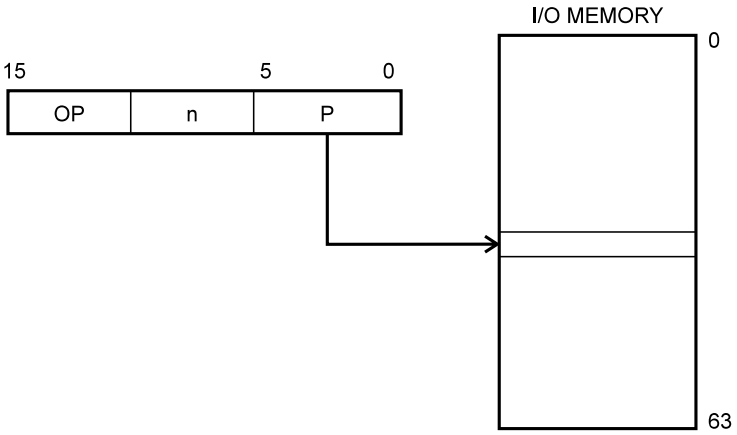
Figure 10. Direct Register Addressing, Two Registers



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

I/O Direct

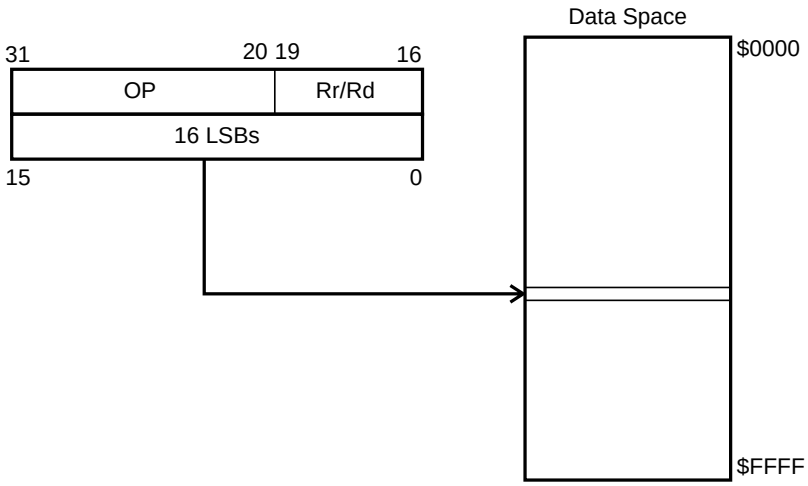
Figure 11. I/O Direct Addressing



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

Data Direct

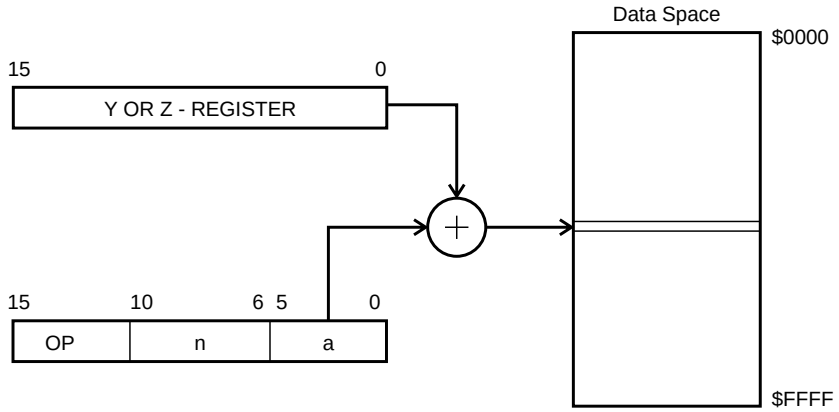
Figure 12. Direct Data Addressing



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

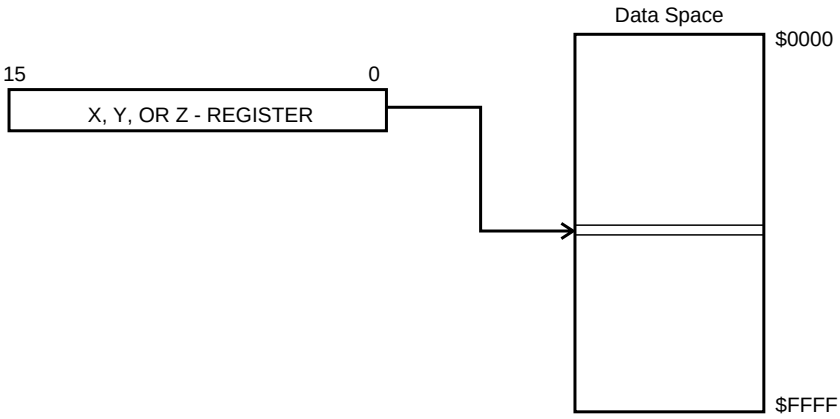
Figure 13. Data Indirect with Displacement



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

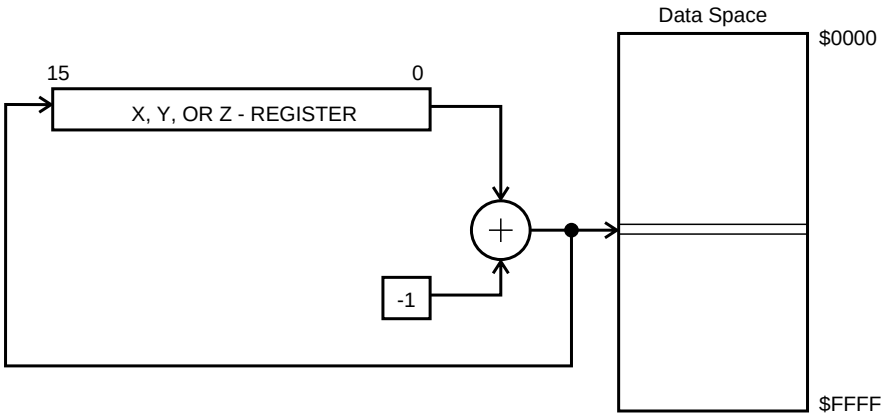
Figure 14. Data Indirect Addressing



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

Data Indirect with Pre-Decrement

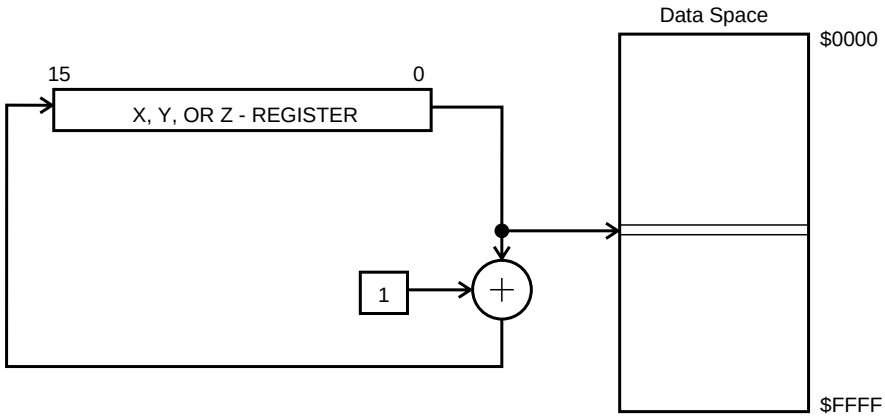
Figure 15. Data Indirect Addressing with Pre-Decrement



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

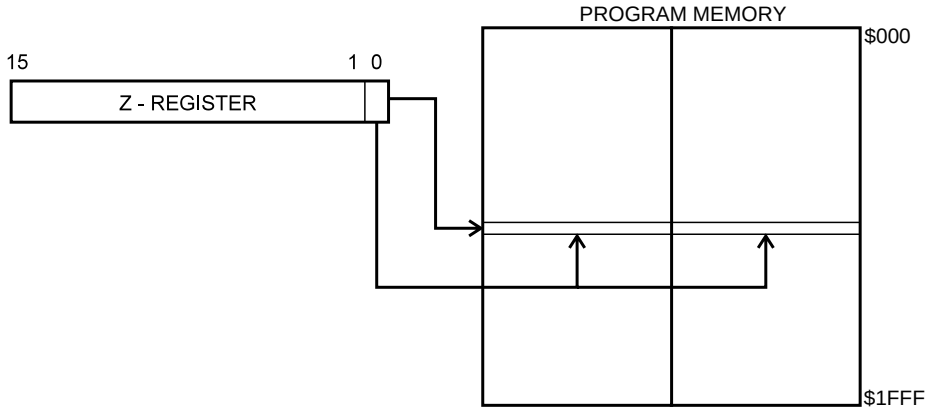
Figure 16. Data Indirect Addressing with Post-Increment



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 Instruction

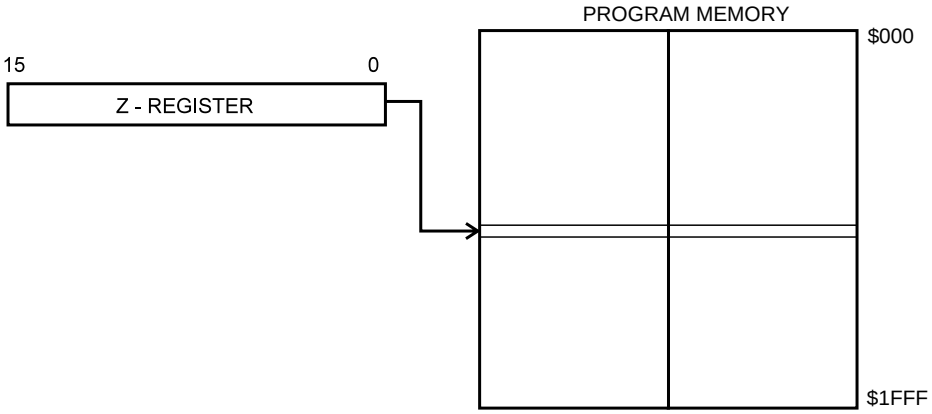
Figure 17. Code Memory Constant Addressing



Constant byte address is specified by the Z-register contents. The 15 MSBs select word address (0 - 8K), the LSB selects low byte if cleared (LSB = 0) or high byte if set (LSB = 1).

Indirect Program Addressing, IJMP and ICALL

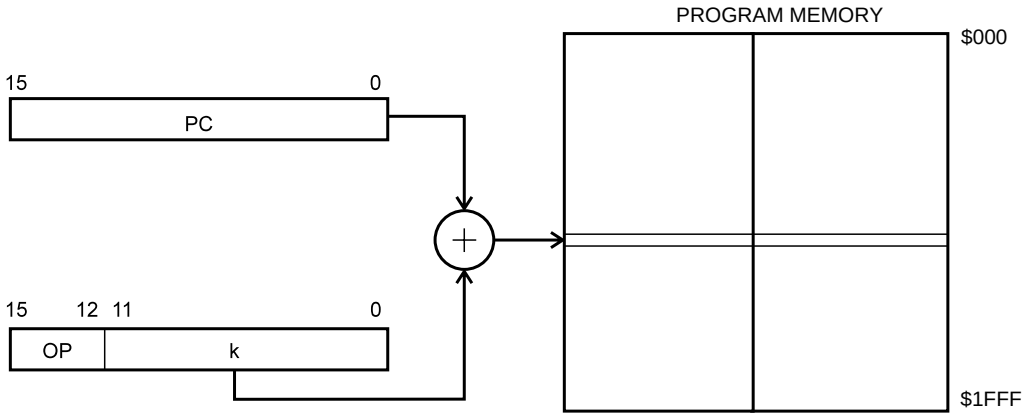
Figure 18. Indirect Program Memory Addressing



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

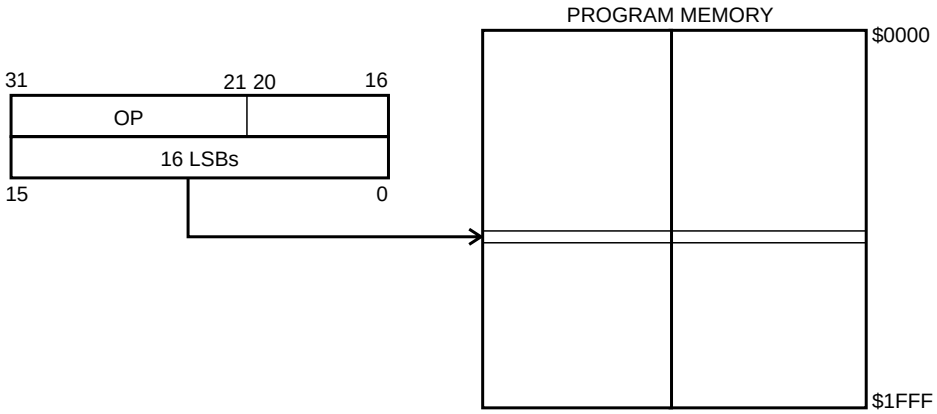
Figure 19. Relative Program Memory Addressing



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

Direct Program Addressing, JMP and CALL

Figure 20. Direct Program Addressing



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

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 21 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 21. The Parallel Instruction Fetches and Instruction Executions

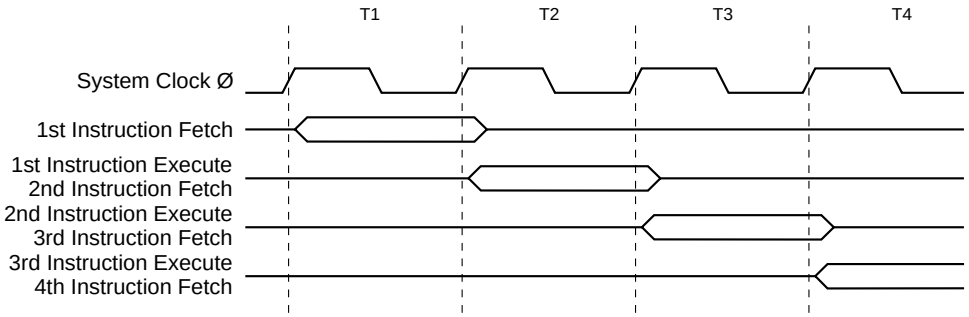
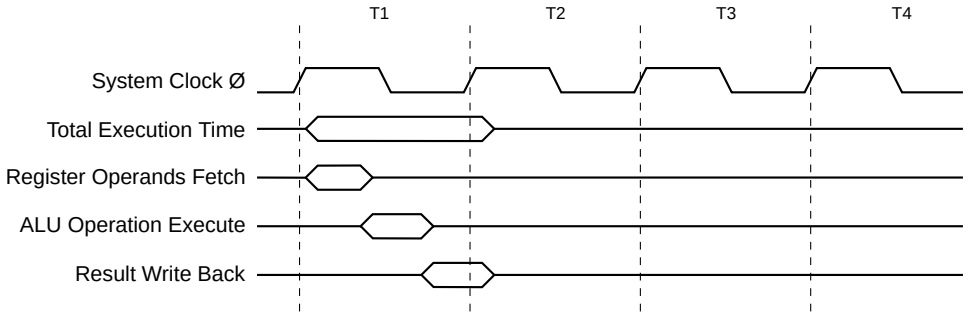


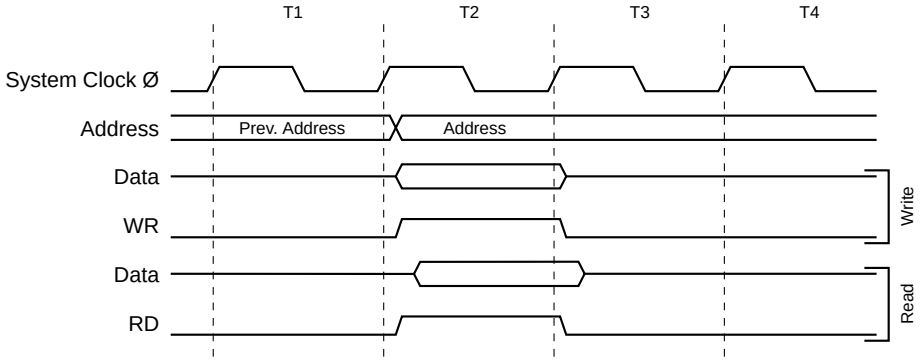
Figure 22 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 22. Single Cycle ALU Operation



The internal data SRAM access is performed in two System Clock cycles as described in Figure 23.

Figure 23. On-chip Data SRAM Access Cycles



I/O Memory

The I/O space definition of the ATmega161 is shown in the following table:

Table 1. ATmega161 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
$3B ($5B)	GIMSK	General Interrupt MaSK register
$3A ($5A)	GIFR	General Interrupt Flag Register
$39 ($59)	TIMSK	Timer/Counter Interrupt MaSK Register
$38 ($58)	TIFR	Timer/Counter Interrupt Flag Register
$37 ($57)	SPMCR	Store Program Memory Control Register
$36 ($56)	EMCUCR	Extended MCU general Control Register
$35 ($55)	MCUCR	MCU general Control Register
$34 ($54)	MCUSR	MCU general 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)	SFIOR	Special Function IO 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 RegisterA High Byte
$2A ($4A)	OCR1AL	Timer/Counter1 Output Compare RegisterA Low Byte
$29 ($49)	OCR1BH	Timer/Counter1 Output Compare RegisterB High Byte
$28 ($48)	OCR1BL	Timer/Counter1 Output Compare RegisterB Low Byte
$27 ($47)	TCCR2	Timer/Counter2 Control Register
$26 ($46)	ASSR	Asynchronous mode StatuS Register
$25 ($45)	ICR1H	Timer/Counter1 Input Capture Register High Byte
$24 ($44)	ICR1L	Timer/Counter1 Input Capture Register Low Byte
$23 ($43)	TCNT2	Timer/Counter2 (8-bit)
$22 ($42)	OCR2	Timer/Counter2 Output Compare Register
$21 ($41)	WDTCR	Watchdog Timer Control Register
$20 ($40)	UBRRHI	UART Baud Register HIgh
$1F ($3F)	EEARH	EEPROM Address Register High
$1E ($3E)	EEARL	EEPROM Address Register Low
$1D ($3D)	EEDR	EEPROM Data Register

Table 1. ATmega161 I/O Space (Continued)

I/O Address (SRAM Address)	Name	Function
$1C ($3C)	EECR	EEPROM Control Register
$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
$14 ($34)	DDRC	Data Direction Register, Port C
$13 ($33)	PINC	Input Pins, 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)	UDR0	UART0 I/O Data Register
$0B ($2B)	UCSR0A	UART0 Control and Status Register
$0A ($2A)	UCSR0B	UART0 Control and Status Register
$09 ($29)	UBRR0	UART0 Baud Rate Register
$08 ($28)	ACSR	Analog Comparator Control and Status Register
$07 ($27)	PORTE	Data Register, Port E
$06 ($26)	DDRE	Data Direction Register, Port E
$05 ($25)	PINE	Input Pins, Port E
$03 ($23)	UDR1	UART1 I/O Data Register
$02 ($22)	UCSR1A	UART1 Control and Status Register
$01 ($21)	UCSR1B	UART1 Control and Status Register
$00 ($20)	UBRR1	UART1 Baud Rate Register

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

All ATmega161 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range $00 - $1F 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. When using the I/O specific commands IN, OUT the I/O addresses $00 - $3F must be used. When addressing I/O registers as SRAM, $20 must be added to this address. All I/O register addresses throughout this document are shown with the SRAM address in parentheses.

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

Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

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

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 separate control registers. If the global interrupt enable bit is cleared (zero), none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts.

- **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 $\oplus$ 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.

Note that the status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt routine. This must be handled by software.

Stack Pointer – SP

The ATmega161 Stack Pointer is implemented as two 8-bit registers in the I/O space locations $3E ($5E) and $3D ($5D). As the ATmega161 supports up to 64KB 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 must be set to point above $60. 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 an address is pushed onto the Stack with subroutine calls and interrupts. 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 an address is popped from the Stack with return from subroutine RET or return from interrupt RETI.

Reset and Interrupt Handling

The ATmega161 provides 20 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 2. 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 2. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	$000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	INT2	External Interrupt Request 2
5	$008	TIMER2 COMP	Timer/Counter2 Compare Match
6	$00a	TIMER2 OVF	Timer/Counter2 Overflow
7	$00c	TIMER1 CAPT	Timer/Counter1 Capture Event
8	$00e	TIMER1 COMPA	Timer/Counter1 Compare Match A
9	$010	TIMER1 COMPB	Timer/Counter1 Compare Match B
10	$012	TIMER1 OVF	Timer/Counter1 Overflow
11	$014	TIMER0 COMP	Timer/Counter0 Compare Match
12	$016	TIMER0 OVF	Timer/Counter0 Overflow
13	$018	SPI, STC	Serial Transfer Complete
14	$01a	UART0, RX	UART0, Rx Complete
15	$01c	UART1, RX	UART1, Rx Complete

Table 2. Reset and Interrupt Vectors (Continued)

Vector No.	Program Address	Source	Interrupt Definition
16	$01e	UART0, UDRE	UART0 Data Register Empty
17	$020	UART1, UDRE	UART1 Data Register Empty
18	$022	UART0, TX	UART0, Tx Complete
19	$024	UART1, TX	UART1, Tx Complete
20	$026	EE_RDY	EEPROM Ready
21	$028	ANA_COMP	Analog Comparator

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

```
Address              Labels     Code              Comments
$000                            jmp    RESET      ; Reset Handler
$002                            jmp    EXT_INT0   ; IRQ0 Handler
$004                            jmp    EXT_INT1   ; IRQ1 Handler
$006                            jmp    EXT_INT2   ; IRQ2 Handler
$008                            jmp    TIM2_COMP  ; Timer2 Compare Handler
$00a                            jmp    TIM2_OVF   ; Timer2 Overflow Handler
$00c                            jmp    TIM1_CAPT  ; Timer1 Capture Handler
$00e                            jmp    TIM1_COMPA ; Timer1 CompareA Handler
$010                            jmp    TIM1_COMPB ; Timer1 CompareB Handler
$012                            jmp    TIM1_OVF   ; Timer1 Overflow Handler
$014                            jmp    TIM0_COMP  ; Timer0 Compare Handler
$016                            jmp    TIM0_OVF   ; Timer0 Overflow Handler
$018                            jmp    SPI_STC;   ; SPI Transfer Complete Handler
$01a                            jmp    UART_RXC0  ; UART0 RX Complete Handler
$01c                            jmp    UART_RXC1  ; UART1 RX Complete Handler
$01e                            jmp    UART_DRE0  ; UDR0 Empty Handler
$020                            jmp    UART_DRE1  ; UDR1 Empty Handler
$022                            jmp    UART_TXC0  ; UART0 TX Complete Handler
$024                            jmp    UART_TXC1  ; UART1 TX Complete Handler
$026                            jmp    EE_RDY     ; EEPROM Ready Handler
$028                            jmp    ANA_COMP   ; Analog Comparator Handler
;
$02a                MAIN:       ldi r16,high(RAMEND); Main program start
$02b                            out SPH,r16
$02c                            ldi r16,low(RAMEND)
$02d                            out SPL,r16
$02e                            <instr>  xxx
…                     …           …      …
```

Reset Sources

The ATmega161 has four sources of reset:

- Power-on Reset. The MCU is reset when the supply voltage is below the power-on reset threshold (V_{POT}).
- External Reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for more than 500 ns.
- Watchdog Reset. The MCU is reset when the Watchdog timer period expires and the Watchdog is enabled.
- Brown-out Reset. The MCU is reset when the supply voltage V_{CC} falls below a certain voltage.

During reset, all I/O registers are then set to their initial values, and the program starts execution from address $000. The instruction placed in address $000 must be an JMP – relative jump – instruction to the reset handling routine. If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. The circuit diagram in Figure 24 shows the reset logic. Table 3 and Table 4 defines the timing and electrical parameters of the reset circuitry

Figure 24. Reset Logic

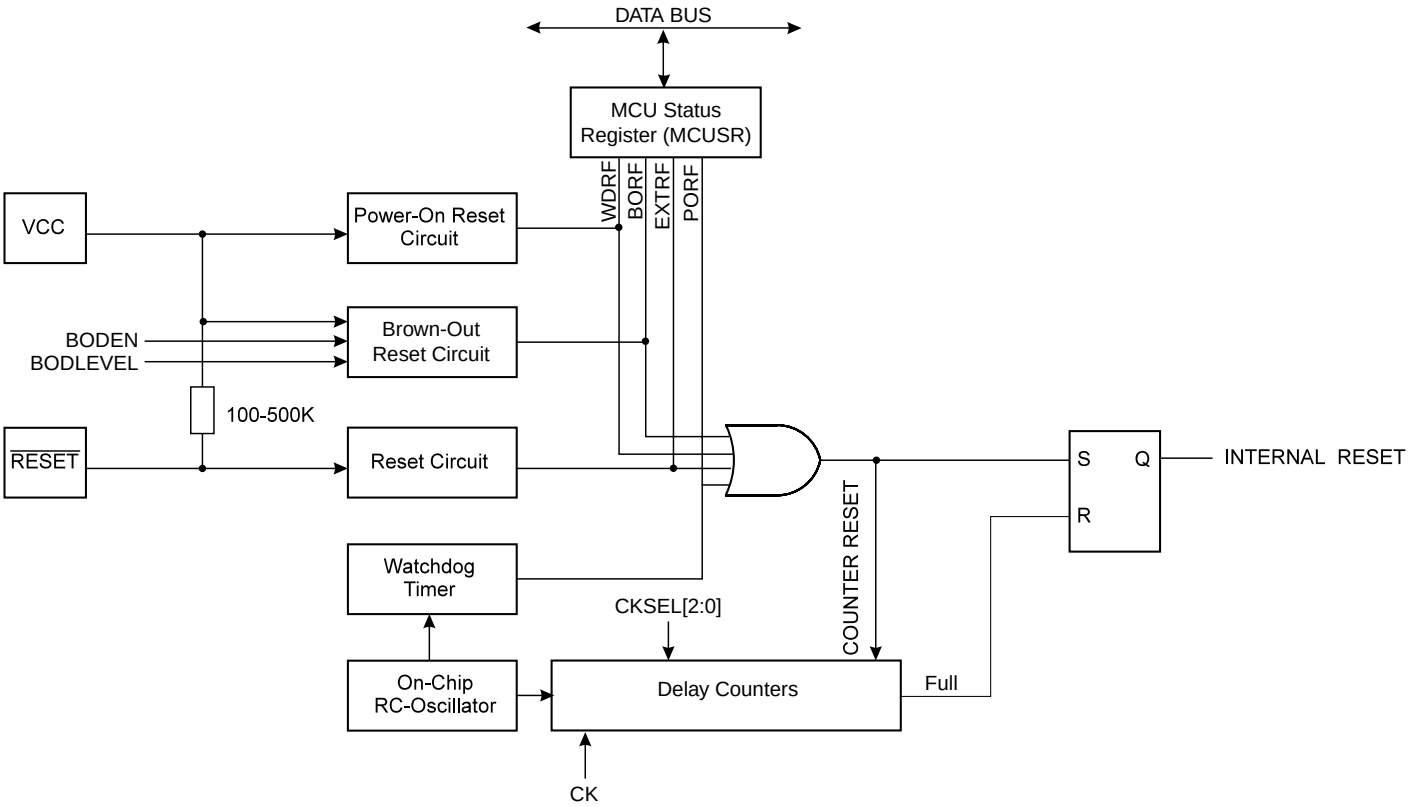


Table 3. Reset Characteristics (V_{CC} = 5.0V)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{POT}	Power-on Reset Threshold Voltage (rising)	BOD disabled	1.0	1.4	1.8	V
		BOD enabled	1.7	2.2	2.7	V
	Power-on Reset Threshold Voltage (falling)	BOD disabled	0.4	0.6	0.8	V
		BOD enabled	1.7	2.2	2.7	V
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		-	-	0.85 V_{CC}	V
V_{BOT}	Brown-out Reset Threshold Voltage	(BODLEVEL = 1)	2.6	2.7	2.8	V
		(BODLEVEL = 0)	3.8	4.0	4.2	

Note: The Power-on Reset will not work unless the supply voltage has been below V_{POT} (falling).

Table 4. Reset Delay Selections

CKSEL [2:0]	Start-up Time, V_{CC} = 2.7V, BODLEVEL Unprogrammed	Start-up Time, V_{CC} = 4.0V, BODLEVEL Programmed	Recommended Usage(1)
000	4.2 ms + 6 CK	5.8 ms + 6 CK	External Clock, fast rising power
001	30 µs + 6 CK	10 µs + 6 CK	External Clock, BOD enabled(2)(3)
010	67 ms + 16K CK	92 ms + 16K CK	Crystal Oscillator, slowly rising power
011	4.2 ms + 16K CK	5.8 ms + 16K CK	Crystal Oscillator, fast rising power
100	30 µs + 16K CK	10 µs + 16K CK	Crystal Oscillator, BOD enabled(2)(3)
101	67 ms + 1K CK	92 ms + 1K CK	Ceramic Resonator/External clock, Slowly rising power
110	4.2 ms + 1K CK	5.8 ms + 1K CK	Ceramic Resonator, fast rising power
111	30 µs + 1K CK	10 µs + 1K CK	Ceramic Resonator, BOD enabled(2)(3)

Notes: 1. The CKSEL fuses control only the start-up time. The oscillator is the same for all selections. On power-up, the real-time part of the start-up time is increased with typ. 0.6ms.
2. Or external power-on reset.
3. When BOD is enabled, there will be a real-time part = 50 µs (typ.)

Table 4 shows the start-up times from reset. From sleep, only the clock counting part of the start-up time is used. The watchdog oscillator is used for timing the real-time part of the start-up time. The number WDT oscillator cycles used for each time-out is shown in Table 5.

Table 5. Number of Watchdog Oscillator Cycles

BODLEVEL	Time-out	Number of cycles
Unprogrammed	4.2 ms (at V_{cc}=2.7V)	1K
Unprogrammed	67 ms (at V_{cc}=2.7V)	16K
Programmed	5.8 ms (at V_{cc}=4.0V)	4K
Programmed	92 ms (at V_{cc}=4.0V)	64K

Note: The bod-level fuse can be used to select start-up times even if the Brown-out detection is disabled (by leaving the BODEN fuse unprogrammed).

The frequency of the watchdog oscillator is voltage dependent as shown in the Electrical Characteristics section. The device is shipped with CKSEL = 010.

Power-on Reset

A Power-on Reset (POR) pulse is generated by an on-chip detection circuit. The detection level is nominally 1.4V (rising VCC). The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the start-up reset, as well as detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is reset from power-on. Reaching the power-on reset threshold voltage invokes a delay counter, which determines the delay, for which the device is kept in RESET after V_{CC} rise. The time-out period of the delay counter can be defined by the user through the CKSEL fuses. The eight different selections for the delay period are presented in Table 4. The RESET signal is activated again, without any delay, when the V_{CC} decreases below detection level.

Figure 25. MCU Start-up, $\overline{\text{RESET}}$ Tied to VCC.

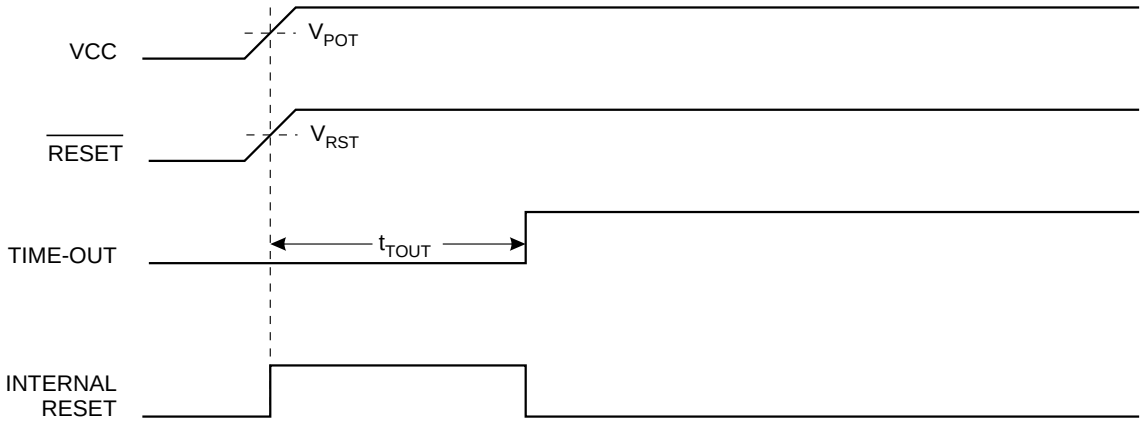
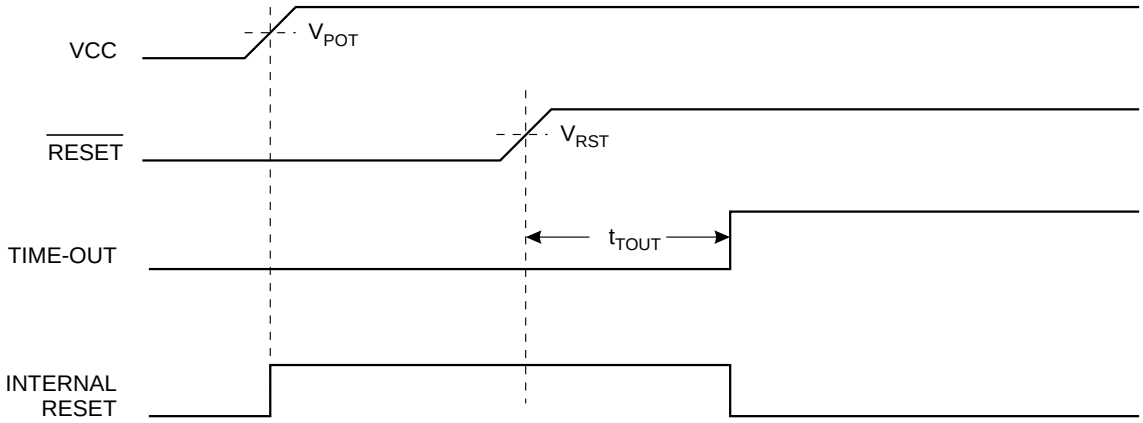


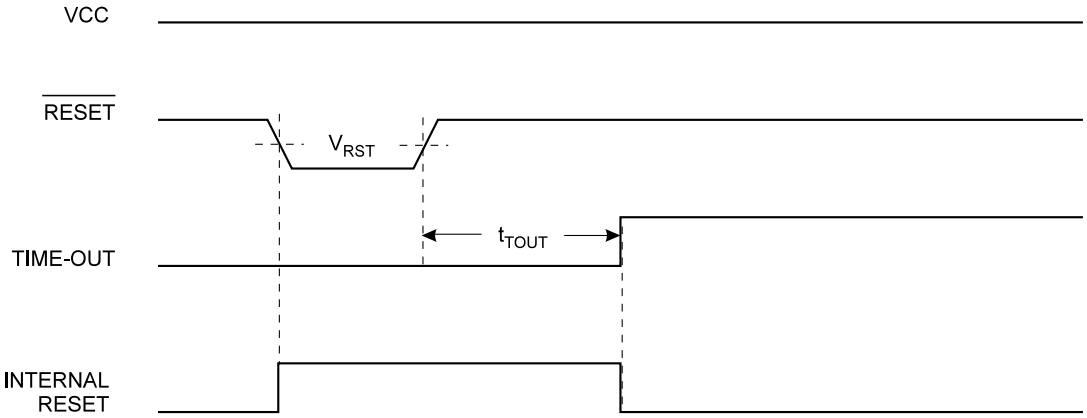
Figure 26. MCU Start-up, $\overline{\text{RESET}}$ Controlled Externally



External Reset

An external reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than 500 ns will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} on its positive edge, the delay timer starts the MCU after the Time-out period t_{TOUT} has expired.

Figure 27. External Reset During Operation

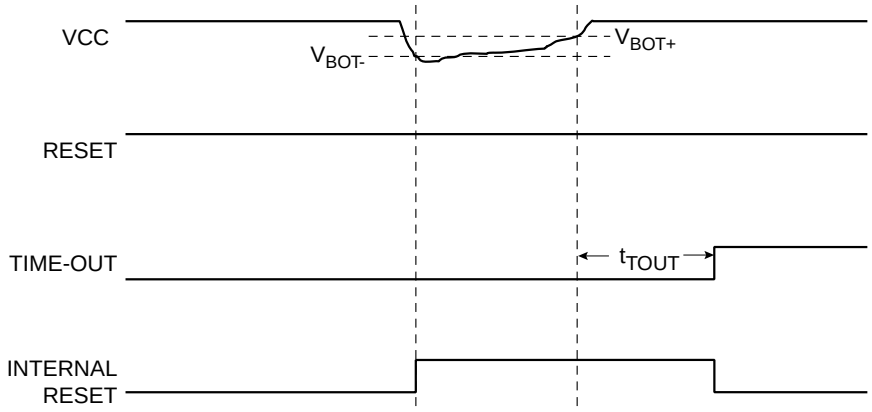


Brown-out Detection

ATmega161 has an on-chip brown-out detection (BOD) circuit for monitoring the V_{CC} level during the operation. The BOD circuit can be enabled/disabled by the fuse BODEN. When BODEN is enabled (BODEN programmed), and V_{CC} decreases to a value below the trigger level, the brown-out reset is immediately activated. When V_{CC} increases above the trigger level, the brown-out reset is deactivated after a delay. The delay is defined by the user in the same way as the delay of POR signal, in Table 4. The trigger level for the BOD can be selected by the fuse BODLEVEL to be 2.7V (BODLEVEL unprogrammed), or 4.0V ((BODLEVEL programmed). The trigger level has a hysteresis of 50 mV to ensure spike free brown-out detection.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than 9 µs for trigger level 4.0V, 21 µs for trigger level 2.7V (typical values).

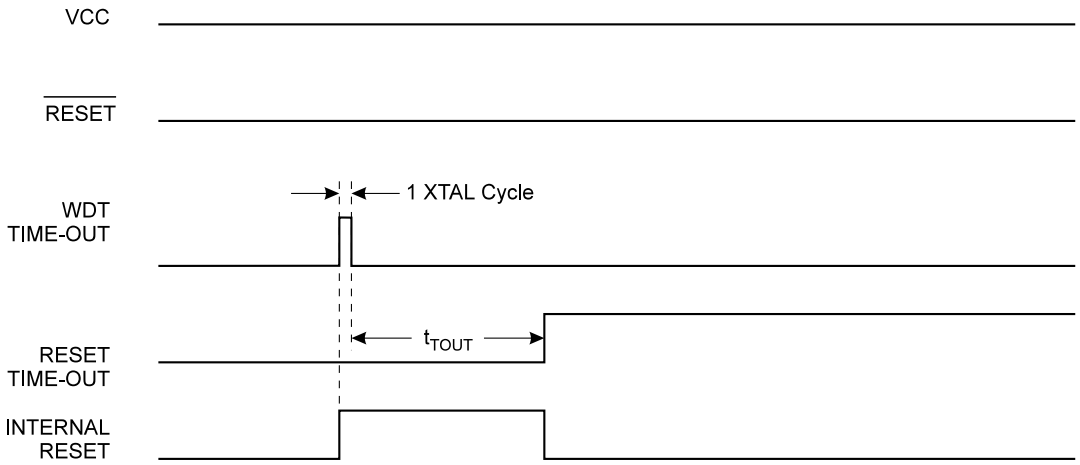
Figure 28. Brown-out Reset During Operation



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 page 52 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)	-	-	-	-	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	See bit description				

- **Bits 7..4 - Res: Reserved Bits**

These bits are reserved bits in the ATmega161 and always read as zero.

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

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

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

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

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

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

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

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

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

Interrupt Handling

The ATmega161 has two 8-bit Interrupt Mask control registers; GIMSK – General Interrupt Mask register and TIMSK – Timer/Counter Interrupt Mask register.

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared (zero) and all interrupts are disabled. The user software can set (one) the I-bit to enable nested interrupts. The I-bit is set (one) when a Return from Interrupt instruction – RETI – is executed.

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.

If an interrupt condition occurs when the corresponding interrupt enable bit is cleared (zero), the interrupt flag will be set and remembered until the interrupt is enabled, or the flag is cleared by software.

If one or more interrupt conditions occur when the global interrupt enable bit is cleared (zero), the corresponding interrupt flag(s) will be set and remembered until the global interrupt enable bit is set (one), and will be executed by order of priority.

Note that external level interrupt does not have a flag, and will only be remembered for as long as the interrupt condition is present.

Note that the status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt routine. This must be handled by software.

Interrupt Response Time

The interrupt execution response for all the enabled AVR interrupts is 4 clock cycles minimum. After 4 clock cycles the program vector address for the actual interrupt handling routine is executed. During this 4 clock cycle period, the Program Counter (13 bits) is pushed onto the Stack. The vector is normally 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. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by 4 clock cycles.

A return from an interrupt handling routine takes 4 clock cycles. During these 4 clock cycles, the Program Counter (2 bytes) is popped back from the Stack, the Stack Pointer is incremented by 2, and the I flag in SREG is set. When *AVR* exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

General Interrupt Mask Register – GIMSK

Bit	7	6	5	4	3	2	1	0	
$3B ($5B)	**INT1**	**INT0**	**INT2**	**-**	**-**	**-**	**-**	**-**	**GIMSK**
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control1 bits 1/0 (ISC11 and ISC10) in the MCU general Control Register (MCUCR) define whether the external interrupt is activated on rising and/or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of External Interrupt Request 1 is executed from program memory address $004. See also “External Interrupts”.

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

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control0 bits 1/0 (ISC01 and ISC00) in the MCU general Control Register (MCUCR) defines whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of External Interrupt Request 0 is executed from program memory address $002. See also “External Interrupts.”

- **Bit 5- INT2: External Interrupt Request 2 Enable**

When the INT2 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is activated. The Interrupt Sense Control2 bit (ISC02 in the Extended MCU Control Register (EMCUCR) defines whether the external interrupt is activated on rising or falling edge of the INT2 pin. Activity on the pin will cause an interrupt request even if INT2 is configured as an output. The corresponding interrupt of External Interrupt Request 2 is executed from program memory address $006. See also “External Interrupts.”

- **Bits 4..0 - Res: Reserved bits**

These bits are reserved bits in the ATmega161 and always read as zero.

General Interrupt Flag Register – GIFR

Bit	7	6	5	4	3	2	1	0	
$3A ($5A)	INTF1	INTF0	INTF2	-	-	-	-	-	GIFR
Read/Write	R/W	R/W	R/W	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - INTF1: External Interrupt Flag1**

When an event on the INT1 pin triggers an interrupt request, INTF1 becomes set (one). If the I-bit in SREG and the INT1 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $004. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 6 - INTF0: External Interrupt Flag0**

When an event on the INT0 pin triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $002. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 5 - INTF2: External Interrupt Flag2**

When an event on the INT2 pin triggers an interrupt request, INTF2 becomes set (one). If the I-bit in SREG and the INT2 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $006. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bits 4..0 - Res: Reserved bits**

These bits are reserved bits in the ATmega161 and always read as zero.

Timer/counter Interrupt Mask Register – TIMSK

Bit	7	6	5	4	3	2	1	0	
$39 ($59)	TOIE1	OCIE1A	OCIE1B	TOIE2	TICIE1	OCIE2	TOIE0	OCIE0	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 - 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 $012) is executed if an overflow in Timer/Counter1 occurs, i.e., when the TOV1 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 6 - 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 $00e) is executed if a CompareA match in Timer/Counter1 occurs, i.e., when the OCF1A bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 5 - 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 $010) is executed if a CompareB match in Timer/Counter1 occurs, i.e., when the OCF1B bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 4 - 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 $00a) is executed if an overflow in Timer/Counter2 occurs, i.e., when the TOV2 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 3 - 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 $00C) is executed if a capture-triggering event occurs on pin 31, ICP, i.e., when the ICF1 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 2 - OCIE2:Timer/Counter2 Output Compare Match 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 $008) is executed if a Compare2 match in Timer/Counter2 occurs, i.e., when the OCF2 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 1 - 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 $016) is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 0 - OCIE0: Timer/Counter0 Output Compare Match 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 $014) is executed if a Compare0 match in Timer/Counter0 occurs, i.e., when the OCF0 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

Timer/Counter Interrupt Flag Register – TIFR

Bit	7	6	5	4	3	2	1	0	
$38 ($58)	TOV1	OCF1A	OCIFB	TOV2	ICF1	OCF2	TOV0	OCF0	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 - 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 6 - 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 match InterruptA Enable), and the OCF1A are set (one), the Timer/Counter1 Compare A match Interrupt is executed.

- **Bit 5 - 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 B match Interrupt is executed.

- **Bit 4 - TOV2: Timer/Counter2 Overflow Flag**

The bit TOV2 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 SREG I-bit, and TOIE2 (Timer/Counter2 Overflow Interrupt Enable), and TOV2 are set (one), the Timer/Counter2 Overflow interrupt is executed.

- **Bit 3 - 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. When the SREG I-bit, and TICIE1 (Timer/Counter1 Input Capture Interrupt Enable), and ICF1 are set (one), the Timer/Counter1 Capture Interrupt is executed.

- **Bit 2 - OCF2: Output Compare Flag 2**

The OCF2 bit is set (one) when compare match occurs between the 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 match InterruptA Enable), and the OCF2 are set (one), the Timer/Counter2 Compare match Interrupt is executed.

- **Bit 1 - 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.

- **Bit 2 - OCF0: Output Compare Flag 0**

The OCF0 bit is set (one) when compare match occurs between the 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/Counter0 Compare match InterruptA Enable), and the OCF0 are set (one), the Timer/Counter0 Compare match Interrupt is executed.

External Interrupts

The external interrupts are triggered by the INT0, INT1 and INT2 pins. Observe that, if enabled, the interrupts will trigger even if the INT0/INT1/INT2 pins are configured as outputs. This feature provides a way of generating a software interrupt. The external interrupts can be triggered by a falling or rising edge or a low level (INT2 is only an edge triggered interrupt). This is set up as indicated in the specification for the MCU Control Register – MCUCR (INT0/INT1) and EMCUCR (INT2). When the external interrupt is enabled and is configured as level triggered (only INT0/INT1), the interrupt will trigger as long as the pin is held low.

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**	**SRW10**	**SE**	**SM1**	**ISC11**	**ISC10**	**ISC01**	**ISC00**	**MCUCR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

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

When the SRE bit is set (one), the external data memory interface is enabled, and the pin functions AD0-7 (Port A), A8-15 (Port C), ALE (Port E), $\overline{WR}$ and $\overline{RD}$ (Port D) are activated as the alternate pin functions. The SRE bit overrides any pin direction settings in the respective data direction registers. See Figure 51 – Figure 54 for a description of the external memory pin functions. When the SRE bit is cleared (zero), the external data memory interface is disabled, and the normal pin and data direction settings are used.

- **Bit 6 - SRW10: External SRAM Wait State**

The SRW10 bit is used to set up extra wait states in the external memory interface. See “Interface to external memory” on page 72 for a detailed description.

- **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.

- **Bit 4 - SM1: Sleep Mode Select bit 1**

The SM1 bit together with the SM0 control bit in EMCUCR selects between the three available sleep modes as shown in the following table.

Table 6. Sleep Mode Select

SM1	SM0	Sleep Mode
0	0	Idle Mode
0	1	Reserved
1	0	Power-down
1	1	Power Save

- **Bits 3, 2 - ISC11, ISC10: Interrupt Sense Control 1 bit 1 and bit 0**

The External Interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask in the GIMSK are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 7. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 7. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

Note: When changing the ISC11/ISC10 bits, INT1 must be disabled by clearing its Interrupt Enable bit in the GIMSK Register. Otherwise an interrupt can occur when the bits are changed.

- **Bit 1, 0 - ISC01, ISC00: Interrupt Sense Control 0 bit 1 and bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask is set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 8. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 8. Interrupt 0 Sense Control

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

Note: When changing the ISC01/ISC00 bits, INT0 must be disabled by clearing its Interrupt Enable bit in the GIMSK Register. Otherwise an interrupt can occur when the bits are changed.

Extended MCU Control Register – EMCUCR

The Extended MCU Control Register contains control bits for external interrupt 2, sleep mode bit and control bits for the external memory interface.

Bit	7	6	5	4	3	2	1	0	
$36 ($56)	**SM0**	**SRL2**	**SRL1**	**SRL0**	**SRW01**	**SRW00**	**SRW11**	**ISC2**	**EMCUCR**
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 - SM0: Sleep mode bit 0**

When this bit is set (one) and sleep mode bit 1 (SM1) in MCUCR is set, Power Save Mode is selected as sleep mode. Refer to page 34 for a detailed description of the sleep modes.

- **Bit 6..4 - SRL2, SRL1, SRL0: External SRAM limit**

It is possible to configure different wait-states for different external memory addresses in ATmega161. The SRL2 – SRL0 bits are used to define at which address the different wait-states will be configured. See “Interface to external memory” on page 72 for a detailed description.

- **Bit 3..1 - SRW01, SRW00, SRW11: External SRAM wait-state select bits.**

The SRW01, SRW00 and SRW11 bits are used to set up extra wait states in the external memory interface. See “Interface to external memory” on page 72 for a detailed description.

- **Bit 0 - ISC2: Interrupt Sense Control 2**

The external interrupt 2 is activated by the external pin INT2 if the SREG I-flag and the corresponding interrupt mask in the GIMSK are set. If ISC2 is cleared (zero) a falling edge on INT2 activates the interrupt. If ISC2 is set (one) a rising edge on INT2 activates the interrupt. Edges on INT2 are registered asynchronously. Pulses on INT2 wider than 50 ns will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt.

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 bit in the MCUCR register and SM0 bit in the EMCUCR register select which sleep mode (Idle, Power-down, or Power Save) will be activated by the SLEEP instruction, see Table 6. If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU awakes. The CPU is then halted for 4 cycles, it executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the register file, SRAM, and I/O memory are unaltered. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector

Idle Mode

When the SM1/SM0 bits are set to 00, the SLEEP instruction makes the MCU enter the Idle Mode, stopping the CPU but allowing SPI, UARTs, Analog Comparator, 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.

Power-down Mode

When the SM1/SM0 bits are set to 10, the SLEEP instruction makes the MCU enter the Power-down Mode. In this mode, the external oscillator is stopped, while the external interrupts and the Watchdog (if enabled) continue operating. Only an external reset, a watchdog reset (if enabled), an external level interrupt on INT0 or INT1, or an external edge interrupt on INT2 can wake-up the MCU.

If INT2 is used for wake-up from power-down mode, the edge is remembered until the MCU wakes up.

If a level triggered interrupt is used for wake-up from power-down mode, the changed level must be held for some time to wake-up the MCU. This makes the MCU less sensitive to noise. The changed level is sampled twice by the watchdog oscillator clock, and if the input has the required level during this time, the MCU will wake-up. The period of the watchdog oscillator is 1 µs (nominal) at 5.0V and 25C. The frequency of the watchdog oscillator is voltage dependent as shown in the Electrical Characteristics section.

When waking up from Power-down Mode, there is a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is defined by the same CKSEL fuses that define the reset time-out period. The wake-up period is equal to the clock counting part of the reset period, as shown in Table 4. If the wake-up condition disappears before the MCU wakes up and starts to execute, e.g. a low level on INT0 is not held long enough, the interrupt causing the wake-up will not be executed.

Power Save Mode

When the SM1/SM0 bits are 11, the SLEEP instruction makes the MCU enter the Power Save Mode. This mode is identical to Power-down, with one exception:

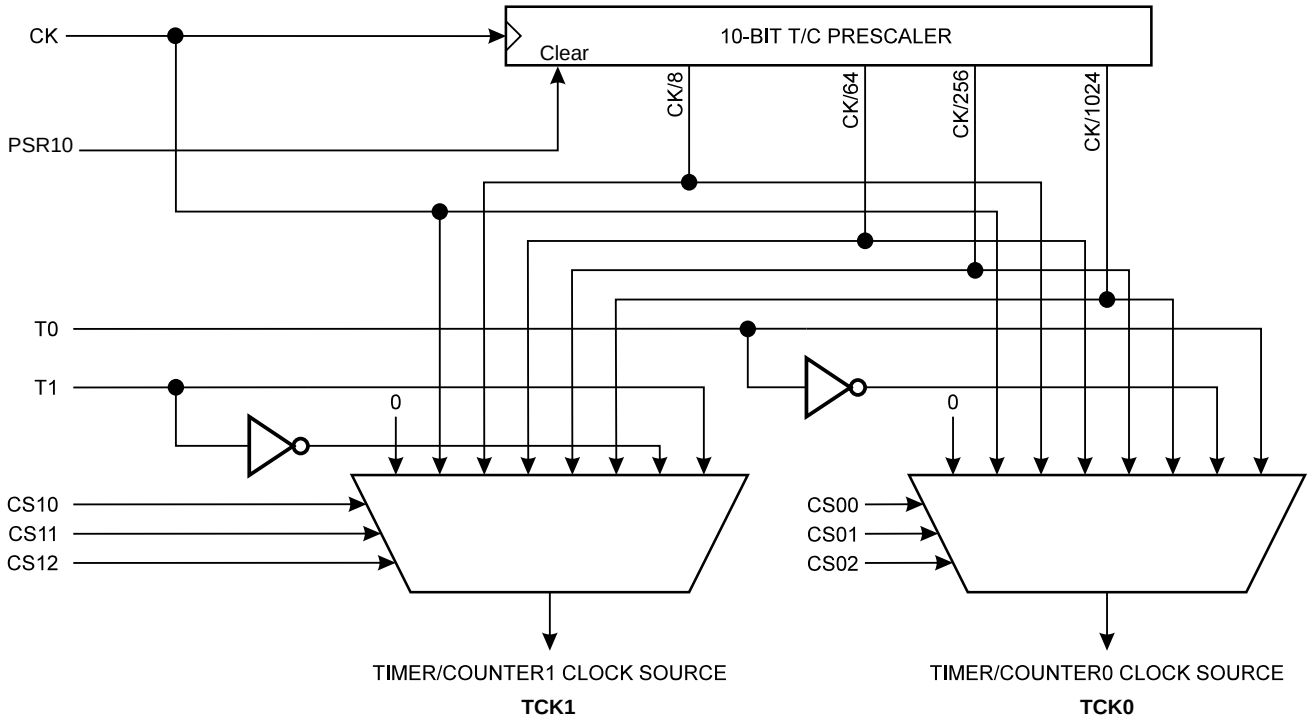
If Timer/Counter2 is clocked asynchronously, i.e. the AS2 bit in ASSR is set, Timer/Counter2 will run during sleep. In addition to the Power-down wake-up sources, the device can also wake-up from either Timer Overflow or Output Compare event from Timer/Counter2 if the corresponding Timer/Counter2 interrupt enable bits are set in TIMSK and the global interrupt enable bit in SREG is set.

Timer/Counters

The ATmega161 provides three general purpose Timer/Counters – two 8-bit T/Cs and one 16-bit T/C. Timer/Counter2 can optionally be asynchronously clocked from an external oscillator. This oscillator is optimized for use with a 32.768 kHz watch crystal, enabling use of Timer/Counter2 as a Real Time Clock (RTC). Timer/Counters 0 and 1 have individual prescaling selection from the same 10-bit prescaling timer. Timer/Counter2 has its own prescaler. Both these prescalers can be reset by setting the corresponding control bits in the Special Functions IO Register (SFIOR). Refer to page 36 for a detailed description. These Timer/Counters can either be used as a timer with an internal clock time-base or as a counter with an external pin connection which triggers the counting.

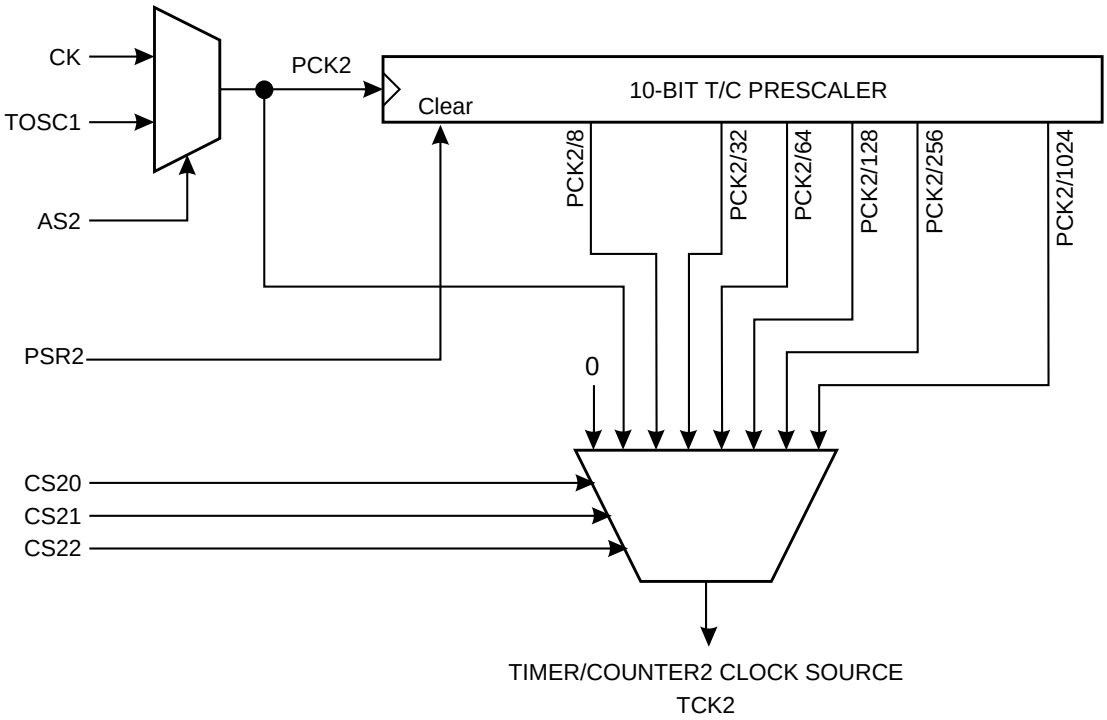
Timer/Counter Prescalers

Figure 30. Prescaler for Timer/Counter0 and 1



For Timer/Counters 0 and 1, the four prescaled selections are: CK/8, CK/64, CK/256 and CK/1024, where CK is the oscillator clock. For the two Timer/Counters 0 and 1, CK, external source, and stop, can also be selected as clock sources. Setting the PSR10 bit in SFIOR resets the prescaler. This allows the user to operate with a predictable prescaler. Note that Timer/Counter1 and Timer/Counter 0 share the same prescaler and a prescaler reset will affect both Timer/Counters.

Figure 31. Timer/Counter2 Prescaler



The clock source for Timer/Counter2 prescaler is named PCK2. PCK2 is by default connected to the main system clock CK. By setting the AS2 bit in ASSR, Timer/Counter2 is asynchronously clocked from the PD4(TOSC1) pin. This enables use of Timer/Counter2 as a Real Time Clock (RTC). When AS2 is set, pins PD4(TOSC1) and PD5(TOSC2) are disconnected from Port D. A crystal can then be connected between the PD4(TOSC1) and PD5(TOSC2) pins to serve as an independent clock source for Timer/Counter2. The oscillator is optimized for use with a 32.768 kHz crystal. Alternatively, an external clock signal can be applied to PD4(TOSC1). The frequency of this clock must be lower than one fourth of the CPU clock and not higher than 256 kHz. Setting the PSR2 bit in SFIOR resets the prescaler. This allows the user to operate with a predictable prescaler.

Special Function IO Register – SFIOR

Bit	7	6	5	4	3	2	1	0	
$30 ($50)	-	-	-	-	-	-	**PSR2**	**PSR10**	**SFIOR**
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

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

These bits are reserved bits in the ATmega161 and always read as zero.

- **Bit 1 - PSR2: Prescaler Reset Timer/Counter2**

When this bit is set (one) the Timer/Counter2 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. This bit will always be read as zero if Timer/Counter2 is clocked by the internal CPU clock. If this bit is written when Timer/Counter2 is operating in asynchronous mode however, the bit will remain as one until the prescaler has been reset. See “Asynchronous Operation of Timer/Counter2” on page 43 for a detailed description of asynchronous operation.

- **Bit 0 - PSR10: Prescaler Reset Timer/Counter1 and Timer/Counter0**

When this bit is set (one) the Timer/Counter1 and Timer/Counter0 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers. This bit will always be read as zero.

8-bit Timers/Counters T/C0 and T/C2

Figure 32 shows the block diagram for Timer/Counter0. Figure 33 shows the block diagram for Timer/Counter2.

Figure 32. Timer/Counter0 Block Diagram

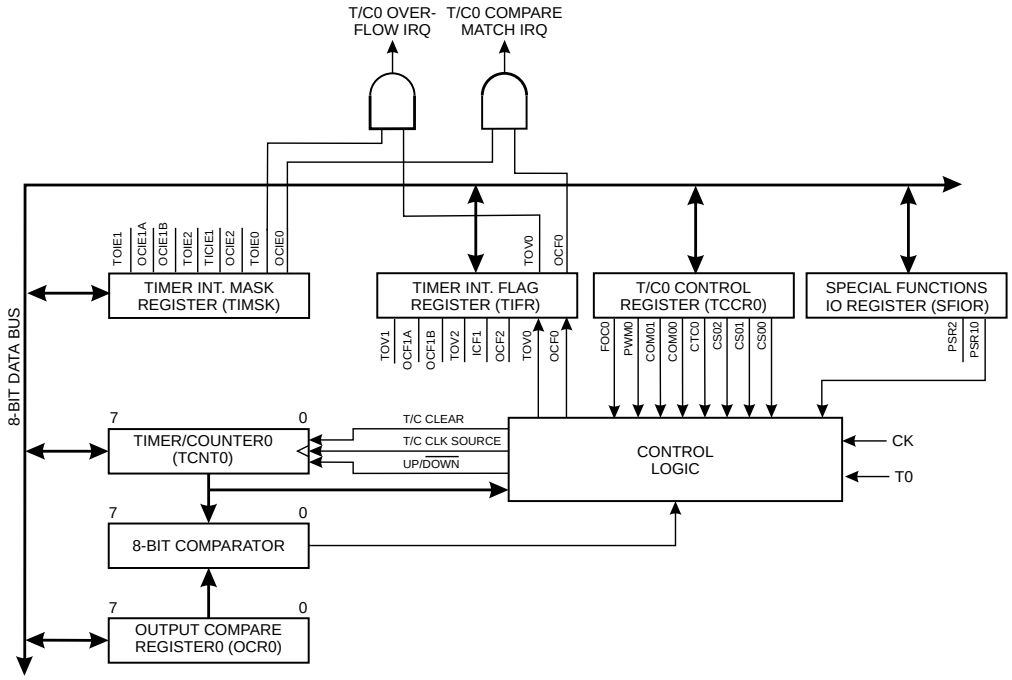
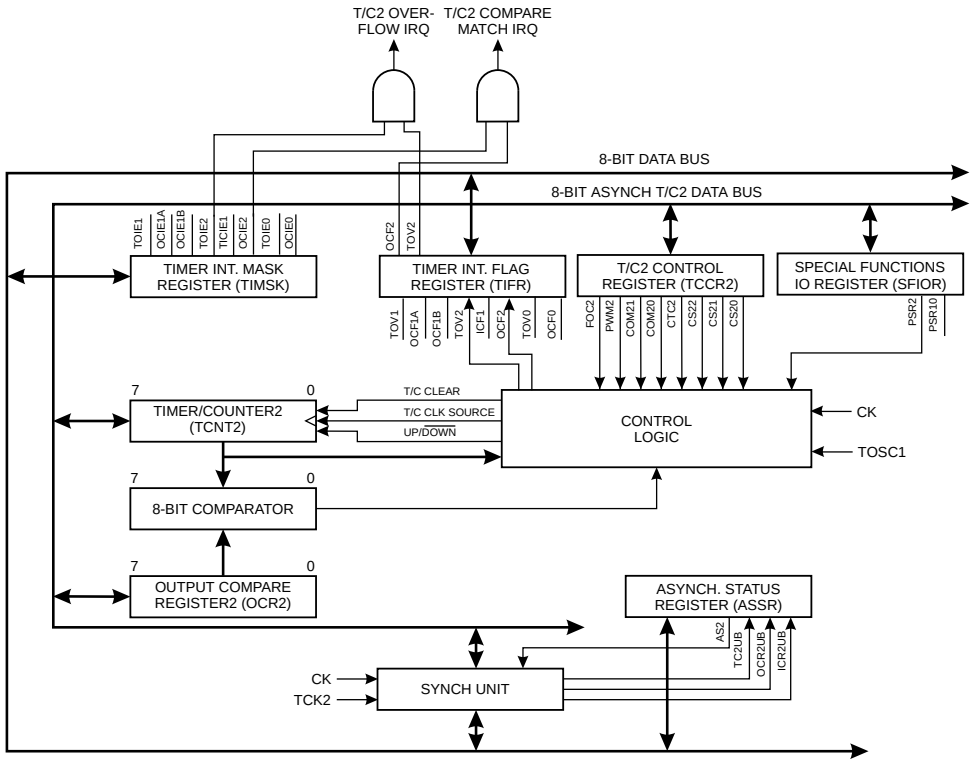


Figure 33. Timer/Counter2 Block Diagram



The 8-bit Timer/Counter0 can select clock source from CK, prescaled CK, or an external pin.

The 8-bit Timer/Counter2 can select clock source from CK, prescaled CK or external TOSC1.

Both Timers/Counters can be stopped as described in section “Timer/Counter0 Control Register – TCCR0” on page 38 and “Timer/Counter2 Control Register – TCCR2” on page 38

The various status flags (overflow and compare match) are found in the Timer/Counter Interrupt Flag Register – TIFR. Control signals are found in the Timer/Counter Control Register – TCCR0 and TCCR2. The interrupt enable/disable settings are found in the Timer/Counter Interrupt Mask Register – TIMSK.

When Timer/Counter0 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 both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter0 useful for lower speed functions or exact timing functions with infrequent actions.

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 41 for a detailed description on this function.

Timer/Counter0 Control Register – TCCR0

Bit	7	6	5	4	3	2	1	0	
$33 ($53)	**FOC0**	**PWM0**	**COM01**	**COM00**	**CTC0**	**CS02**	**CS01**	**CS00**	**TCCR0**
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 Control Register – TCCR2

Bit	7	6	5	4	3	2	1	0	
$27 ($47)	**FOC2**	**PWM2**	**COM21**	**COM20**	**CTC2**	**CS22**	**CS21**	**CS20**	**TCCR2**
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 - FOC0/FOC2: Force Output Compare**

Writing a logical one to this bit, forces a change in the compare match output pin PB0 (Timer/Counter0) and PB1 (Timer/Counter2) according to the values already set in COMn1 and COMn0. If the COMn1 and COMn0 bits are written in the same cycle as FOC0/FOC2, the new settings will not take effect until next compare match or Forced Output Compare match occurs. The Force Output Compare bit can be used to change the output pin without waiting for a compare match in the timer. The automatic action programmed in COMn1 and COMn0 happens as if a Compare Match had occurred, but no interrupt is generated and the Timer/Counters will not be cleared even if CTC0/CTC2 is set. The FOC0/FOC2 bits will always be read as zero. The setting of the FOC0/FOC2 bits has no effect in PWM mode.

- **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 41.

- **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/Counter0 or Timer/Counter2. Output pin actions affect pins PB0(OC0) or PB1(OC2). This is an alternative function to an I/O port, and the corresponding direction control bit must be set (one) to control an output pin. The control configuration is shown in Table 9.

Table 9. Compare Mode Select

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

Notes: 1. In PWM mode, these bits have a different function. Refer to Table 12 for a detailed description.
2. n = 0 or 2

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

When the CTC0 or CTC2 control bit is set (one), Timer/Counter0 or Timer/Counter2 is reset to $00 in the CPU clock cycle after a compare match. If the control bit is cleared, Timer/Counter continues counting and is unaffected by a compare match. When a prescaling of 1 is used, and the compare register is set to C, the timer will count as follows if CTC0/CTC2 is set:

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

In PWM mode, this bit has a different function. If the CTC0 or CTC2 bit is cleared in PWM mode, the Timer/Counter acts as an up/down counter. If the CTC0 or CTC2 bit is set (one), the Timer/Counter wraps when it reaches $FF. Refer to page 41 for a detailed description.

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

The Clock Select bits 2,1 and 0 define the prescaling source of Timer/Counter0 and Timer/Counter2.

Table 10. Clock 0 Prescale Select

CS02	CS01	CS00	Description
0	0	0	Stop, the Timer/Counter0 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 PB0(T0), falling edge
1	1	1	External Pin PB0(T0), rising edge

Table 11. Clock 2 Prescale Select

CS22	CS21	CS20	Description
0	0	0	Stop, the Timer/Counter2 is stopped.
0	0	1	PCK2
0	1	0	PCK2/8
0	1	1	PCK2/32
1	0	0	PCK2/64
1	0	1	PCK2/128
1	1	0	PCK2/256
1	1	1	PCK2/1024

The Stop condition provides a Timer Enable/Disable function. The prescaled modes are scaled directly from the CK oscillator clock for Timer/Counter0 and PCK2 for Timer/Counter2. If the external pin modes are used for Timer/Counter0, transitions on PB0/(T0) will clock the counter even if the pin is configured as an output. This feature can give the user SW control of the counting.

Timer Counter0 – TCNT0

Bit	7	6	5	4	3	2	1	0	
$32 ($52)	**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	
$23 ($43)	**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 registers contain the value of the Timer/Counters.

Both Timer/Counters is 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 following the write operation.

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	
$22 ($42)	**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 contains 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 the Timer/Counter counts to the OCR value. A software write that sets 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.

Timer/Counter 0 and 2 in PWM Mode

When PWM mode is selected, the Timer/Counter either wraps (overflows) when it reaches $FF or it acts as an up/down counter.

If the up/down mode is selected, the Timer/Counter and the Output Compare Registers – OCR0 or OCR2 form an 8-bit, free-running, glitch-free and phase correct PWM with outputs on the PB0(OC0/PWM0) or PB1(OC2/PWM2) pin.

If the overflow mode is selected, the Timer/Counter and the Output Compare Registers – OCR0 or OCR2 form an 8-bit, free-running and glitch-free PWM, operating with twice the speed of the up/down counting mode.

PWM Modes (Up/Down and Overflow)

The two different PWM modes are selected by the CTC0 or CTC2 bit in the Timer/Counter Control Registers – TCCR0 or TCCR2 respectively.

If CTC0/CTC2 is cleared and PWM mode is selected, the Timer/Counter acts as an up/down counter, counting up from $00 to $FF, where 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 PB0(OC0/PWM0) or PB1(OC2/PWM2) pin is set or cleared according to the settings of the COMn1/COMn0 bits in the Timer/Counter Control Registers TCCR0 or TCCR2.

If CTC0/CTC2 is set and PWM mode is selected, the Timer/Counters will wrap and start counting from $00 after reaching $FF. The PB0(OC0/PWM0) or PB1(OC2/PWM2) pin will be set or cleared according to the settings of COMn1/COMn0 on a Timer/Counter overflow or when the counter value matches the contents of the Output Compare Register. Refer to Table 12 for details.

Table 12. Compare Mode Select in PWM Mode

CTCn	COMn1	COMn0	Effect on Compare Pin	Frequency
0	0	0	Not connected	
0	0	1	Not connected	
0	1	0	Cleared on compare match, up-counting. Set on compare match, down-counting (non-inverted PWM).	$f_{TCK0/2}/510$
0	1	1	Cleared on compare match, down-counting. Set on compare match, up-counting (inverted PWM).	$f_{TCK0/2}/510$
1	0	0	Not connected	
1	0	1	Not connected	
1	1	0	Cleared on compare match, set on overflow.	$f_{TCK0/2}/256$
1	1	1	Set on compare match, cleared on overflow.	$f_{TCK0/2}/256$

Note: n = 0 or 2

Note that in PWM mode, the value to be written to the Output Compare Register is first transferred to a temporary location, and then latched into the OCR 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 and Figure 35 for examples.

Figure 34. Effects of Unsynchronized OCR Latching in up/down mode

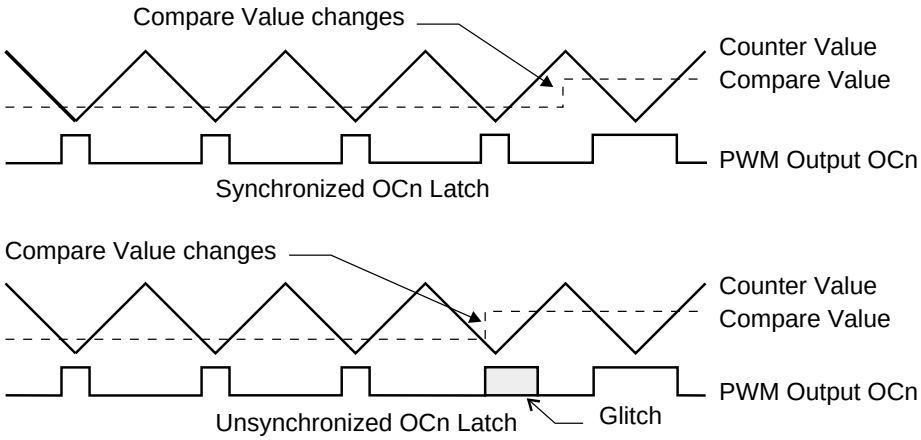
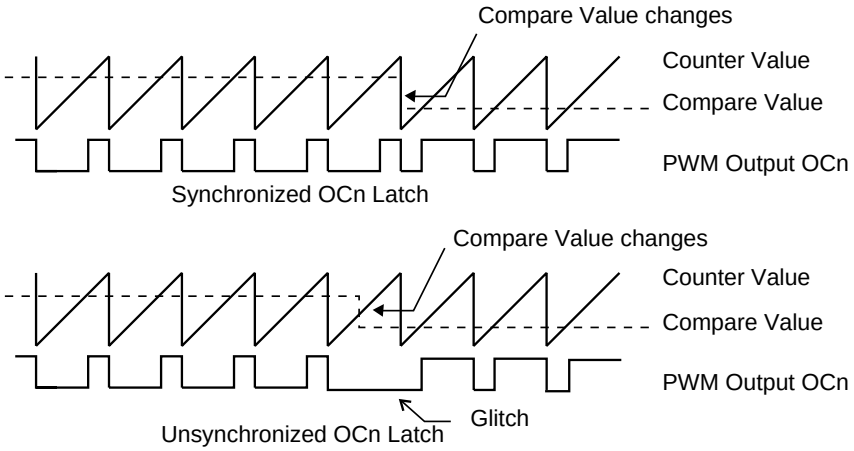


Figure 35. Effects of Unsynchronized OCR Latching in overflow mode.



Note: n = 0 or 2 (Figure 34 and Figure 35)

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

When the Output Compare Register contains $00 or $FF, and the up/down PWM mode is selected, the output PB0(OC0/PWM0)/PB1(OC2/PWM2) is updated to low or high on the next compare match according to the settings of COMn1/COMn0. This is shown in Table 13. In overflow PWM mode, the output PB0(OC0/PWM0)/PB1(OC2/PWM2) is held low or high only when the Output Compare Register contains $FF.

Table 13. 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 overflow PWM mode, the table above is only valid for OCRn = $FF.

In up/down PWM mode, the Timer Overflow Flag, TOV0 or TOV2, is set when the counter advances from $00. In overflow PWM mode, the Timer Overflow Flag is set as in normal Timer/Counter mode. Timer Overflow Interrupt0 and 2 operate exactly as in normal Timer/Counter mode, i.e. they are executed when TOV0 or TOV2 are set provided that Timer Overflow Interrupt and global interrupts are enabled. This does also apply to the Timer Output Compare flag and interrupt.

Asynchronous Status Register – ASSR

Bit	7	6	5	4	3	2	1	0	
$26 ($46)	-	-	-	-	AS2	TCN2UB	OCR2UB	TCR2UB	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 ATmega161 and always read as zero.

- **Bit 3 - AS2: Asynchronous Timer/Counter2 mode**

When this bit is cleared (zero) Timer/Counter2 is clocked from the internal system clock, CK. If AS2 is set, the Timer/Counter2 is clocked from the TOSC1 pin. Pins PD4 and PD5 become connected to a crystal oscillator and cannot be used as general I/O pins. When the value of this bit is changed the contents of TCNT2, OCR2 and TCCR2 might get corrupted.

- **Bit 2 - TCN2UB: Timer/Counter2 Update Busy**

When Timer/Counter2 operates asynchronously and TCNT2 is written, this bit becomes set (one). When TCNT2 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that TCNT2 is ready to be updated with a new value.

- **Bit 1 - OCR2UB: Output Compare Register2 Update Busy**

When Timer/Counter2 operates asynchronously and OCR2 is written, this bit becomes set (one). When OCR2 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that OCR2 is ready to be updated with a new value.

- **Bit 0 - TCR2UB: Timer/Counter Control Register2 Update Busy**

When Timer/Counter2 operates asynchronously and TCCR2 is written, this bit becomes set (one). When TCCR2 has been updated from the temporary storage register, this bit is cleared (zero) by hardware. A logical zero in this bit indicates that TCCR2 is ready to be updated with a new value.

If a write is performed to any of the three Timer/Counter2 registers while its update busy flag is set (one), the updated value might get corrupted and cause an unintentional interrupt to occur.

The mechanisms for reading TCNT2, OCR2, and TCCR2 are different. When reading TCNT2, the actual timer value is read. When reading OCR2 or TCCR2, the value in the temporary storage register is read.

Asynchronous Operation of Timer/Counter2

When Timer/Counter2 operates asynchronously, some considerations must be taken.

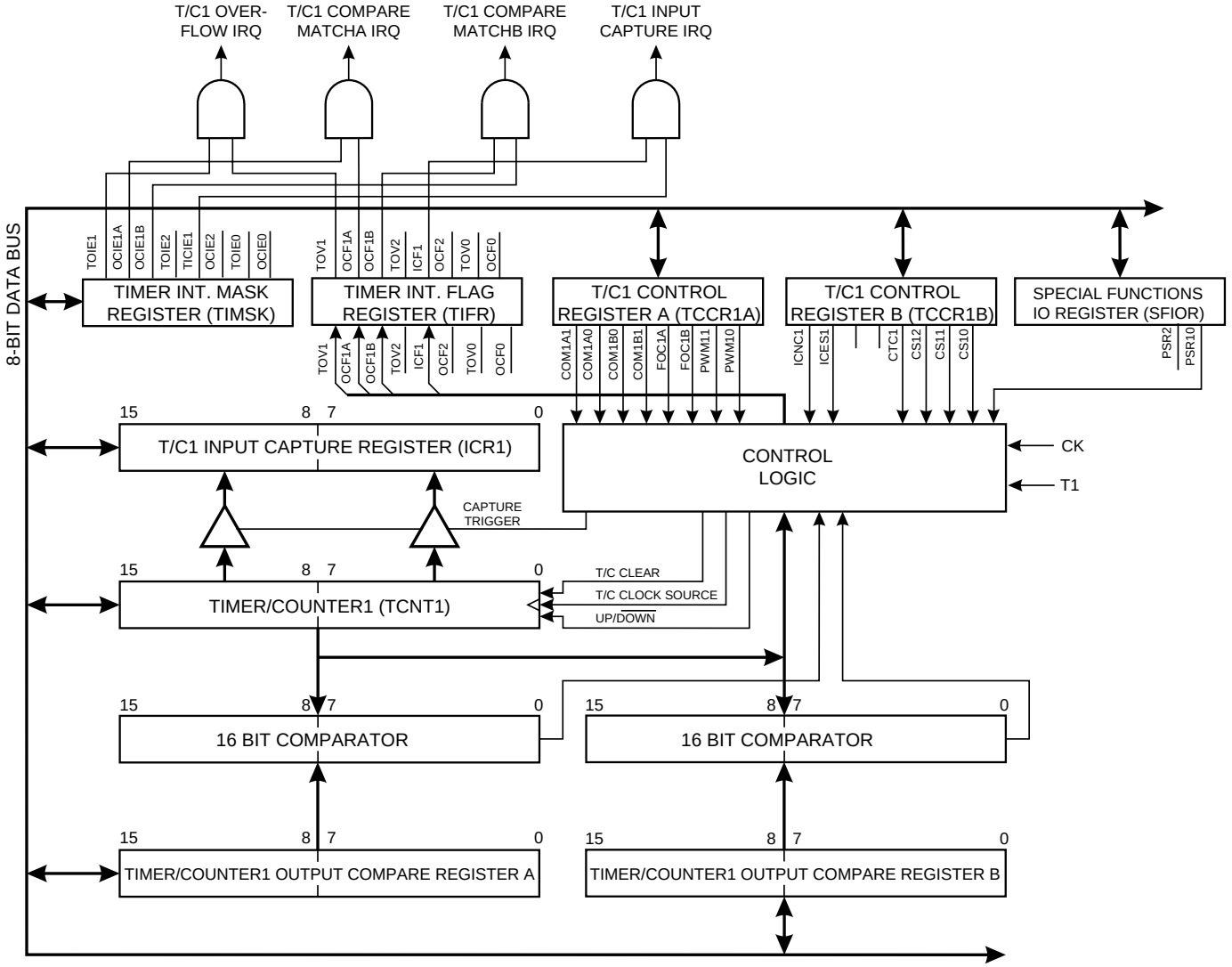
- Warning: When switching between asynchronous and synchronous clocking of Timer/Counter2, the timer registers; TCNT2, OCR2 and TCCR2 might get corrupted. A safe procedure for switching clock source is:
  1. Disable the Timer/Counter2 interrupts by clearing OCIE2 and TOIE2.
  2. Select clock source by setting AS2 as appropriate.
  3. Write new values to TCNT2, OCR2, and TCCR2.
  4. To switch to asynchronous operation: Wait for TCN2UB, OCR2UB, and TCR2UB.
  5. Enable interrupts, if needed.
- The oscillator is optimized for use with a 32,768 Hz watch crystal. An external clock signal applied to this pin goes through the same amplifier having a bandwidth of 256 kHz. The external clock signal should therefore be in the interval 0 Hz - 256 kHz. The frequency of the clock signal applied to the TOSC1 pin must be lower than one fourth of the CPU main clock frequency.

- When writing to one of the registers TCNT2, OCR2, or TCCR2, 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 TCNT2 does not disturb an OCR2 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 Power Save mode after having written to TCNT2, OCR2, or TCCR2, the user must wait until the written register has been updated if Timer/Counter2 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 Compare2 interrupt is used to wake-up the device; Output compare is disabled during write to OCR2 or TCNT2. If the write cycle is not finished (i.e. the MCU enters sleep mode before the OCR2UB bit returns to zero), the device will never get a compare match and the MCU will not wake-up.
- If Timer/Counter2 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 be 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 TCCR2, TCNT2, or OCR2
    2. Wait until the corresponding Update Busy flag in ASSR returns to zero.
    3. Enter Power Save mode
- When asynchronous operation is selected, the 32 kHz oscillator for Timer/Counter2 is always running, except in power-down mode. After a power-up reset or wake-up from power-down, the user should be aware of the fact that this oscillator might take as long as one second to stabilize. Therefore, the contents of all Timer2 registers must be considered lost after a wake-up from power-down, due to the unstable clock signal. The user is advised to wait for at least one second before using Timer/Counter2 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.

Timer/Counter1.

Figure 36 shows the block diagram for Timer/Counter1.

Figure 36. Timer/Counter1 Block Diagram



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 section “Timer/Counter1 Control Register B – TCCR1B” on page 47. 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/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.

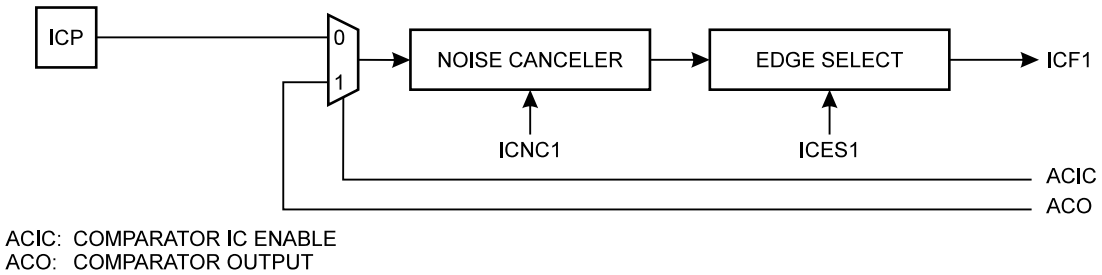
The 16-bit Timer/Counter1 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities makes the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

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.

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 stand-alone PWM with centered pulses. Alternatively, the Timer/Counter1 can be configured to operate at twice the speed in PWM mode, but without centered pulses. Refer to page 50 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 – ICP. 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 section, “The Analog Comparator”, for details on this. The ICP pin logic is shown in Figure 37.

Figure 37. ICP Pin Schematic Diagram



ACIC: COMPARATOR IC ENABLE
ACO: COMPARATOR OUTPUT

If the noise canceler function is enabled, the actual trigger condition for the capture event is monitored over 4 samples, and all 4 must be equal to activate the capture flag.

Timer/Counter1 Control Register A – TCCR1A

Bit	7	6	5	4	3	2	1	0	
$2F ($4F)	**COM1A1**	**COM1A0**	**COM1B1**	**COM1B0**	**FOC1A**	**FOC1B**	**PWM11**	**PWM10**	**TCCR1A**
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 7,6 - 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. This is an alternative function to an I/O port, and the corresponding direction control bit must be set (one) to control an output pin. The control configuration is shown in Table 14.

- **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. This is an alternative function to an I/O port, and the corresponding direction control bit must be set (one) to control an output pin. The following control configuration is given:

Table 14. 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).

X = A or B

In PWM mode, these bits have a different function. Refer to Table 18 for a detailed description.

- **Bit 3 - FOC1A: Force Output Compare1A**

Writing a logical one to this bit, forces a change in the compare match output pin PD5 according to the values already set in COM1A1 and COM1A0. If the COM1A1 and COM1A0 bits are written in the same cycle as FOC1A, the new settings will not take effect until next compare match or forced compare match occurs. The Force Output Compare bit can be used to change the output pin without waiting for a compare match in the timer. The automatic action programmed in COM1A1 and COM1A0 happens as if a Compare Match had occurred, but no interrupt is generated and it will not clear the timer even if CTC1 in TCCR1B is set. The FOC1A bit will always be read as zero. The setting of the FOC1A bit has no effect in PWM mode.

- **Bit 2 - FOC1B: Force Output Compare1B**

Writing a logical one to this bit, forces a change in the compare match output pin PE2 according to the values already set in COM1B1 and COM1B0. If the COM1B1 and COM1B0 bits are written in the same cycle as FOC1B, the new settings will not take effect until next compare match or forced compare match occurs. The Force Output Compare bit can be used to change the output pin without waiting for a compare match in the timer. The automatic action programmed in COM1B1 and COM1B0 happens as if a Compare Match had occurred, but no interrupt is generated. The FOC1B bit will always be read as zero. The setting of the FOC1B bit has no effect in PWM mode.

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

These bits select PWM operation of Timer/Counter1 as specified in Table 15. This mode is described on page 50.

Table 15. 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

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 ICP – input capture pin – as specified. When the ICNC1 bit is set (one), four successive samples are measures on the ICP – input capture pin, 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 – ICP. 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 – ICP.

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

These bits are reserved bits in the ATmega161 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. When a prescaling of 1 is used, and the compareA register is set to C, the timer will count as follows if CTC1 is set:

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

In PWM mode, this bit has a different function. If the CTC1 bit is cleared in PWM mode, the Timer/Counter1 acts as an up/down counter. If the CTC1 bit is set (one), the Timer/Counter wraps when it reaches the TOP value. Refer to page 50 for a detailed description.

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

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

Table 16. 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, falling edge
1	1	1	External Pin T1, 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/Counter1, transitions on PB1/(T1) will clock the counter even if the pin is configured as an output. This feature can give the user SW control of the counting.

Timer/Counter1 Register – 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 and interrupt routines.

- **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.

- **TCNT1 Timer/Counter1 Read:**

When the CPU reads the low byte TCNT1L, the data of the low byte 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.

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 timer 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 ($4B)	MSB								OCR1AH
$2A ($4A)								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 ($49)	MSB								OCR1BH
$28 ($48)								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, OCR1AL or OCR1BL, the TEMP register is simultaneously written to OCR1AH or OCR1BH. Consequently, the high byte OCR1AH or OCR1BH 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 and interrupt routines.

Timer/Counter1 Input Capture Register – ICR1H AND ICR1L

Bit	15	14	13	12	11	10	9	8	
$25 ($45)	MSB								ICR1H
$24 ($44)								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 – ICP – is detected, the current value of the Timer/Counter1 Register – TCNT1 is transferred to the Input Capture Register – ICR1. In the same cycle, 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 and interrupt routine.

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 PE2(OC1B) pins. In this mode the Timer/Counter1 acts as an up/down counter, counting up from $0000 to TOP (see Table 17), where it turns and counts down again to zero before the cycle is repeated. When the counter value matches the contents of the 8,9 or 10 least significant bits (depends of the resolution) of OCR1A or OCR1B, the PD5(OC1A)/PE2(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.

Alternatively, the Timer/Counter1 can be configured to a PWM that operates at twice the speed as in the mode described above. Then the Timer/Counter1 and the Output Compare Register1A – OCR1A and the Output Compare Register1B – OCR1B, form a dual 8, 9 or 10-bit, free-running and glitch-free PWM with outputs on the PD5(OC1A) and PE2(OC1B) pins.

Table 17. Timer TOP Values and PWM Frequency

CTC1	PWM11	PWM10	PWM Resolution	Timer TOP value	Frequency
0	0	1	8-bit	$00FF (255)	$f_{TCK1}/510$
0	1	0	9-bit	$01FF (511)	$f_{TCK1}/1022$
0	1	1	10-bit	$03FF(1023)	$f_{TCK1}/2046$
1	0	1	8-bit	$00FF (255)	$f_{TCK1}/256$
1	1	0	9-bit	$01FF (511)	$f_{TCK1}/512$
1	1	1	10-bit	$03FF(1023)	$f_{TCK1}/1024$

Note: X = A or B

As shown in Table 17, the PWM operates at either 8-, 9- or 10 bits resolution. Note the unused bits in OCR1A, OCR1B and TCNT1 will automatically be written to zero by hardware. I.e. bit 9 to 15 will be set to zero in OCR1A, OCR1B and TCNT1 if the 9-bit PWM resolution is selected. This makes it possible for the user to perform read-modify-write operations in any of the three resolution modes and the unused bits will be treated as don't care.

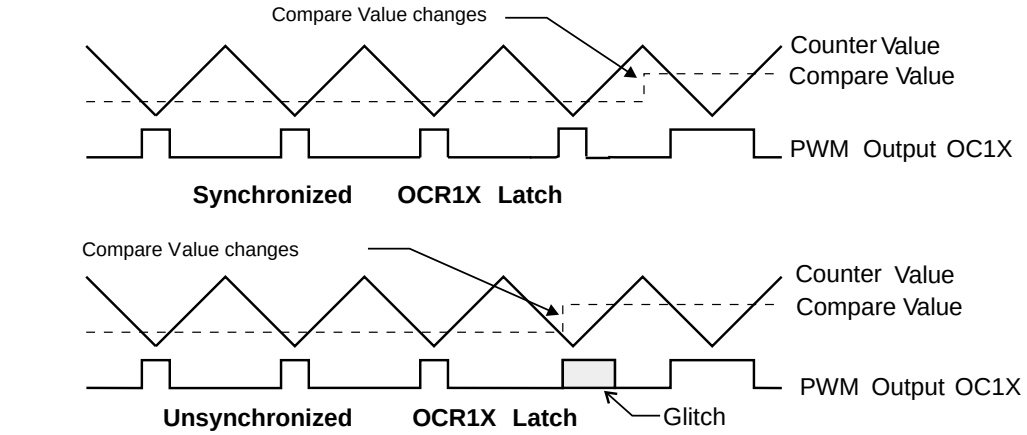
Table 18. Compare1 Mode Select in PWM Mode

CTC1	COM1X1	COM1X0	Effect on OCX1
0	0	0	Not connected
0	0	1	Not connected
0	1	0	Cleared on compare match, up-counting. Set on compare match, down-counting (non-inverted PWM).
0	1	1	Cleared on compare match, down-counting. Set on compare match, up-counting (inverted PWM).
1	0	0	Not connected
1	0	1	Not connected
1	1	0	Cleared on compare match, set on overflow.
1	1	1	Set on compare match, cleared on overflow.

Note: X = A or B

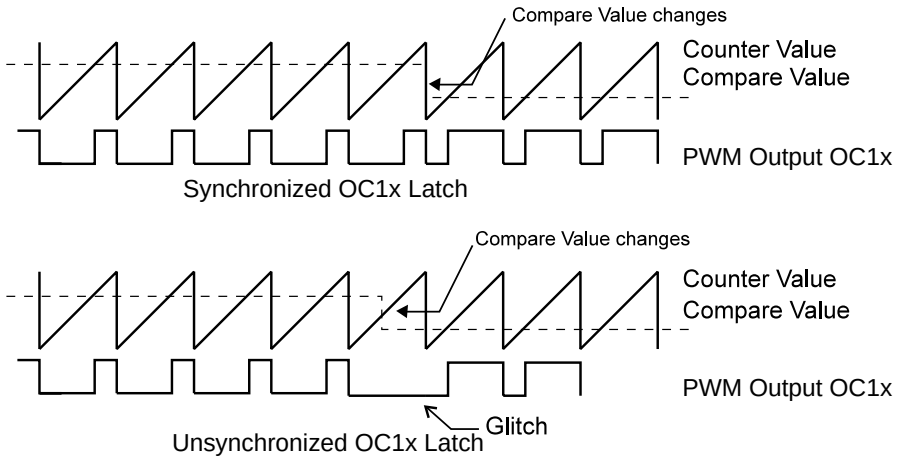
Note that in the PWM mode, the 8, 9 or 10 least significant OCR1A/OCR1B bits (depends of resolution), when written, are transferred to a temporary location. They are latched when Timer/Counter1 reaches the value TOP. This prevents the occurrence of odd-length PWM pulses (glitches) in the event of an unsynchronized OCR1A/OCR1B write. See Figure 38 and Figure 39 for an example in each mode.

Figure 38. Effects on Unsynchronized OCR1 Latching



Note: X = A or B

Figure 39. Effects of Unsynchronized OCR1 Latching in overflow mode.



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 the OCR1X contains $0000 or TOP, and the up/down PWM mode is selected, the output OC1A/OC1B is updated to low or high on the next compare match according to the settings of COM1A1/COM1A0 or COM1B1/COM1B0. This is shown in Table 19. In overflow PWM mode, the output OC1A/OC1B is held low or high only when the Output Compare Register contains TOP.

Table 19. 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 overflow PWM mode, the table above is only valid for OCR1X = TOP.

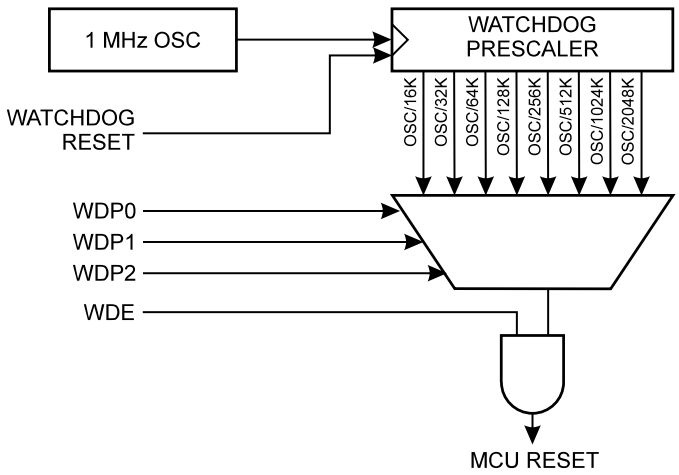
In up/down PWM mode, the Timer Overflow Flag1, TOV1, is set when the counter advances from $0000. In overflow PWM mode, the Timer Overflow flag is set as in normal Timer/Counter mode. Timer Overflow Interrupt1 operates exactly as in normal Timer/Counter mode, i.e. it is executed when TOV1 is set provided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

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, see Table 20 on page 53 for a detailed description. The WDR – Watchdog Reset – instruction resets the Watchdog Timer. Eight different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog reset, the ATmega161 resets and executes from the reset vector. For timing details on the Watchdog reset, refer to page 27.

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

Figure 40. Watchdog Timer



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 ATmega161 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 Time-out Periods are shown in Table 20.

Table 20. Watch Dog Timer Prescale Select

WDP2	WDP1	WDP0	Number of WDT Oscillator cycles	Typical time-out at Vcc = 3.0V	Typical time-out at Vcc = 5.0V
0	0	0	16K cycles	47 ms	15 ms
0	0	1	32K cycles	94 ms	30 ms
0	1	0	64K cycles	0.19 s	60 ms
0	1	1	128K cycles	0.38 s	0.12 s
1	0	0	256K cycles	0.75 s	0,24 s
1	0	1	512K cycles	1.5 s	0.49 s
1	1	0	1,024K cycles	3.0 s	0.97 s
1	1	1	2,048K cycles	6.0 s	1.9 s

Note: The frequency of the watchdog oscillator is voltage dependent as shown in the Electrical Characteristics section.
The WDR – Watchdog Reset – instruction should always be executed before the Watchdog Timer is enabled. This ensures that the reset period will be in accordance with the Watchdog Timer prescale settings. If the Watchdog Timer is enabled without reset, the watchdog timer may not start counting from zero.

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 - 4 ms, depending on the V_{CC} voltages. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains code that writes the EEPROM, some precaution must be taken. In heavily filtered power supplies, V_{CC} is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. CPU operation under these conditions is likely cause the program counter to perform unintentional jumps and eventually execute the EEPROM write code. To secure EEPROM integrity, the user is advised to use an external under-voltage reset circuit in this case.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM Control Register for details on this.

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

EEPROM Address Register – EEARH and EEARL

Bit	15	14	13	12	11	10	9	8	
$1F ($3F)	-	-	-	-	-	-	-	**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	R	R	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	X	
	X	X	X	X	X	X	X	X	

- **Bits 15..9 - Res: Reserved bits**

These bits are reserved bits in the ATmega161 and will always read as zero.

- **Bits 8..0 - EEAR8..0: EEPROM Address**

The EEPROM Address Registers – EEARH and EEARL specify the EEPROM address in the 512 bytes EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 511. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

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.

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/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7..4 - Res: Reserved bits**

These bits are reserved bits in the ATmega161 and will always 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 generates a constant interrupt 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 an 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 not essential):

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.

Caution: An interrupt between step 4 and step 5 will make the write cycle fail, since the EEPROM Master Write Enable will time-out. If an interrupt routine accessing the EEPROM is interrupting another EEPROM access, the EEAR or EEDR register will be modified, causing the interrupted EEPROM access to fail. It is recommended to have the global interrupt flag cleared during the 4 last steps to avoid these problems.

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 and 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):

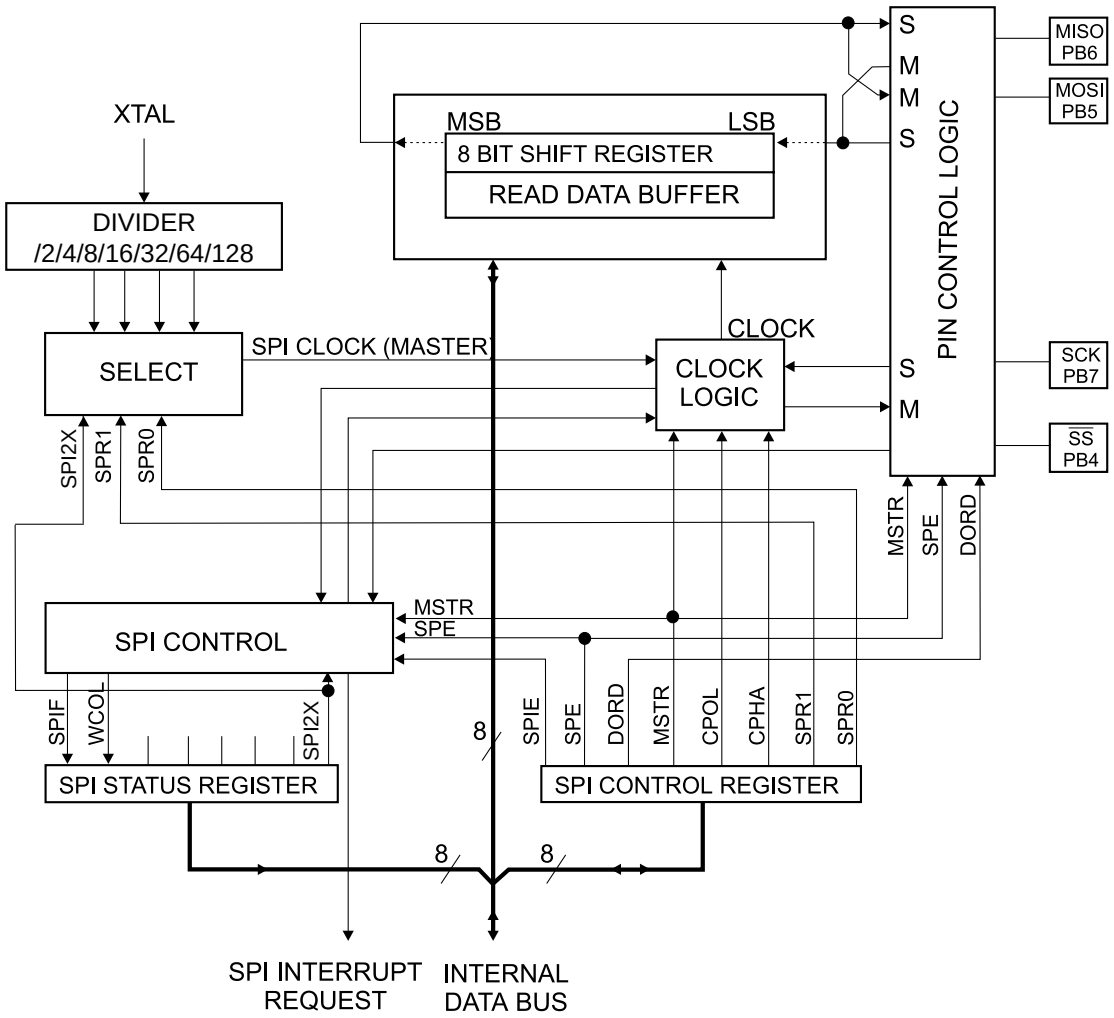
1. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} Reset Protection circuit can be applied.
2. Keep the AVR core 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.

Serial Peripheral Interface – SPI

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the ATmega161 and peripheral devices or between several AVR devices. The ATmega161 SPI features include the following:

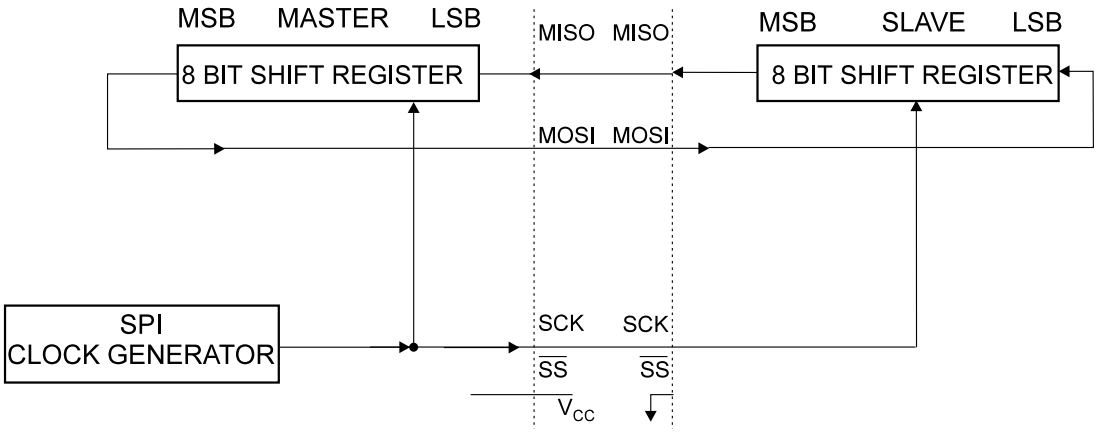
- Full-duplex, 3-wire Synchronous Data Transfer
- Master or Slave Operation
- LSB First or MSB First Data Transfer
- Seven Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wake-up from Idle Mode (Slave Mode Only)
- Double Speed (CK/2) Master SPI Mode

Figure 41. SPI Block Diagram



The interconnection between master and slave CPUs with SPI is shown in Figure 42. The PB7(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 PB5(MOSI) pin and into the PB5(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, PB4($\overline{SS}$), is set low to select an individual slave SPI device. 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 42. 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 42. SPI Master-slave Interconnection



The system is single buffered in the transmit direction and double buffered in the receive direction. This means that bytes to be transmitted cannot be written to the SPI Data Register before the entire shift cycle is completed. When receiving data, however, a received byte must be read from the SPI Data Register before the next byte has been completely shifted in. Otherwise, the first byte 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 21. SPI Pin Overrides

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

Note: See “Alternate functions of Port B” on page 80 for a detailed description of how to define the direction of the user defined SPI pins.

$\overline{SS}$ 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 the $\overline{SS}$ pin is driven low by peripheral circuitry when the SPI is configured as master with the $\overline{SS}$ pin defined as an input, 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 and the I-bit in SREG is set, the interrupt routine will be executed.

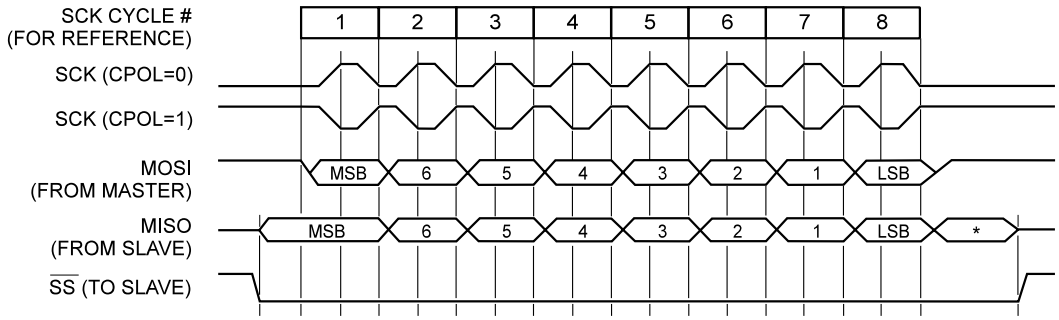
Thus, when interrupt-driven SPI transmission is used in master mode, and there exists a possibility that $\overline{SS}$ is driven low, the interrupt should always check that the MSTR bit is still set. Once the MSTR bit has been cleared by a slave select, it must be set by the user to re-enable SPI master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

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. Note that the SPI logic will be reset once the $\overline{SS}$ pin is brought high. If the $\overline{SS}$ pin is brought high during a transmission, the SPI will stop sending and receiving immediately and both data received and data sent must be considered as lost.

Data Modes

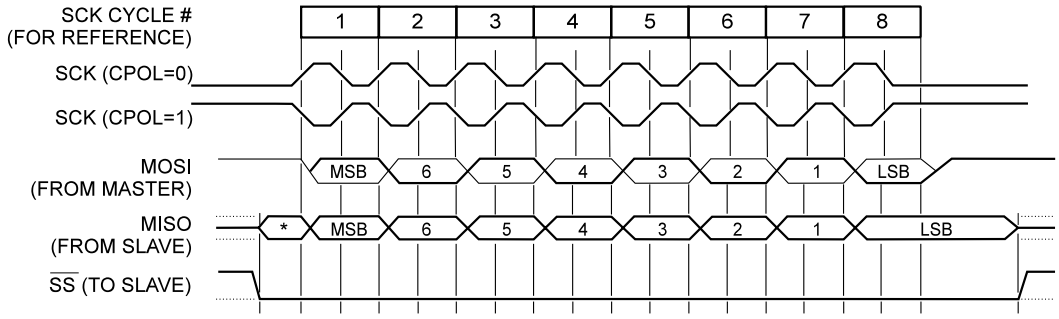
There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 43 and Figure 44.

Figure 43. SPI Transfer Format with CPHA = 0 and DORD = 0



* Not defined but normally MSB of character just received

Figure 44. SPI Transfer Format with CPHA = 1 and DORD = 0



* Not defined but normally LSB of previously transmitted character

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 the SPI interrupt to be executed if SPIF bit in the SPSR register is set and the if the global interrupt enable bit in SREG is set.

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

When the SPE bit is set (one), the SPI is enabled. This bit must be set to enable any SPI operations.

- **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 43 and Figure 44 for additional information.

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

Refer to Figure 43 or Figure 44 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 Table 22:

Table 22. Relationship Between SCK and the Oscillator Frequency

SPI2X	SPR1	SPR0	SCK Frequency
0	0	0	$f_{cl}/4$
0	0	1	$f_{cl}/16$
0	1	0	$f_{cl}/64$
0	1	1	$f_{cl}/128$
1	0	0	$f_{cl}/2$
1	0	1	$f_{cl}/8$
1	1	0	$f_{cl}/32$
1	1	1	$f_{cl}/64$

Note: When the SPI is configured as Slave, the SPI is only guaranteed to work at $f_{cl}/4$.

SPI Status Register – SPSR

Bit	7	6	5	4	3	2	1	0	
$0E ($2E)	SPIF	WCOL	-	-	-	-	-	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
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. If $\overline{SS}$ is an input and is driven low when the SPI is in master mode, this will also set the SPIF flag. 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. 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..1 - Res: Reserved bits**

These bits are reserved bits in the ATmega161 and will always read as zero.

- **Bit 0 - SPI2X: Double SPI speed bit**

When this bit is set (one) the SPI speed (SCK Frequency) will be doubled when the SPI is in master mode (see Table 22). This means that the maximum SCK period will be 2 CPU clock periods. When the SPI is configured as Slave, the SPI is only guaranteed to work at fcl/4.

The SPI interface on the ATmega161 is also used for program memory and EEPROM downloading or uploading. See page 109 for serial programming and verification.

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	x	x	x	x	x	x	x	x	Undefined

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.

UARTs

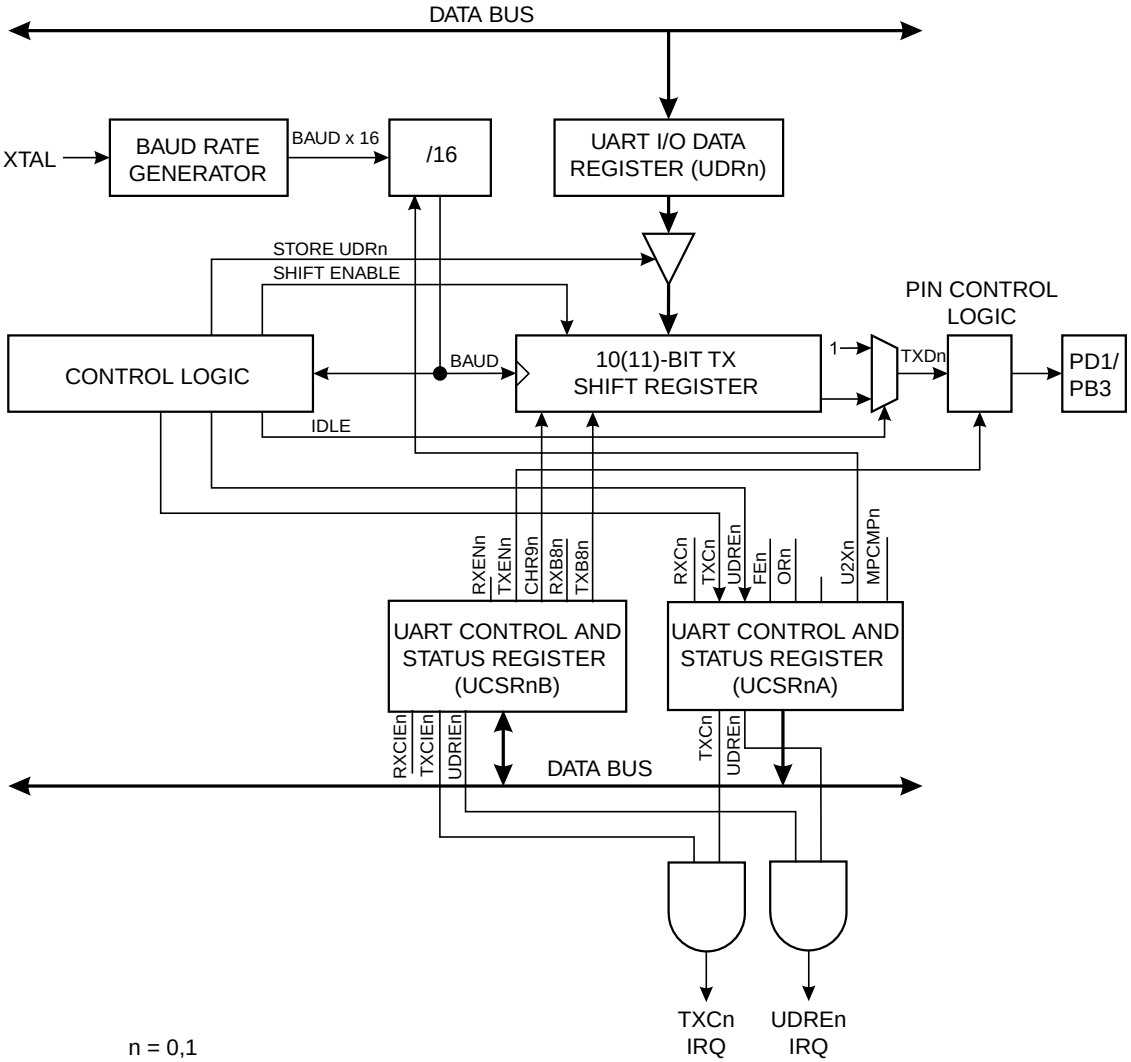
The ATmega161 features two full duplex (separate receive and transmit registers) Universal Asynchronous Receiver and Transmitter (UART). The main features are:

- 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
- Multi-processor Communication Mode
- Double Speed UART Mode

Data Transmission

A block schematic of the UART transmitter is shown in Figure 45. The two UARTs are identical and the functionality is described in general for the two UARTs.

Figure 45. UART Transmitter



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

- A new character has been written to UDRn 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 UDRn 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 UDRn to the shift register. At this time the UDREn (UART Data Register Empty) bit in the UART Control and Status Register, UCSRnA, is set. When this bit is set (one), the UART is ready to receive the next character. At the same time as the data is transferred from UDRn 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 CHR9n bit in the UART Control and Status Register, UCSRnB is set), the TXB8 bit in UCSRnB is transferred to bit 9 in the Transmit shift register.

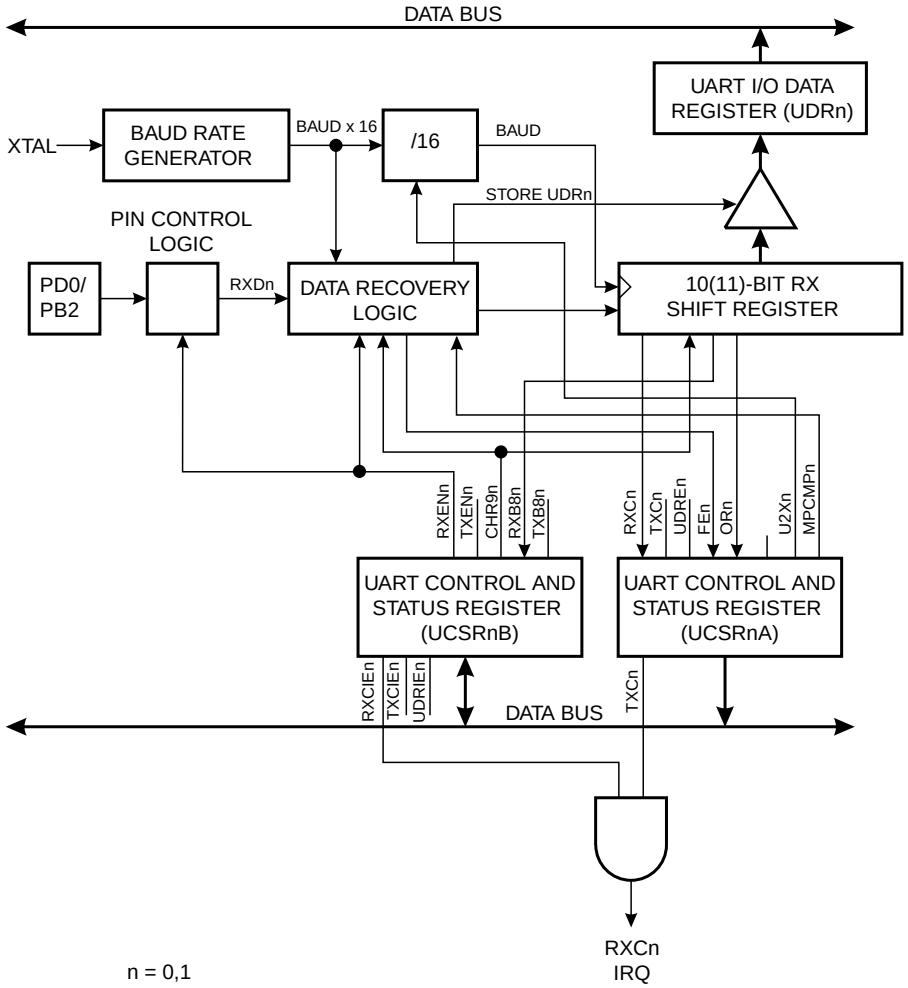
On the Baud Rate clock following the transfer operation to the shift register, the start bit is shifted out on the TXDn pin. Then follows the data, LSB first. When the stop bit has been shifted out, the shift register is loaded if any new data has been written to the UDRn during the transmission. During loading, UDREn is set. If there is no new data in the UDRn register to send when the stop bit is shifted out, the UDREn flag will remain set until UDRn is written again. When no new data has been written, and the stop bit has been present on TXDn for one bit length, the TX Complete flag, TXCn, in UCSRnA is set.

The TXENn bit in UCSRnB enables the UART transmitter when set (one). When this bit is cleared (zero), the PD1 (UART0) or PB3 (UART1) pin can be used for general I/O. When TXENn is set, the UART Transmitter will be connected to PD1 (UART0) or PB3 (UART1), which is forced to be an output pin regardless of the setting of the DDD1 bit in DDRD (UART0) or DDB3 in DDRB (UART1). Note that PB3 (UART1) also is used as one of the input pins to the Analog Comparator. It is therefore not recommended to use UART1 if the Analog Comparator also is used in the application at the same time.

Data Reception

Figure 46 shows a block diagram of the UART Receiver

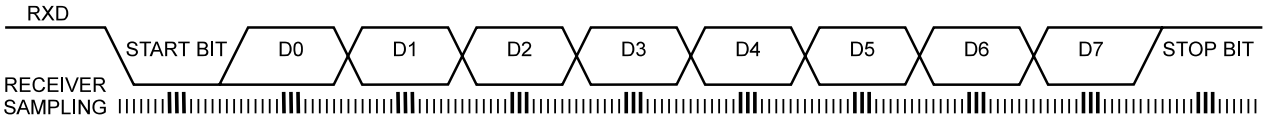
Figure 46. UART Receiver



The receiver front-end logic samples the signal on the RXDn 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 RXDn pin at samples 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 47. Note that the description above is not valid when the UART transmission speed is doubled. See “Double Speed Transmission” on page 68 for a detailed description.

Figure 47. Sampling Received Data



Note: This figure is not valid when the UART speed is doubled. See"Double Speed Transmission" on page 68 for a detailed description.

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 (FEn) flag in the UART Control and Status Register (UCSRnA) is set. Before reading the UDRn register, the user should always check the FEn bit to detect Framing Errors.

Whether or not a valid stop bit is detected at the end of a character reception cycle, the data is transferred to UDRn and the RXCn flag in UCSRnA is set. UDRn is in fact two physically separate registers, one for transmitted data and one for received data. When UDRn is read, the Receive Data register is accessed, and when UDRn is written, the Transmit Data register is accessed. If 9 bit data word is selected (the CHR9n bit in the UART Control and Status Register, UCSRnB is set), the RXB8n bit in UCSRnB is loaded with bit 9 in the Transmit shift register when data is transferred to UDRn.

If, after having received a character, the UDRn register has not been read since the last receive, the OverRun (ORn) flag in UCSRnB is set. This means that the last data byte shifted into to the shift register could not be transferred to UDRn and has been lost. The ORn bit is buffered, and is updated when the valid data byte in UDRn is read. Thus, the user should always check the ORn bit after reading the UDRn register in order to detect any overruns if the baud rate is high or CPU load is high.

When the RXEN bit in the UCSRnB register is cleared (zero), the receiver is disabled. This means that the PD0 pin can be used as a general I/O pin. When RXEN is set, the UART Receiver will be connected to PD0 (UART0) or PB2 (UART1), which is forced to be an input pin regardless of the setting of the DDD0 in DDRD (UART0) or DDB2 bit in DDRB (UART1). When PD0 (UART0) or PB2 (UART1) is forced to input by the UART, the PORTD0 (UART0) or PORTB2 (UART1) bit can still be used to control the pull-up resistor on the pin.

Note that PB2 (UART1) also is used as one of the input pins to the Analog Comparator. It is therefor not recommended to use UART1 if the Analog Comparator also is used in the application at the same time.

When the CHR9n bit in the UCSRnB register is set, transmitted and received characters are 9-bit long plus start and stop bits. The 9th data bit to be transmitted is the TXB8n bit in UCSRnB register. This bit must be set to the wanted value before a transmission is initiated by writing to the UDRn register. The 9th data bit received is the RXB8n bit in the UCSRnB register.

Multi-processor Communication Mode

The Multi-processor Communication Mode enables several slave MCUs to receive data from a master MCU. This is done by first decoding an address byte to find out which MCU has been addressed. If a particular slave MCU has been addressed, it will receive the following data bytes as normal, while the other slave MCUs will ignore the data bytes until another address byte is received.

For an MCU to act as a master MCU, it should enter 9-bit transmission mode (CHR9n in UCSRnB set). The 9th bit must be one to indicate that an address byte is being transmitted, and zero to indicate that a data byte is being transmitted.

For the slave MCUs, the mechanism appears slightly differently for 8-bit and 9-bit reception mode. In 8-bit reception mode (CHR9n in UCSRnB cleared), the stop bit is one for an address byte and zero for a data byte. In 9-bit reception mode (CHR9n in UCSRnB set), the 9th bit is one for an address byte and zero for a data byte, whereas the stop bit is always high.

The following procedure should be used to exchange data in Multi-processor Communication Mode:

1. All slave MCUs are in Multi-processor Communication Mode (MPCMn in UCSRnA is set).
2. The master MCU sends an address byte, and all slaves receive and read this byte. In the slave MCUs, the RXCn flag in UCSRnA will be set as normal.
3. Each slave MCU reads the UDRn register and determines if it has been selected. If so, it clears the MPCMn bit in UCSRnA, otherwise it waits for the next address byte.

4. For each received data byte, the receiving MCU will set the receive complete flag (RXCn in UCSRnA. In 8-bit mode, the receiving MCU will also generate a framing error (FEn in UCSRnA set), since the stop bit is zero. The other slave MCUs, which still have the MPCMn bit set, will ignore the data byte. In this case, the UDRn register and the RXCn, FEn, or flags will not be affected.
5. After the last byte has been transferred, the process repeats from step 2.

UART Control

UART0 I/O Data Register – UDR0

Bit	7	6	5	4	3	2	1	0	
$0C ($2C)	MSB							LSB	UDR0
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	

UART1 I/O Data Register – UDR1

Bit	7	6	5	4	3	2	1	0	
$03 ($23)	MSB							LSB	UDR1
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 UDRn 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 UDRn, the UART Receive Data register is read.

UART0 Control and Status Registers – UCSR0A

Bit	7	6	5	4	3	2	1	0	
$0B ($2B)	RXC0	TXC0	UDRE0	FE0	OR0	-	U2X0	MPCM0	UCSR0A
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial value	0	0	1	0	0	0	0	0	

UART1 Control and Status Registers – UCSR1A

Bit	7	6	5	4	3	2	1	0	
$02 ($22)	RXC1	TXC1	UDRE1	FE1	OR1	-	U2X1	MPCM1	UCSR1A
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial value	0	0	1	0	0	0	0	0	

- **Bit 7 - RXC0/RXC1: UART Receive Complete**

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

- **Bit 6 - TXC0/TXC1: 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 UDRn. 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 TXCIEn bit in UCSRnB is set, setting of TXCn causes the UART Transmit Complete interrupt to be executed. TXCn is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, the TXCn bit is cleared (zero) by writing a logical one to the bit.

- **Bit 5 - UDRE0/UDRE1: UART Data Register Empty**

This bit is set (one) when a character written to UDRn 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 UDRIEn bit in UCSRnB is set, the UART Transmit Complete interrupt will be executed as long as UDREn is set and the global interrupt enable bit in SREG is set. UDREn is cleared by writing UDRn. When interrupt-driven data transmittal is used, the UART Data Register Empty Interrupt routine must write UDRn in order to clear UDREn, otherwise a new interrupt will occur once the interrupt routine terminates.

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

- **Bit 4 - FE0/FE1: 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 FEn bit is cleared when the stop bit of received data is one.

- **Bit 3 - OR0/OR1: OverRun**

This bit is set if an Overrun condition is detected, i.e. when a character already present in the UDRn register is not read before the next character has been shifted into the Receiver Shift register. The ORn bit is buffered, which means that it will be set once the valid data still in UDRn is read.

The ORn bit is cleared (zero) when data is received and transferred to UDRn.

- **Bit 2 - Res: Reserved bit**

This bit is reserved bit in the ATmega161 and will always read as zero.

- **Bits 1 - U2X0/U2X1: Double the UART transmission speed**

When this bit is set (one) the UART speed will be doubled. This means that a bit will be transmitted/received in 8 CPU clock periods instead of 16 CPU clock periods. For a detailed description, see “Double Speed Transmission” on page 68”.

- **Bit 0 - MPCM0/MPCM1: Multi-processor Communication Mode**

This bit is used to enter Multi-processor Communication Mode. The bit is set when the slave MCU waits for an address byte to be received. When the MCU has been addressed, the MCU switches off the MPCMn bit, and starts data reception.

For a detailed description, see “Multi-processor Communication Mode”.

UART0 Control and Status Registers – UCSR0B

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	**RXCIE0**	**TXCIE0**	**UDRIE0**	**RXEN0**	**TXEN0**	**CHR90**	**RXB80**	**TXB80**	**UCSR0B**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial value	0	0	0	0	0	0	1	0	

UART1 Control and Status Registers – UCSR1B

Bit	7	6	5	4	3	2	1	0	
$01 ($21)	**RXCIE1**	**TXCIE1**	**UDRIE1**	**RXEN1**	**TXEN1**	**CHR91**	**RXB81**	**TXB81**	**UCSR1B**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial value	0	0	0	0	0	0	1	0	

- **Bit 7 - RXCIE0/RXCIE1: RX Complete Interrupt Enable**

When this bit is set (one), a setting of the RXCn bit in UCSRnA will cause the Receive Complete interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 6 - TXCIE0/TXCIE1: TX Complete Interrupt Enable**

When this bit is set (one), a setting of the TXCn bit in UCSRnA will cause the Transmit Complete interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 5 - UDRIE0/UDREI1: UART Data Register Empty Interrupt Enable**

When this bit is set (one), a setting of the UDREn bit in UCSRnA will cause the UART Data Register Empty interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 4 - RXEN0/RXEN1: Receiver Enable**

This bit enables the UART receiver when set (one). When the receiver is disabled, the TXCn, ORn and FEn status flags cannot become set. If these flags are set, turning off RXEN does not cause them to be cleared.

- **Bit 3 - TXEN0/TXEN1: 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 UDRn has been completely transmitted.

- **Bit 2 - CHR90/CHR91: 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 RXB8n and TXB8 bits in UCSRnB, respectively. The 9th data bit can be used as an extra stop bit or a parity bit.

- **Bit 1 - RXB80/RXB81: Receive Data Bit 8**

When CHR9n is set (one), RXB8n is the 9th data bit of the received character.

- **Bit 0 - TXB80/TXB81: Transmit Data Bit 8**

When CHR9n is set (one), TXB8n is the 9th data bit in the character to be transmitted.

Baud Rate Generator

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

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

- BAUD = Baud-rate
- f_{CK}= Crystal Clock frequency
- UBR = Contents of the UBRRH and UBRR registers, (0-4095)
- Note that this equation is not valid when the UART transmission speed is doubled. See “Double Speed Transmission” on page 68 for a detailed description.

For standard crystal frequencies, the most commonly used baud rates can be generated by using the UBR settings in Table 23. UBR values which yield an actual baud rate differing less than 2% from the target baud rate, are bold in the table. However, using baud rates that have more than 1% error is not recommended. High error ratings give less noise resistance.

Table 23. UBR Settings at Various Crystal Frequencies

Baud Rate	1 MHz	% Error	1.8432 MHz	% Error	2 MHz	% Error
2400	UBR= **25**	**0.2**	UBR= **47**	**0.0**	UBR= **51**	**0.2**
4800	UBR= **12**	**0.2**	UBR= **23**	**0.0**	UBR= **25**	**0.2**
9600	UBR= 6	7.5	UBR= **11**	**0.0**	UBR= **12**	**0.2**
14400	UBR= 3	7.8	UBR= **7**	**0.0**	UBR= 8	3.7
19200	UBR= 2	7.8	UBR= **5**	**0.0**	UBR= 6	7.5
28800	UBR= 1	7.8	UBR= **3**	**0.0**	UBR= 3	7.8
38400	UBR= 1	22.9	UBR= **2**	**0.0**	UBR= 2	7.8
57600	UBR= 0	7.8	UBR= **1**	**0.0**	UBR= 1	7.8
76800	UBR= 0	22.9	UBR= 1	33.3	UBR= 1	22.9
115200	UBR= 0	84.3	UBR= **0**	**0.0**	UBR= 0	7.8

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

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

UART0 and UART1 High byte Baud Rate Register UBRRHI

Bit	7	6	5	4	3	2	1	0	
$20 ($40)	**MSB1**			**LSB1**	**MSB0**			**LSB0**	**UBRRHI**
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 UART baud register is a 12-bit register. The 4 most significant bits are located in a separate register, UBRRHI. Note that both UART0 and UART1 share this register. Bit 7 to bit 4 of UBRRHI contain the 4 most significant bits of the UART1 baud register. Bit3 to Bit0 contain the 4 most significant bits of the UART0 baud register.

UART0 Baud Rate Register Low byte – UBRR0

Bit	7	6	5	4	3	2	1	0	
$09 ($29)	MSB							LSB	UBRR0
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	

UART1 Baud Rate Register Low byte – UBRR1

Bit	7	6	5	4	3	2	1	0	
$00 ($20)	MSB							LSB	UBRR1
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	

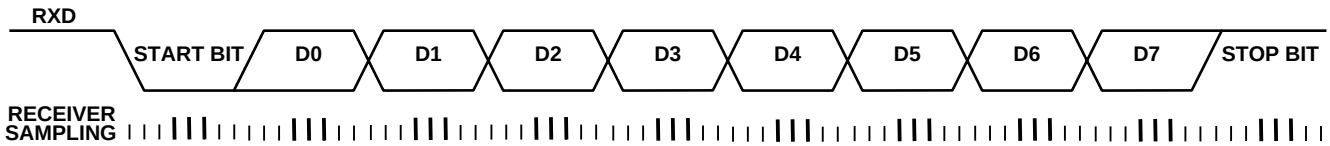
UBRRn stores the 8 least significant bits of the UART baud rate register.

Double Speed Transmission

The ATmega161 provides a separate UART mode that allows the user to double the communication speed. By setting the U2X bit in UART Control and Status Register UCSRnA, the UART speed will be doubled. The data reception will differ slightly from normal mode. Since the speed is doubled, the receiver front-end logic samples the signals on RXDn pin at a frequency 8 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 RXDn pin at samples 4, 5 and 6. 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 4, 5 and 6. 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 48.

Figure 48. Sampling Received Data when the transmission speed is doubled



The Baud Rate Generator in double UART speed mode

Note that the baud-rate equation is different from the equation at page 66 when the UART speed is doubled:

$$BAUD = \frac{f_{CK}}{8(UBR + 1)}$$

- BAUD = Baud-rate
- f_{CK}= Crystal Clock frequency
- UBR = Contents of the UBRRHI and UBRR registers, (0-4095)
- Note that this equation is only valid when the UART transmission speed is doubled.

For standard crystal frequencies, the most commonly used baud rates can be generated by using the UBR settings in Table 23. UBR values which yield an actual baud rate differing less than 1.5% from the target baud rate, are bold in the table. However since the number of samples are reduced and the system clock might have some variance (this applies especially when using resonators), it is recommended that the baud rate error is less than 0.5%.

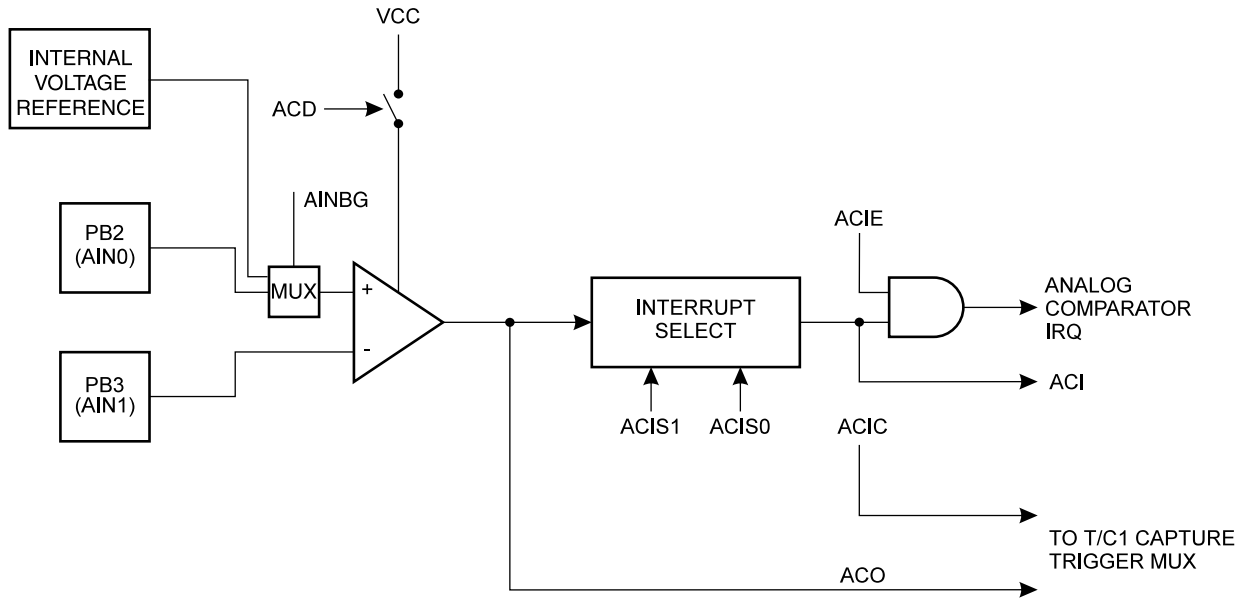
Table 24. UBR Settings at Various Crystal Frequencies in Double Speed Mode

Baud Rate	1.0000 MHz	% Error	1.8432 MHz	% Error	2.0000 MHz	% Error
2400	UBR = **51**	**0.2**	UBR = **95**	**0.0**	UBR = **103**	**0.2**
4800	UBR = **25**	**0.2**	UBR = **47**	**0.0**	UBR = **51**	**0.2**
9600	UBR = **12**	**0.2**	UBR = **23**	**0.0**	UBR = **25**	**0.2**
14400	UBR = 8	3.7	UBR = **15**	**0.0**	UBR = 16	2.1
19200	UBR = 6	7.5	UBR = **11**	**0.0**	UBR = **12**	**0.2**
28800	UBR = 3	7.8	UBR = **7**	**0.0**	UBR = 8	3.7
38400	UBR = 2	7.8	UBR = **5**	**0.0**	UBR = 6	7.5
57600	UBR = 1	7.8	UBR = **3**	**0.0**	UBR = 3	7.8
76800	UBR = 1	22.9	UBR = **2**	**0.0**	UBR = 2	7.8
115200	UBR = 0	84.3	UBR = **1**	**0.0**	UBR = 1	7.8
230400	-	-	UBR = **0**	**0.0**	UBR = 0	84.3
Baud Rate	3.2768 MHz	% Error	3.6864 MHz	% Error	4.0000 MHz	% Error
2400	UBR = **170**	**0.2**	UBR = **191**	**0.0**	UBR = **207**	**0.2**
4800	UBR = **84**	**0.4**	UBR = **95**	**0.0**	UBR = **103**	**0.2**
9600	UBR = **42**	**0.8**	UBR = **47**	**0.0**	UBR = **51**	**0.2**
14400	UBR = 27	1.6	UBR = **31**	**0.0**	UBR = **34**	**0.8**
19200	UBR = 20	1.6	UBR = **23**	**0.0**	UBR = **25**	**0.2**
28800	UBR = 13	1.6	UBR = **15**	**0.0**	UBR = 16	2.1
38400	UBR = 10	3.1	UBR = **11**	**0.0**	UBR = **12**	**0.2**
57600	UBR = 6	1.6	UBR = **7**	**0.0**	UBR = 8	3.7
76800	UBR = 4	6.2	UBR = **5**	**0.0**	UBR = 6	7.5
115200	UBR = 3	12.5	UBR = **3**	**0.0**	UBR = 3	7.8
230400	UBR = 1	12.5	UBR = **1**	**0.0**	UBR = 1	7.8
460800	UBR = 0	12.5	UBR = **0**	**0.0**	UBR = 0	7.8
912600	-	-	-	-	UBR = 0	84.3
Baud Rate	7.3728 MHz	% Error	8.0000 MHz	% Error	9.2160 MHz	% Error
2400	UBR = **383**	**0.0**	UBR = **416**	**0.1**	UBR = **479**	**0.0**
4800	UBR = **191**	**0.0**	UBR = **207**	**0.2**	UBR = **239**	**0.0**
9600	UBR = **95**	**0.0**	UBR = **103**	**0.2**	UBR = **119**	**0.0**
14400	UBR = **63**	**0.0**	UBR = **68**	**0.6**	UBR = **79**	**0.0**
19200	UBR = **47**	**0.0**	UBR = **51**	**0.2**	UBR = **59**	**0.0**
28800	UBR = **31**	**0.0**	UBR = **34**	**0.8**	UBR = **39**	**0.0**
38400	UBR = **23**	**0.0**	UBR = **25**	**0.2**	UBR = **29**	**0.0**
57600	UBR = **15**	**0.0**	UBR = 16	2.1	UBR = **19**	**0.0**
76800	UBR = **11**	**0.0**	UBR = 12	**0.2**	UBR = **14**	**0.0**
115200	UBR = **7**	**0.0**	UBR = 8	3.7	UBR = **9**	**0.0**
230400	UBR = **3**	**0.0**	UBR = 3	7.8	UBR = **4**	**0.0**
460800	UBR = **1**	**0.0**	UBR = 1	7.8	UBR = 2	20.0
912600	UBR = **0**	**0.0**	UBR = 0	7.8	UBR = 0	20.0

Analog Comparator

The analog comparator compares the input values on the positive input PB2 (AIN0) and negative input PB3 (AIN1). When the voltage on the positive input PB2 (AIN0) is higher than the voltage on the negative input PB3 (AIN1), 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 49.

Figure 49. Analog Comparator Block Diagram



Analog Comparator Control And Status Register – ACSR

Bit	7	6	5	4	3	2	1	0	
$08 ($28)	**ACD**	**AINBG**	**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 - AINBG: Analog Comparator Bandgap Select**

When this bit is set, a fixed bandgap voltage of 1.22 ± 0.05V replaces the normal input to the positive input (AIN0) of the comparator. When this bit is cleared, the normal input pin PB2 is applied to the positive input of the comparator.

- **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.

- **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 enabled. 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 25.

Table 25. 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.

Caution: Using the SBI or CBI instruction on other bits than ACI in this register, will write a one back into ACI if it is read as set, thus clearing the flag.

The Analog Comparator pins (PB2 and PB3) are also used as the TXD1 and RXD1 pins for UART1. Note that if the UART1 transceiver or receiver is enabled, the UART1 will override the settings in the DDRB register even if the Analog Comparator is enabled. Therefore it is not recommended to use UART1 if the Analog Comparator is needed in the same application at the same time. See “UARTs” on page 60 for more details.

Internal Voltage reference

ATmega161 features an internal voltage reference with a nominal voltage of 1.22V. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator.

Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence on the way it should be used. The maximum start-up time is TBD. To save power, the reference is on during the following situations only:

1. When BOD is enabled (by programming the BODEN fuse).
2. When the bandgap reference is connected to the Analog Comparator (by setting the AINBG bit in ACSR).

Thus, when BOD is not enabled, after setting the AINBG bit, the user must always allow the reference to start up before the output from the Analog Comparator is used. The bandgap reference uses approx. 10 µA, and to reduce the power consumption in power-down mode, the user can turn off the reference when entering this mode.

Interface to external memory

With all the features the external memory interface provides, it is well suited to operate as an interface to memory devices such as external SRAM and FLASH, and peripherals as LCD-display, A/D, D/A etc. The control bits for the external memory interface are located in two registers, the MCU Control Register – MCUCR and the Extended MCU Control Register – EMCUCR.

MCU Control Register – MCUCR

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	SRE	SRW10	SE	SM1	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Extended MCU Control Register – EMCUCR

Bit	7	6	5	4	3	2	1	0	
$36 ($56)	SM0	SRL2	SRL1	SRL0	SRW01	SRW00	SRW11	ISC20	EMCUCR
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 MCUCR – SRE: External SRAM enable**

When the SRE bit is set (one), the external memory interface is enabled, and the pin functions AD0-7 (Port A), A8-15 (Port C), ALE (Port E), $\overline{WR}$ and $\overline{RD}$ (Port D) are activated as the alternate pin functions. The SRE bit overrides any pin direction settings in the respective data direction registers. See Figure 51 – Figure 54 for description of the external memory pin functions. When the SRE bit is cleared (zero), the external data memory interface is disabled, and the normal pin and data direction settings are used

- **Bit 6..4 EMCUCR – SRL2, SRL1, SRL0: Wait state page limit**

It is possible to configure different wait-states for different external memory addresses. The external memory address space can be divided in two pages with different wait-state bits. The SRL2, SRL1 and SRL0 bits select the split of the pages, see Table 27 and Figure 50. As default the SRL2, SRL1 and SRL0 bits are set to zero and the entire external memory address space is treated as one page. When the entire SRAM address space is configured as one page, the wait-states are configured by the SRW11 and SRW10 bits.

- **Bit 1 EMCUCR and Bit 6 MCUCR – SRW11, SRW10: Wait state select bits for upper page**

The SRW11 and SRW10 bits control the number of wait-states for the upper page of the external memory address space, see Table 26. Note that if the SRL2, SRL1 and SRL0 bits are set to zero, the SRW11 and SRW10 bit settings will define the wait-state of the entire SRAM address space.

- **Bit 3..2 EMCUCR – SRW01, SRW00: Wait state select bits for lower page**

The SRW01 and SRW00 bits control the number of wait-states for the lower page of the external memory address space, see Table 26.

Table 26. Wait-states

SRWn1	SRWn0	Wait-states
0	0	No wait states
0	1	Wait one cycle during read/write strobe
1	0	Wait two cycles during read/write strobe
1	1	Wait two cycles during read/write and wait one cycle before driving out new address

Note: n = 0 or 1 (lower/upper page).

For further details of the timing and wait-states of the external memory interface, see Figure 51 – Figure 54 how the setting of the SRW bits affects the timing.

Table 27. Page limits with different settings of SRL2..0

SRL2	SRL1	SRL0	Page Limits
0	0	0	Lower page = N/A Upper page = $0460-$FFFF
0	0	1	Lower page = $0460-$1FFF Upper page = $2000-$FFFF
0	1	0	Lower page = $0460-$4FFF Upper page = $4000-$FFFF
0	1	1	Lower page = $0460-$5FFF Upper page = $6000-$FFFF
1	0	0	Lower page = $0460-$7FFF Upper page = $8000-$FFFF
1	0	1	Lower page = $0460-$9FFF Upper page = $A000-$FFFF
1	1	0	Lower page = $0460-$BFFF Upper page = $C000-$FFFF
1	1	1	Lower page = $0460-$DFFF Upper page = $E000-$FFFF

Figure 50. External memory with page select

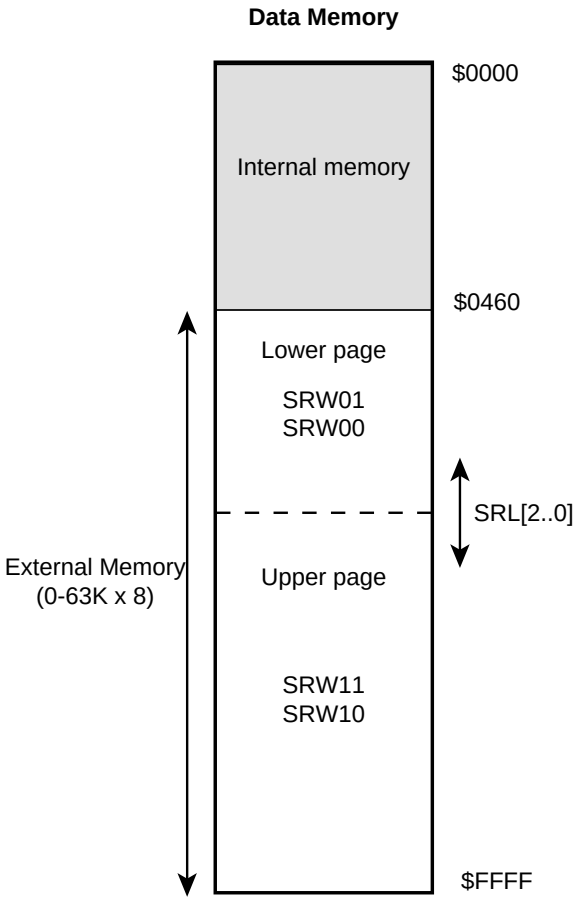
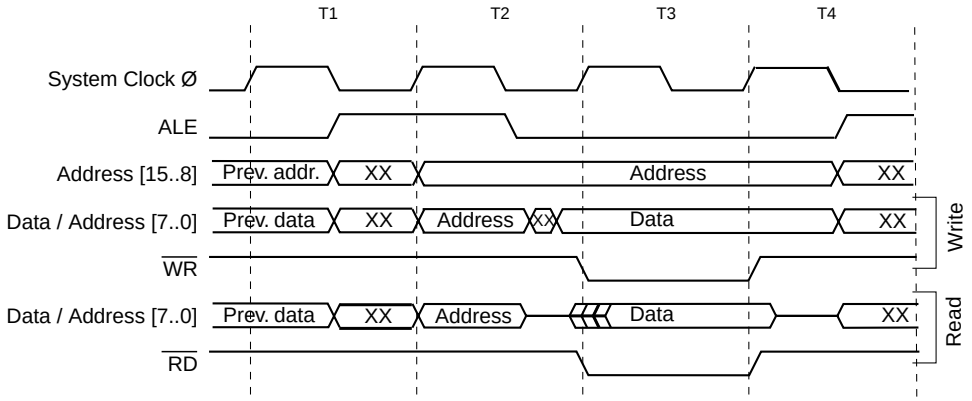
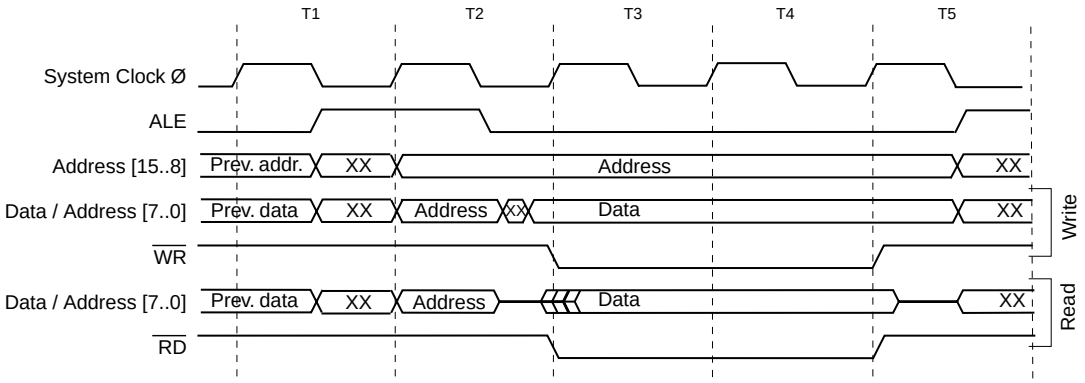


Figure 51. External Data Memory Cycles without Wait State (SRWn1 = 0 and SRWn0 =0)



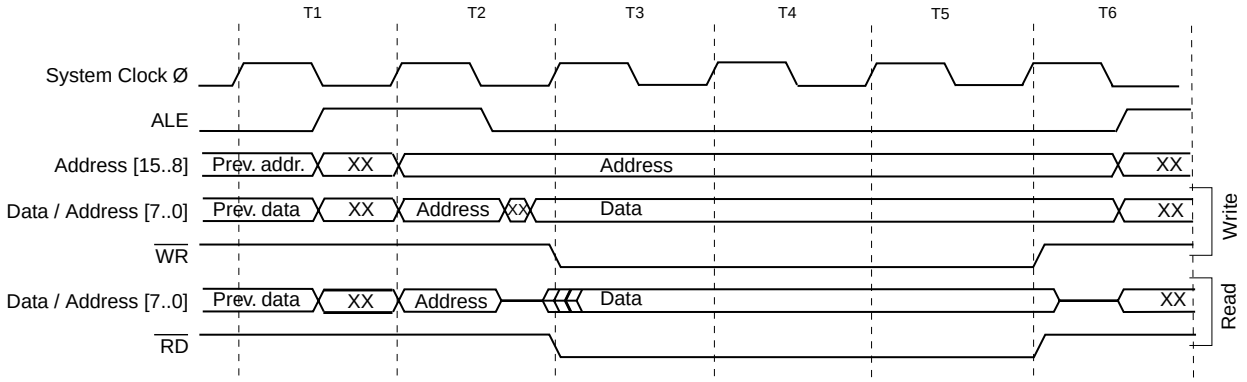
Note: SRWn1 = SRW11 (upper page) or SRW01 (lower page), SRWn0 = SRW10 (upper page) or SRW00 (lower page)
The ALE pulse in period T4 is only present if the next instruction accesses the RAM (internal or external). The Data and Address will only change in T4 if ALE is present (the next instruction accesses the RAM).

Figure 52. External Data Memory Cycles with SRWn1 = 0 and SRWn0 = 1



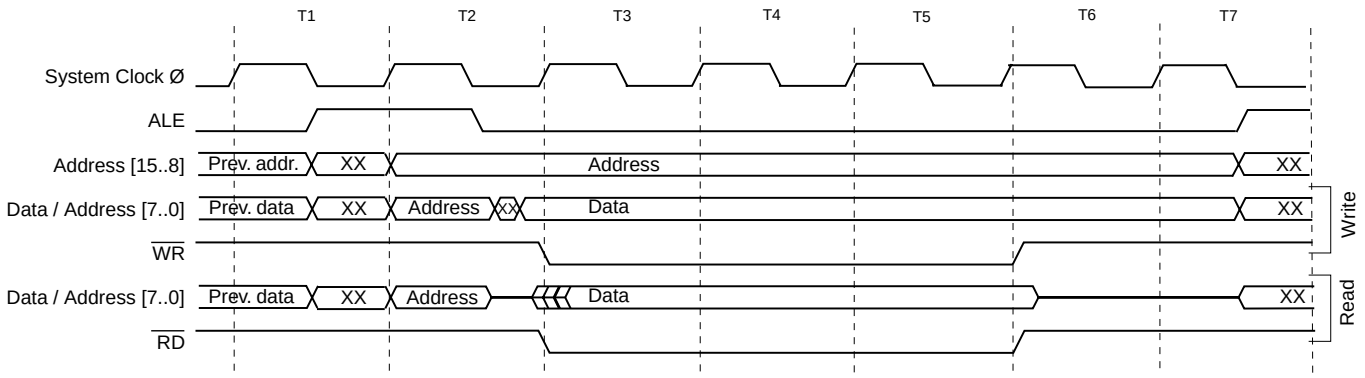
Note: SRWn1 = SRW11 (upper page) or SRW01 (lower page), SRWn0 = SRW10 (upper page) or SRW00 (lower page)
The ALE pulse in period T5 is only present if the next instruction accesses the RAM (internal or external). The Data and Address will only change in T5 if ALE is present (the next instruction accesses the RAM).

Figure 53. External Data Memory Cycles with SRWn1 = 1 and SRWn0 = 0



Note: SRWn1 = SRW11 (upper page) or SRW01 (lower page), SRWn0 = SRW10 (upper page) or SRW00 (lower page)
The ALE pulse in period T6 is only present if the next instruction accesses the RAM (internal or external). The Data and Address will only change in T6 if ALE is present (the next instruction accesses the RAM).

Figure 54. External Data Memory Cycles with SRWn1 = 1 and SRWn0 = 1



Note: SRWn1 = SRW11 (upper page) or SRW01 (lower page), SRWn0 = SRW10 (upper page) or SRW00 (lower page)
The ALE pulse in period T7 is only present if the next instruction accesses the RAM (internal or external). The Data and Address will only change in T7 if ALE is present (the next instruction accesses the RAM).

Using the External Memory Interface

The interface consists of:

- Port A: Multiplexed low-order address bus and data bus
- Port C: High-order address bus
- The ALE-pin: Address latch enable
- The $\overline{RD}$ and $\overline{WR}$-pin: Read and write strobes.

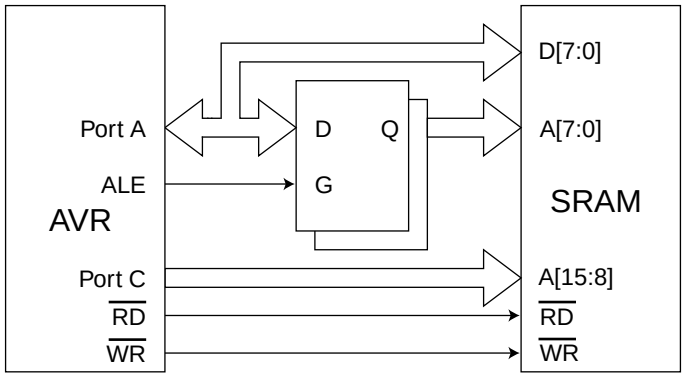
The external memory interface is enabled by setting the SRE – External SRAM enable bit of the MCUCR – MCU control register, and will over-ride the setting of the data direction register DDRA, DDRD and DDRE. When the SRE bit is cleared (zero), the external memory interface is disabled, and the normal pin and data direction settings are used. When SRE is low, the address space above the internal SRAM boundary is not mapped into the internal SRAM, as in AVR parts not having external memory interface.

When ALE goes from high to low, there is a valid address on Port A. ALE is low during a data transfer. $\overline{RD}$ and $\overline{WR}$ are active when accessing the external memory only.

When the external memory interface is enabled, the ALE signal may have short pulses when accessing the internal RAM, but the ALE signal is stable when accessing the external memory.

Figure 55 sketches how to connect an external SRAM to the AVR using 8 latches which are transparent when G is high.

Figure 55. External SRAM connected to the AVR



For details in the timing for the SRAM interface, please refer to Figure 84 – Figure 87 and Table 49 – Table 56.

I/O-Ports

All AVR ports have true Read-modify-write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies for changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input).

Port A

Port A is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for the 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 20 mA 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 memory interface. Port A can be configured to be the multiplexed low-order address/data bus during accesses to the external data memory. In this mode, Port A has internal pull-up resistors.

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.

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	

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	

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 Port A Data Latch is read, and when reading PINA, the logical values present on the pins are read.

Port A as General Digital I/O

All 8 pins in Port A have equal functionality 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, the PORTAn has to be cleared (zero) or the pin has to be configured as an output pin. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Table 28. DDAn Effects on Port A Pins

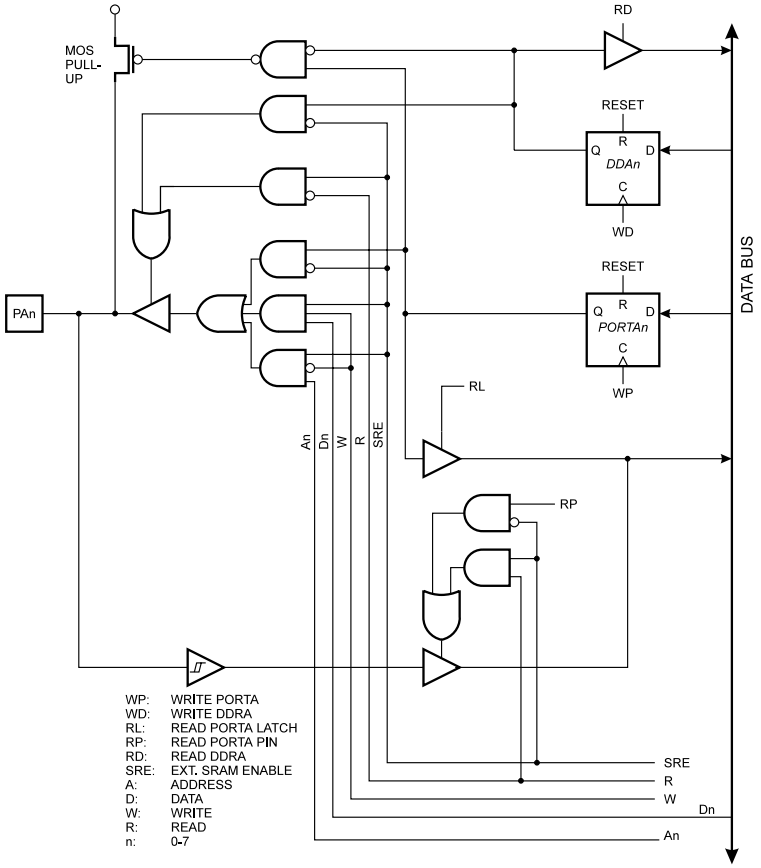
DDAn	PORTAn	I/O	Pull-up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PAn will source current if ext. pulled low.
1	0	Output	No	Push-pull Zero Output
1	1	Output	No	Push-pull One Output

n: 7,6…0, pin number.

Port A Schematics

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

Figure 56. Port A Schematic Diagrams (Pins PA0 - PA7)



Port B

Port B is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for the 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 20 mA 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	OC0 (Timer/Counter 0 Compare match Output) /T0 (Timer/Counter 0 external counter input)
PB1	OC2 (Timer/Counter 2 Compare match Output) /T1 (Timer/Counter 1 external counter input)
PB2	RXD1 (UART1 input line) /AIN0 (Analog comparator positive input)
PB3	TXD1 (UART1 output line) /AIN1 (Analog comparator negative input)
PB4	$\overline{SS}$ (SPI Slave Select input)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB7	SCK (SPI Bus Serial Clock)

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

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	

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	

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 Port B Data Latch is read, and when reading PINB, the logical values present on the pins are read.

Port B as General Digital I/O

All 8 pins in Port B have equal functionality 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. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running

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 Port B

The alternate pin configuration is as follows:

- **SCK - Port B, Bit 7**

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 DDB7. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB7. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB7 bit. See the description of the SPI port for further details.

- **MISO - Port B, Bit 6**

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 DDB6. When the SPI is enabled as a slave, the data direction of this pin is controlled by DDB6. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB6 bit. See the description of the SPI port for further details.

- **MOSI - Port B, Bit 5**

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 DDB5. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB5. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB5 bit. See the description of the SPI port for further details.

- **SS - Port B, Bit 4**

$\overline{SS}$: Slave port select input. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB5. 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 DDB5. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB5 bit. See the description of the SPI port for further details.

- **TXD1/AIN1 - Port B, Bit 3**

AIN1, Analog Comparator Negative Input. This pin also serves as the negative input of the on-chip analog comparator.

TXD1, Transmit Data (Data output pin for the UART1). When the UART1 transmitter is enabled, this pin is configured as an output regardless of the value of DDRB3.

- **RXD1/AIN0 - Port B, Bit 2**

AIN0, Analog Comparator Positive Input. This pin also serves as the positive input of the on-chip analog comparator.

RXD1, receive Data (Data input pin for the UART1). When the UART1 receiver is enabled this pin is configured as an input regardless of the value of DDRB2. When the UART1 forces this pin to be an input, a logical one in PORTB2 will turn on the internal pull-up.

- **OC2/T1 - Port B, Bit 1**

T1, Timer/Counter1 counter source. See the “Timer/Counter1.” on page 45 for further details.

OC2, Output compare match output: The PB1 pin can serve as an external output when the Timer/Counter2 compare matches. The PB1 pin has to be configured as an output (DDB1 set (one)) to serve this function. See “8-bit Timers/Counters T/C0 and T/C2” on page 37 for further details. The OC2 pin is also the output pin for the PWM mode timer function.

- **OC0/T0 - Port B, Bit 0**

T0: Timer/Counter0 counter source. See the “8-bit Timers/Counters T/C0 and T/C2” on page 37 further details.

OC0, Output compare match output: The PB0 pin can serve as an external output when the Timer/Counter0 compare matches. The PB0 pin has to be configured as an output (DDB0 set (one)) to serve this function. See “8-bit Timers/Counters T/C0 and T/C2” on page 37 for further details, and how to enable the output. The OC0 pin is also the output pin for the PWM mode timer function.

Port B Schematics

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

Figure 57. Port B Schematic Diagram (Pins PB0 and PB1)

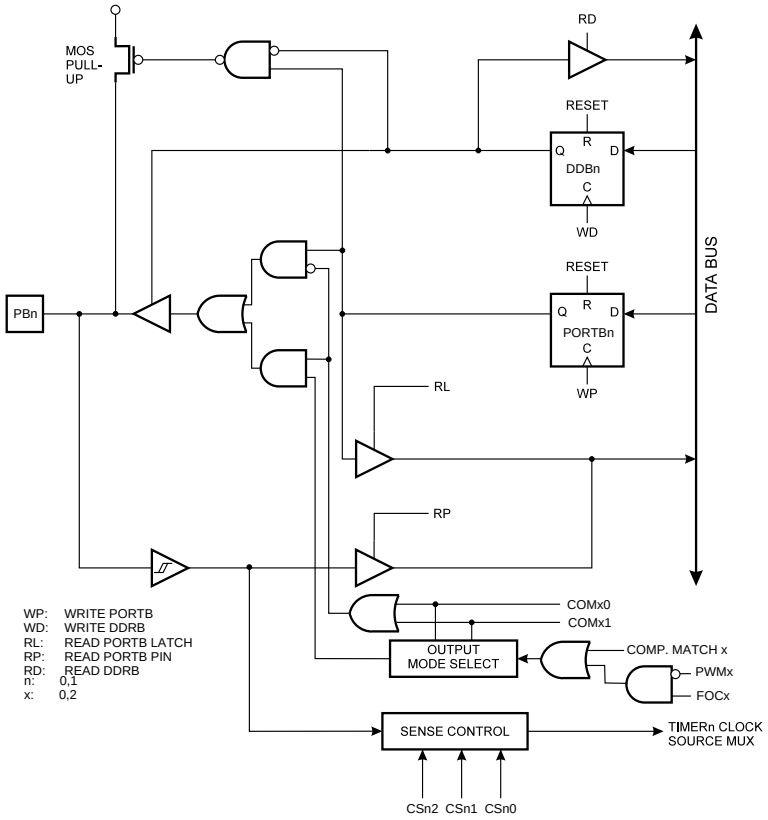


Figure 58. Port B Schematic Diagram (Pin PB2)

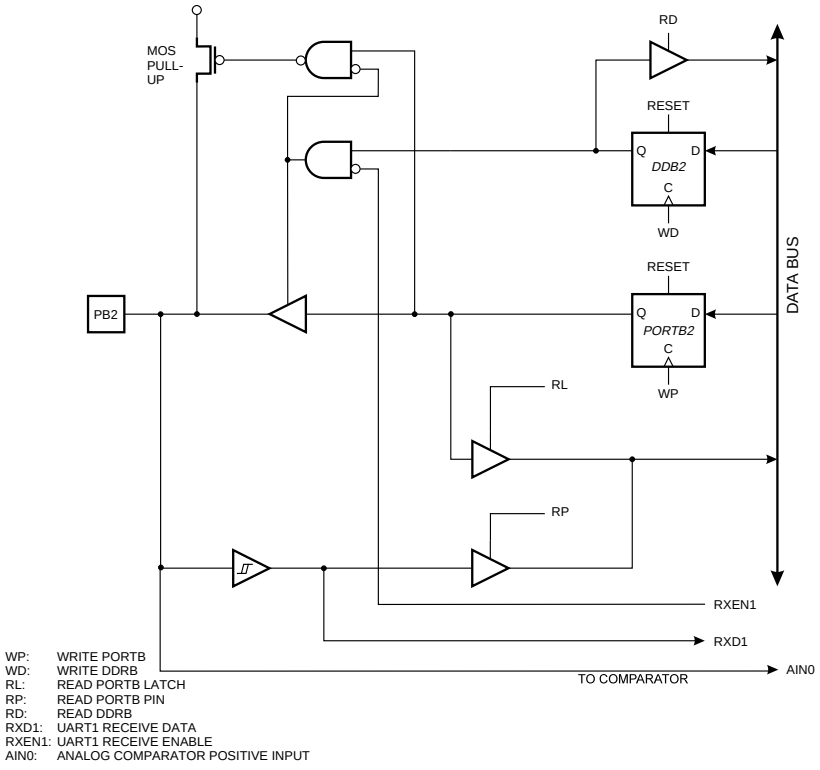


Figure 59. Port B Schematic Diagram (Pin PB3)

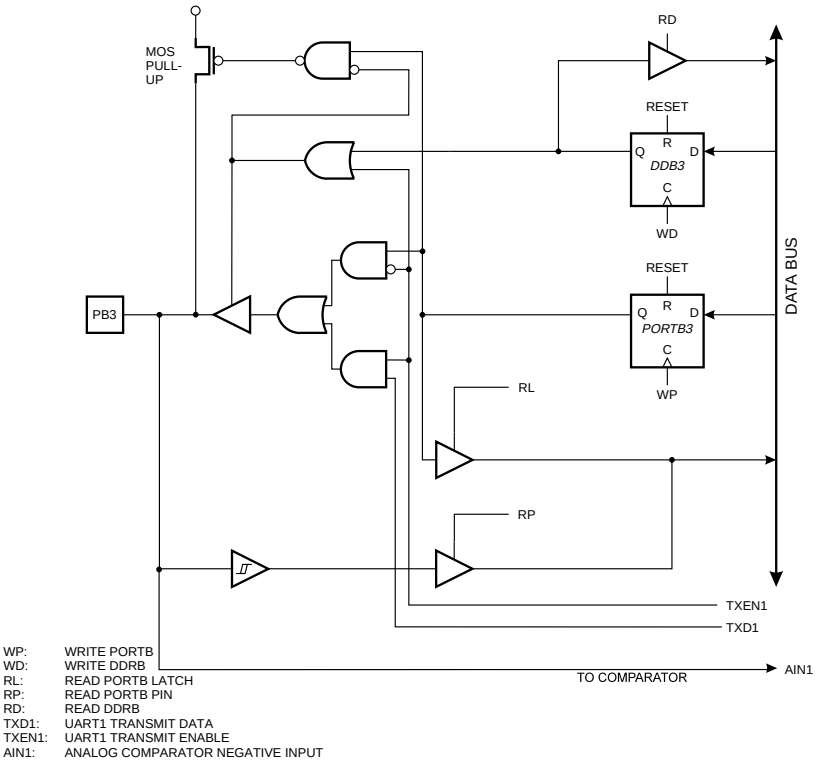


Figure 60. Port B Schematic Diagram (Pin PB4)

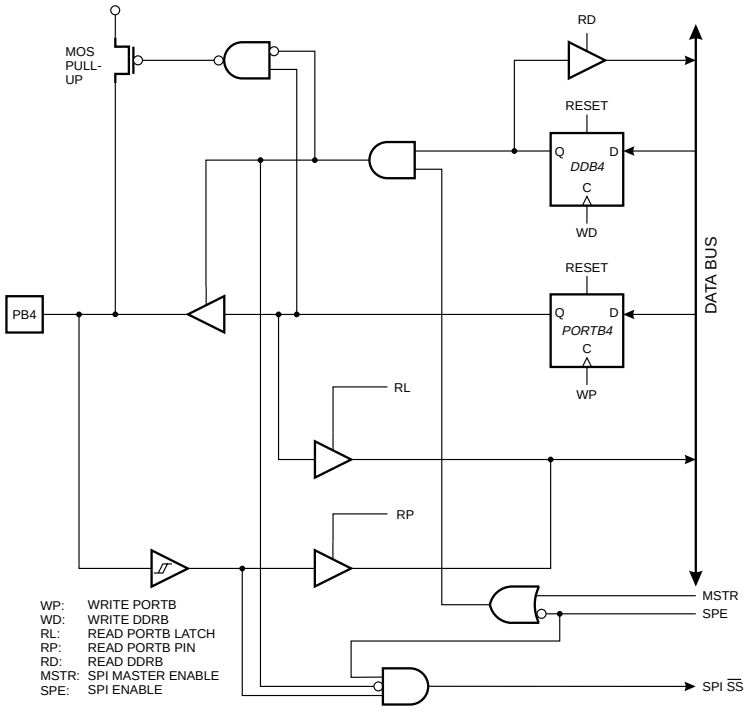


Figure 61. Port B Schematic Diagram (Pin PB5)

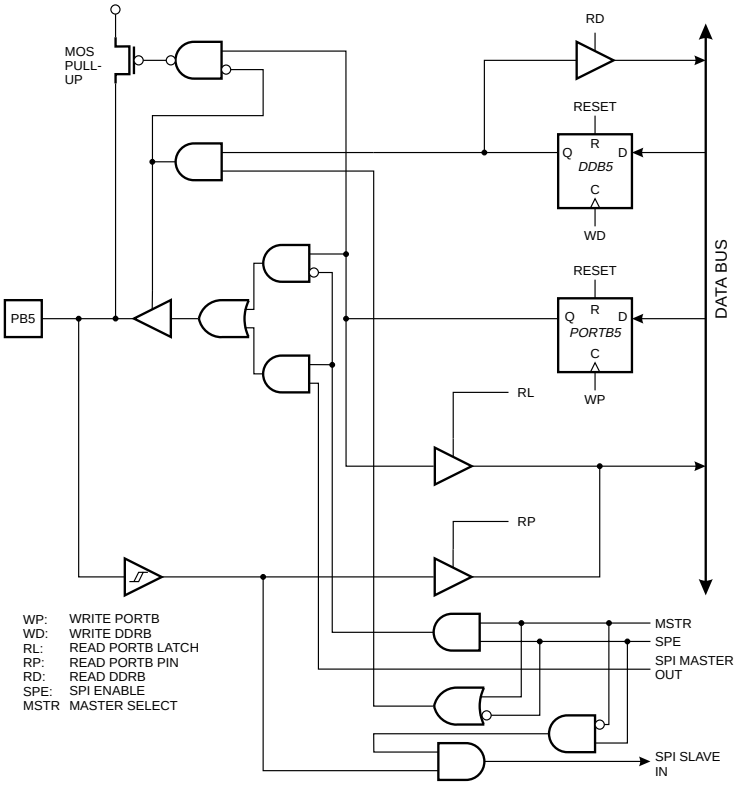


Figure 62. Port B Schematic Diagram (Pin PB6)

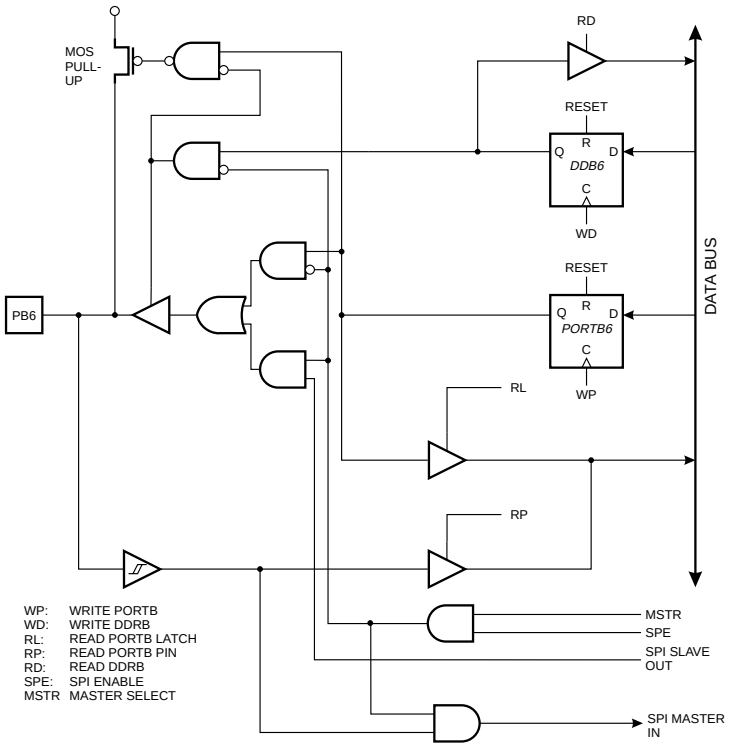
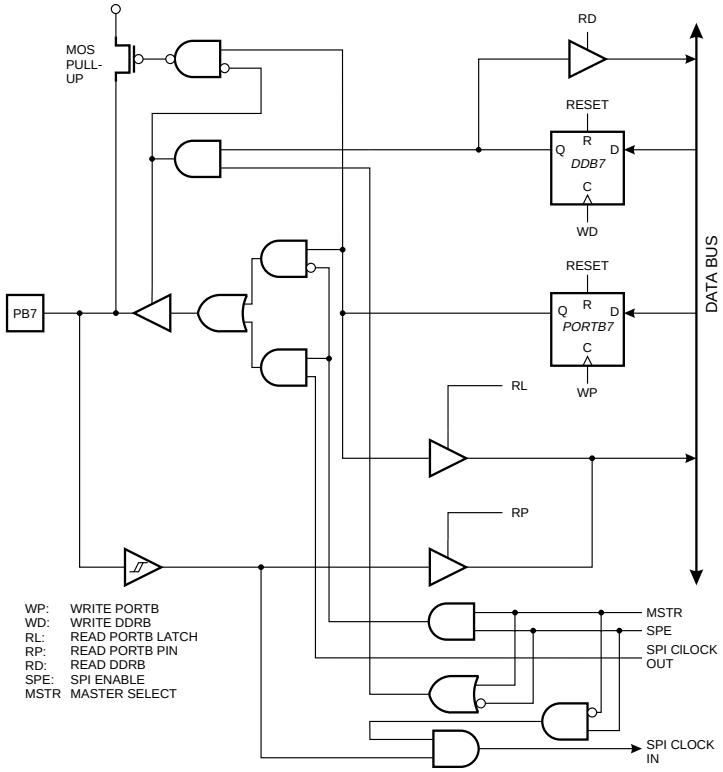


Figure 63. Port B Schematic Diagram (Pin PB7)



Port C

Port C is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for the Port C, one each for the Data Register – PORTC, $15($35), Data Direction Register – DDRC, $14($34) and the Port C Input Pins – PINC, $13($33). The Port C 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 C output buffers can sink 20 mA and thus drive LED displays directly. When pins PC0 to PC7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

The Port C pins have alternate functions related to the optional external memory interface. Port C can be configured to be the high-order address byte during accesses to external data memory.

When Port C 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.

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 Data Direction Register – DDRC

Bit	7	6	5	4	3	2	1	0	
$14 ($34)	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
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 Input Pins Address – PINC

Bit	7	6	5	4	3	2	1	0	
$13 ($33)	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
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 C Input Pins address – PINC – is not a register, and this address enables access to the physical value on each Port C pin. When reading PORTC, the Port C Data Latch is read, and when reading PINC, the logical values present on the pins are read.

Port C as General Digital I/O

All 8 pins in Port C have equal functionality when used as digital I/O pins.

PCn, General I/O pin: The DDCn bit in the DDRC register selects the direction of this pin, if DDCn is set (one), PCn is configured as an output pin. If DDCn is cleared (zero), PCn is configured as an input pin. If PORTCn 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, PORTCn has to be cleared (zero) or the pin has to be configured as an output pin.The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Table 31. DDCn Effects on Port C Pins

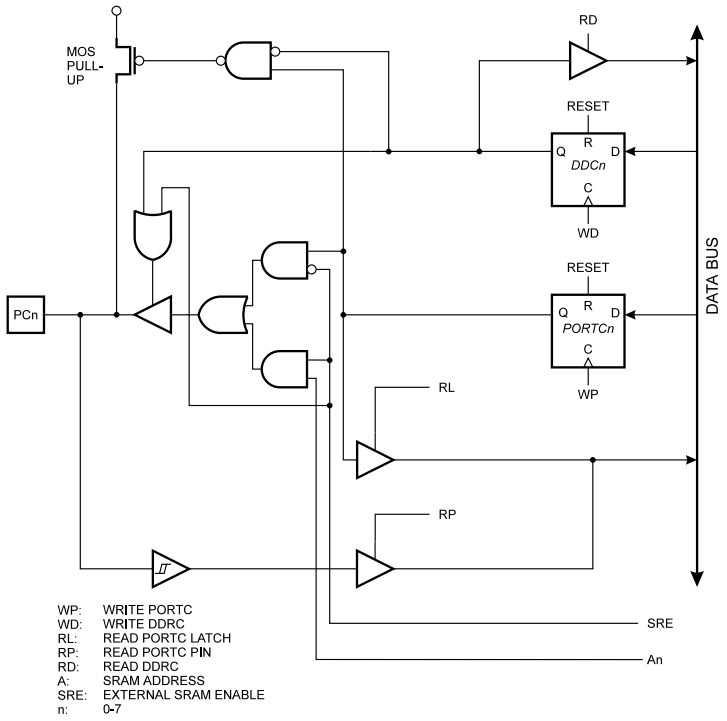
DDCn	PORTCn	I/O	Pull-up	Comment
0	0	Input	No	Tri-state (Hi-Z)
0	1	Input	Yes	PCn will source current if ext. pulled low.
1	0	Output	No	Push-pull Zero Output
1	1	Output	No	Push-pull One Output

n: 7, 6,…0, pin number

Port C Schematics

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

Figure 64. Port C Schematic Diagram (Pins PC0 - PC7)



Port D

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

Three I/O 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 20 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 32. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD0	RXD0 (UART0 Input line)
PD1	TXD0 (UART0 Output line)
PD2	INT0 (External interrupt 0 input)
PD3	INT1 (External interrupt 1 input)
PD3	TOSC1 (RTC oscillator Timer/Counter 2)
PD5	TOSC2 (RTC oscillator Timer/Counter 2)/OC1A (Timer/Counter1 Output compareA match output)
PD6	$\overline{WR}$ (Write strobe to external memory)
PD7	$\overline{RD}$ (Read strobe to external memory)

When the PD5 pin is used for the alternate function (OC1A) the DDRD and PORTD register has to be set according to the alternate function description.

Port D Data Register – PORTD

Bit	7	6	5	4	3	2	1	0	
$12 ($32)	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	

Port D Data Direction Register – DDRD

Bit	7	6	5	4	3	2	1	0	
$11 ($31)	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	

Port D Input Pins Address – PIND

Bit	7	6	5	4	3	2	1	0	
$10 ($30)	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 Port D Data Latch is read, and when reading PIND, the logical values present on the pins are read.

Port D 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 PORTDn is set (one) when configured as an input pin the MOS pull-up resistor is activated. To switch the pull-up resistor off, the PORTDn has to be cleared (zero) or the pin has to be configured as an output pin. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Table 33. 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 Port D

- **$\overline{RD}$ - Port D, Bit 7**

$\overline{RD}$ is the external data memory read control strobe.

- **$\overline{WR}$ - Port D, Bit 6**

$\overline{WR}$ is the external data memory write control strobe.

- **OC1 - Port D, Bit 5**

OC1, Output compare match output: The PD5 pin can serve as an external output when the Timer/Counter1 compare matches. The PD5 pin has to be configured as an output (DDD5 set (one)) to serve this function. See “Timer/Counter1.” on page 45 for further details, and how to enable the output. The OC1 pin is also the output pin for the PWM mode timer function.

- **TOSC1/TOSC2 - Port D, Bit 5 and 4**

When the AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter2, pins PD5 and PD4 are disconnected from the port. In this mode, a crystal oscillator is connected to the pins, and the pins can not be used as I/O pins.

- **INT1 - Port D, Bit 3**

INT1, External Interrupt source 1: The PD3 pin can serve as an external interrupt source to the MCU. See “MCU Control Register – MCUCR” on page 32 for further details.

- **INT0 - Port D, Bit 2**

INT0, External Interrupt source 0: The PD2 pin can serve as an external interrupt source to the MCU. See “MCU Control Register – MCUCR” on page 32 for further details.

- **TXD0 - Port D, Bit 1**

Transmit Data (Data output pin for the UART0). When the UART0 transmitter is enabled, this pin is configured as an output regardless of the value of DDRD1.

- **RXD0 - Port D, Bit 0**

Receive Data (Data input pin for the UART0). When the UART receiver is enabled this pin is configured as an input regardless of the value of DDRD0. When the UART0 forces this pin to be an input, a logical one in PORTD0 will turn on the internal pull-up.

Port D Schematics

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

Figure 65. Port D Schematic Diagram (Pin PD0)

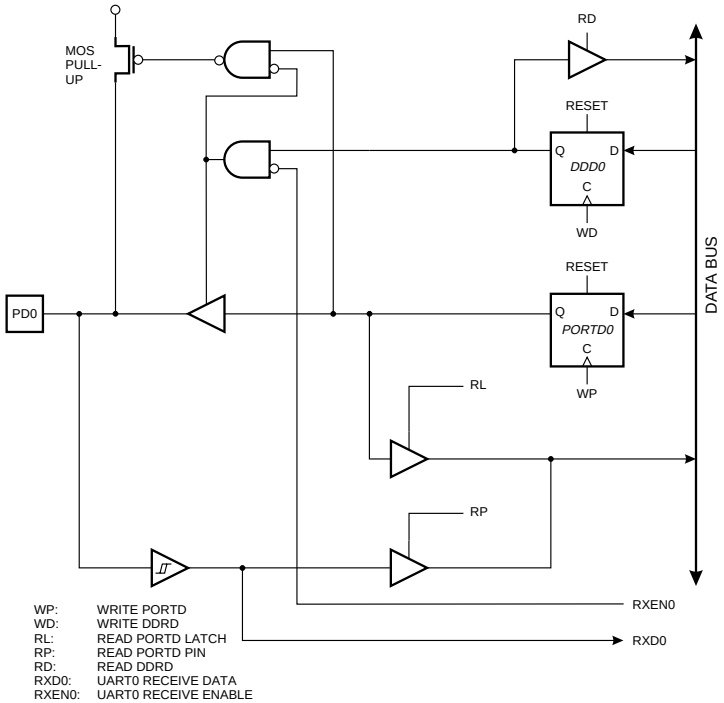


Figure 66. Port D Schematic Diagram (Pin PD1)

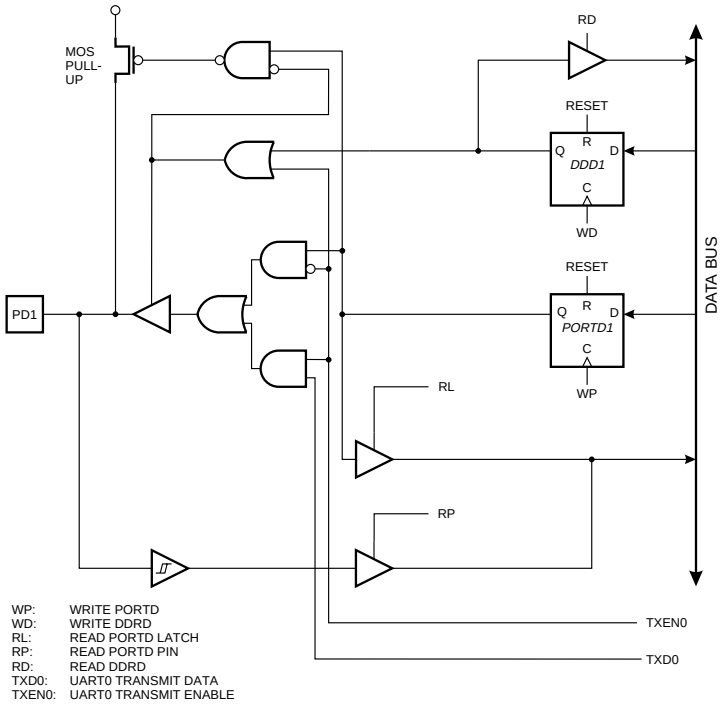


Figure 67. Port D Schematic Diagram (Pins PD2 and PD3)

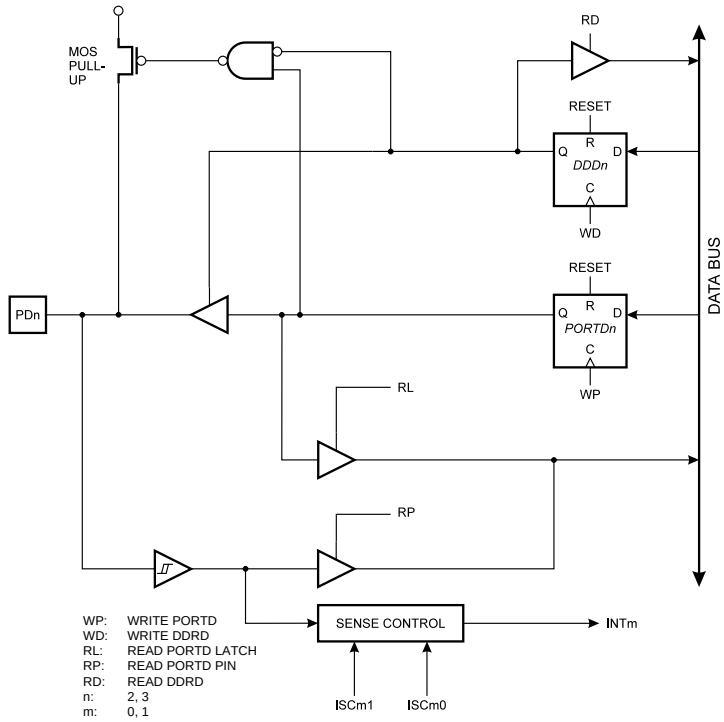


Figure 68. Port D Schematic Diagram (Pin PD4)

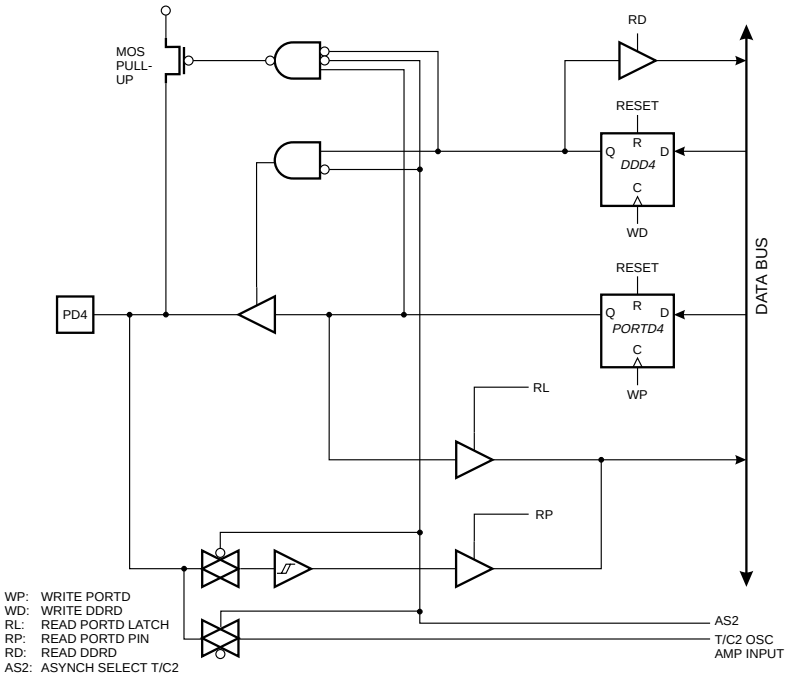


Figure 69. Port D Schematic Diagram (Pin PD5)

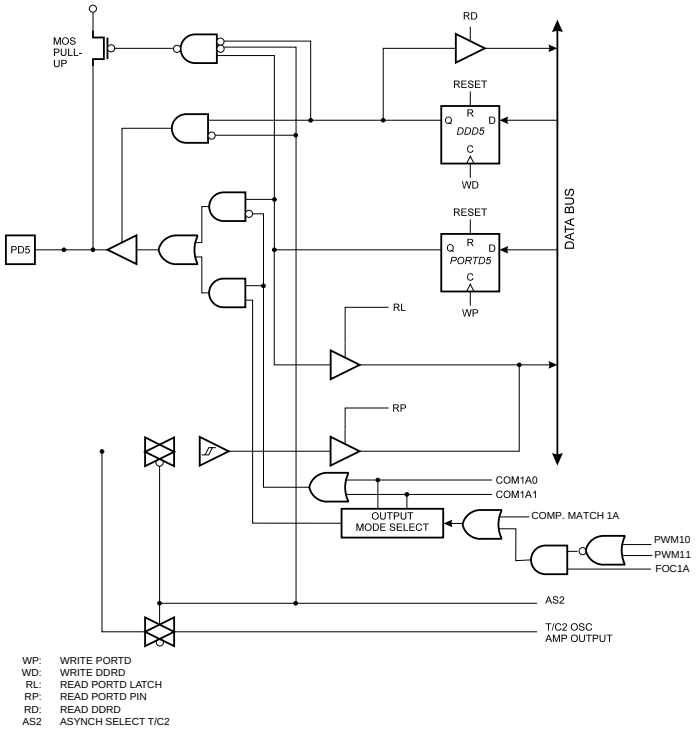


Figure 70. Port D Schematic Diagram (Pin PD6)

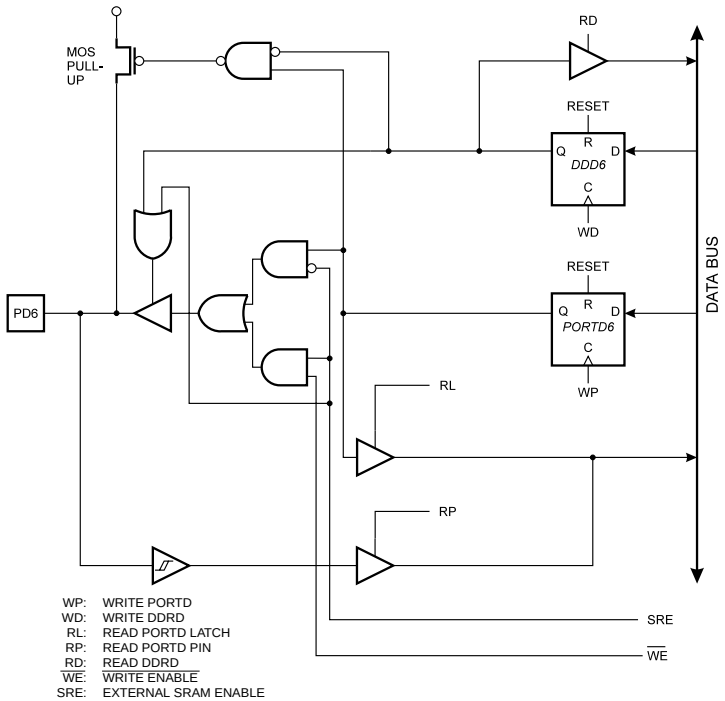
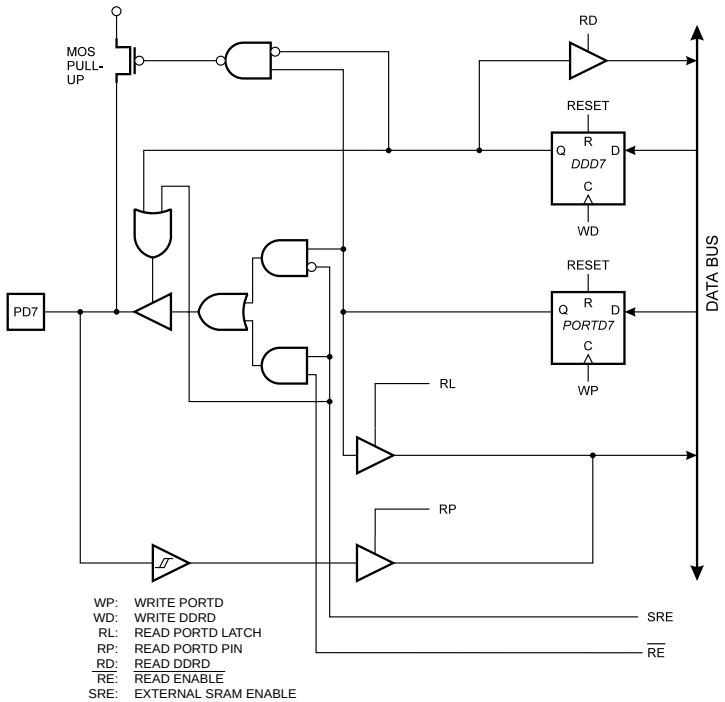


Figure 71. Port D Schematic Diagram (Pin PD7)



Port E

Port E is a 3 bit bi-directional I/O port with internal pull-up resistors.

Three I/O address locations are allocated for the Port E, one each for the Data Register – PORTE, $07($27), Data Direction Register – DDRE, $06($26) and the Port E Input Pins – PINE, $05($25). 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 20 mA. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated.

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

Table 34. Port E Pins Alternate Functions

Port Pin	Alternate Function
PE0	ICP (Input Capture Pin Timer/Counter1)/INT2 (External Interrupt 2 input)
PE1	OC1B (Timer/Counter1 Output CompareB match output)
PE2	ALE (Address Latch Enable, External Memory)

When the PE1 pin is used for the alternate function the DDRE and PORTE register has to be set according to the alternate function description.

Port E Data Register – PORTE

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	-	-	-	-	-	PORTE2	PORTE1	PORTE0	PORTE
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Port E Data Direction Register – DDRE

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	-	-	-	-	-	DDE2	DDE1	DDE0	DDRE
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Port E Input Pins Address – PINE

Bit	7	6	5	4	3	2	1	0	
$05 ($25)	-	-	-	-	-	PINE2	PINE1	PINE0	PINE
Read/Write	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	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 Port E Data Latch is read, and when reading PINE, the logical values present on the pins are read.

Port E as General Digital I/O

PEn, General I/O pin: The DDEn bit in the DDRE 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 PORTEn is set (one) when configured as an input pin the MOS pull-up resistor is activated. To switch the pull-up resistor off the PORTEn has to be cleared (zero) or the pin has to be configured as an output pin.The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Table 35. 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	PEn will source current if ext. pulled low.
1	0	Output	No	Push-pull Zero Output
1	1	Output	No	Push-pull One Output

n: 2,1,0, pin number.

Alternate functions of Port E

- **OC1B - Port E, Bit 2**

OC1B, Output compare match output: The PE2 pin can serve as an external output when the Timer/Counter1 compare matches. The PE2 pin has to be configured as an output (DDE2 set (one)) to serve this function. See “Timer/Counter1.” on page 45 for further details. The OC1B pin is also the output pin for the PWM mode timer function.

- **ALE - Port E, Bit 1**

ALE: When the External Memory is enabled, the PE1 pin serves as the Dress Latch Enable. Note that enabling of External Memory will override both the direction and port value. See “Interface to external memory” on page 72 for a detailed description.

- **ICP/INT2 - Port E, Bit 0**

ICP, input capture pin: The PE0 pin can serve as the input capture source for Timer/Counter 1. See page 49 for a detailed description.

INT2, External Interrupt source 2: The PE0 pin can serve as an external interrupt source to the MCU. See “Extended MCU Control Register – EMCUCR” on page 33 for further details.

Port E Schematics

Figure 72. Port E Schematic Diagram (Pin PE0)

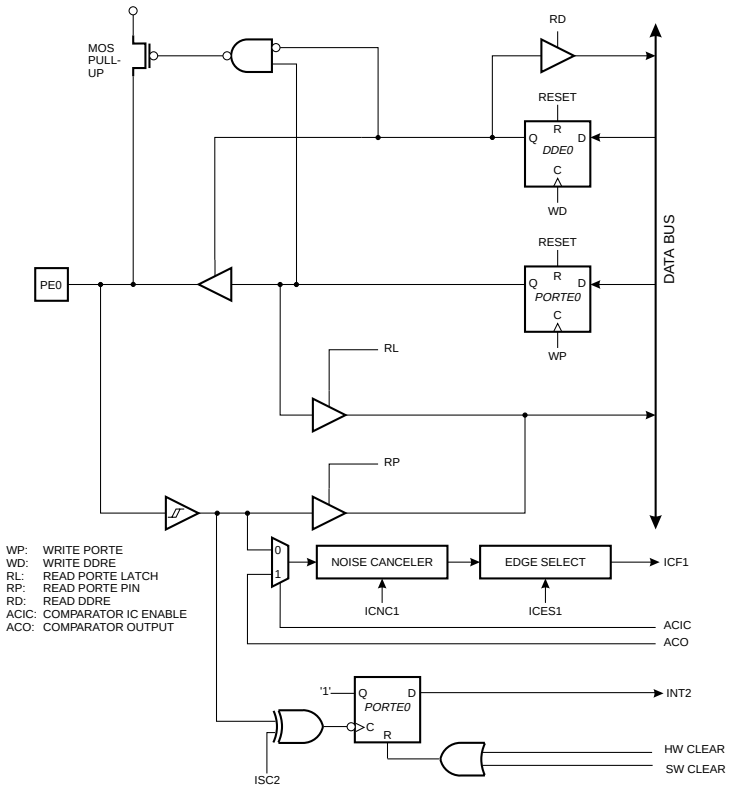


Figure 73. Port E Schematic Diagram (Pin PE1)

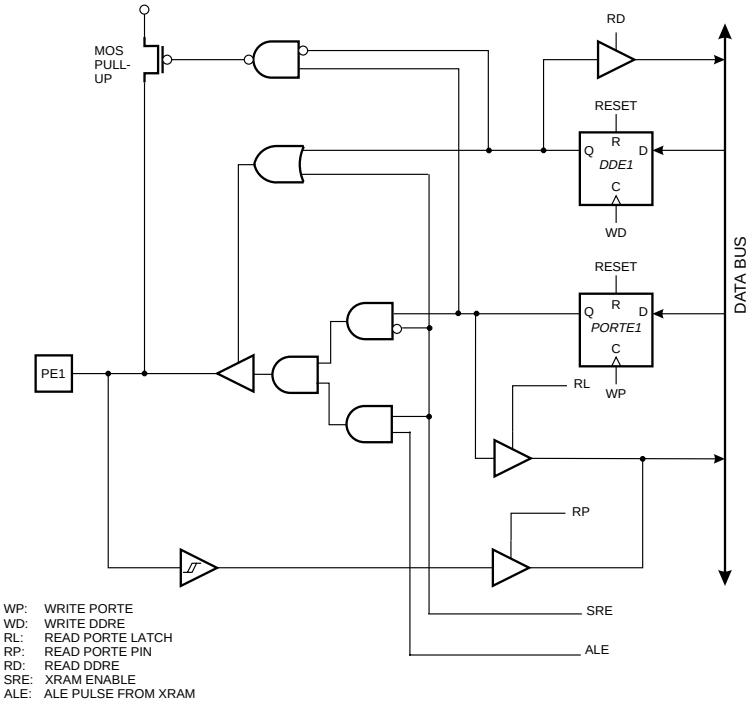
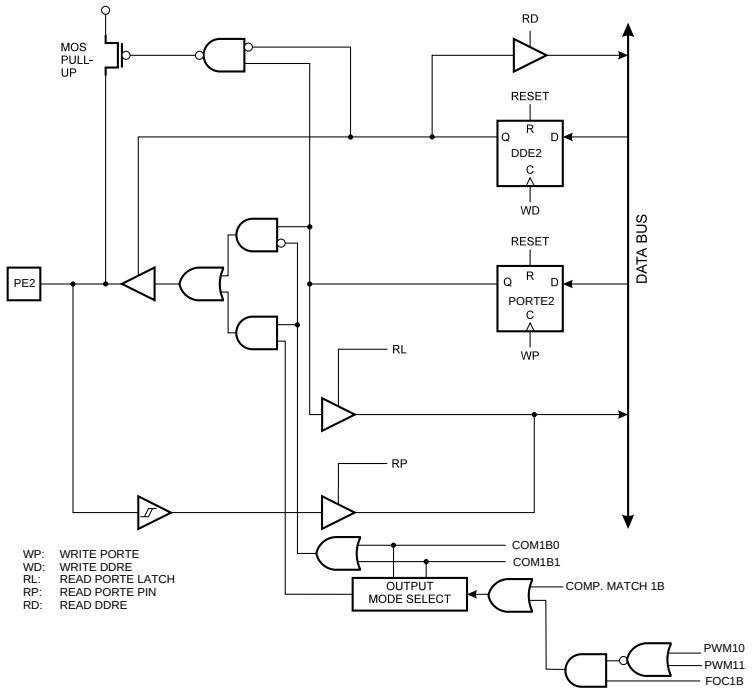


Figure 74. Port E Schematic Diagram (Pin PE2



Memory Programming

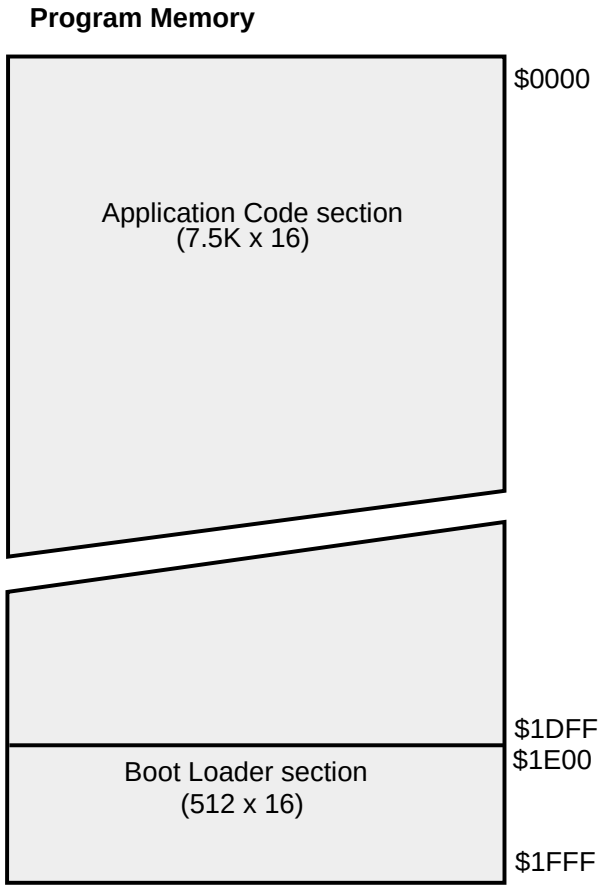
Boot Loader Support

The ATmega161 provides a mechanism for downloading and uploading program code by the MCU itself. This feature allows flexible application software updates, controlled by the MCU using a Flash-resident Boot Loader program.

The ATmega161 FLASH memory is organized in two main sections;

1. The Application code section (address $0000 - $1DFF)
2. The Boot Loader section/Boot block (address $1E00 - $1FFF)

Figure 75. Memory sections



Boot Loader program can use any available data interface and associated protocol, such as UART serial bus interface, to input or output program code, and write (program) that code into the Flash memory, or read the code from the program memory.

The program Flash memory is divided into pages that contains 128 bytes each. The Boot Loader FLASH section occupies 8 pages from $1E00 to $1FFF by 16 bit words.

The Store Program Memory (SPM) instruction can access the entire FLASH, but it can only be executed from the Boot Loader FLASH section. If no Boot Loader capability is needed, the entire FLASH is available for application code. The ATmega161 has two separate sets of Boot Lock Bits which can be set independently. This gives the user a unique flexibility to select different levels of protection. The user can select:

- **To protect the entire FLASH from a software update by the Boot Loader Program.**
- **To only protect the Boot Loader section from a software update by the Boot Loader Program.**
- **To only protect the Application code section from a software update by the Boot Loader Program.**
- **Allowing software update in the entire FLASH**

See Table 36 and Table 37 for further details. The Boot Lock bits can be set in software and in Serial or Parallel Programming mode, but they can only be cleared by a chip erase command.

Table 36. Boot Lock Bit0 Protection modes (Application section)

BLB0 mode	BLB01	BLB02	Protection
1	1	1	No restrictions for SPM, LPM in the Application code section (address $0000 - $1DFF)
2	0	1	It is not allowed to update the Application code section (address $0000 - $1DFF) by SPM.
3	0	0	LPM read and SPM write prohibited in the Application code section (address $0000 - $1DFF)
4	1	0	It is not allowed to read program code located in the Application code section (address $0000 - $1DFF) by LPM

Note: ’1’ means unprogrammed, ‘0’ means programmed

Table 37. Boot Lock Bit1 Protection modes (Boot Loader section)

BLB1 mode	BLB11	BLB12	Protection
1	1	1	No restrictions for SPM, LPM in the Boot Loader section (address $1E00 - $1FFF)
2	0	1	It is not allowed to update the Boot Loader section (address $1E00 - $1FFF) by SPM
3	0	0	LPM and SPM prohibited in the Boot Loader section (address $1E00 - $1FFF)
4	1	0	It is not allowed to read program code located in the Boot Loader section (address $1E00 - $1FFF) by LPM

Note: ’1’ means unprogrammed, ‘0’ means programmed

Entering the Boot Loader Program

Entering the Boot Loader takes place by a jump or call from the application program. This may be initiated by some trigger such as a command received via UART or SPI interface. Alternatively, the Boot Reset Fuse (BOOTRST) can be programmed so that the reset vector is pointing to address $1E00 after a reset. In this case, the Boot Loader is started after the reset. After the application code is loaded, the program can start executing the application code. Note that the fuses cannot be changed by the MCU itself. This means that once the Boot Reset Fuse is programmed, the Reset Vector will always point to the Boot Loader Reset and the fuse can only be changed through the serial or parallel programming interface. The BOOTRST fuse can also be locked by programming LB1. When LB1 is programmed it is not possible to change the BOOTRST fuse unless a chip erase command is performed first.

Table 38. Boot Reset Fuse, BOOTRST

BOOTRST	Reset Address
1	Reset Vector = Application reset (address $0000)
0	Reset Vector = Boot Loader reset (address $1E00)

Note: ’1’ means unprogrammed, ‘0’ means programmed

Capabilities of the Boot Loader

The program code within the Boot Loader section has the capability to read from and write into the entire FLASH, including the Boot Loader Memory. This allows the user to update both the Application code and the Boot Loader code that handles the software update. The Boot Loader can thus even modify itself, and it can also erase itself from the code if the feature is not needed anymore. Special care must be taken if the user allows the Boot Loader section to be updated by leaving Boot Lock bit11 unprogrammed. An accidental write to the Boot Loader itself can corrupt the entire Boot Loader, and further software updates might be impossible. If it is not needed to change the Boot Loader software itself, it is recommended to program the Boot Lock bit11 to protect the Boot Loader software from software changes.

Self-programming the Flash

Programming of the Flash is executed one page at a time. The Flash page must be erased first for correct programming. The general Write Lock (Lock Bit 2) does not control the programming of the Flash memory by SPM instruction. Similarly, the Read/Write Lock (Lock Bit 1) does not control reading nor writing by LPM/SPM, if it is attempted.

The program memory can only be updated page by page, not word by word. One page is 128 bytes (64 words). The program memory will be modified by first performing page erase, then filling the temporary page buffer one word at a time using SPM, and then executing page write. If only part of the page needs to be changed, the other parts must be stored (for example in the temporary page buffer) before the erase, and then be rewritten. The temporary page buffer can be accessed in a random sequence. The CPU is halted both during page erase and during page write. It is essential that the page address used in both the page erase and page write operation is addressing the same page.

- **Setting the Boot Loader Lock Bits by SPM**

To set the Boot Loader Lock bits, write the desired data to R0, write "1001" to SPMCR and execute SPM within four clock cycles after writing SPMCR. The only accessible lock bits are the Boot Lock bits that may prevent the Application and Boot Loader section from any software update by the MCU. See Table 36 and Table 37 how the different settings of the Boot Loader Bits affect the FLASH access.

Bit	7	6	5	4	3	2	1	0	
	-	-	BLB12	BLB11	BLB02	BLB01	-	-	R0

If bit5 - bit2 in R0 is cleared (zero), the corresponding Boot Lock Bit will be programmed if a SPM instruction is executed within four cycles after BLBSET and SPMEN are set in SPMCR.

- **Performing Page Erase by SPM**

To execute page erase, set up the address in the Z pointer, write "0011" to SPMCR and execute SPM within four clock cycles after writing SPMCR. The data in R1 and R0 are ignored. The page address must be written to Z13:Z7. Other bits in the Z pointer will be ignored during this operation.

- **Fill the temporary buffer**

To write an instruction word, set up the address in the Z pointer and data in R1:R0, write "0001" to SPMCR and execute SPM within four clock cycles after writing SPMCR. The content of Z6:Z1 is used to address the data in the temporary buffer. Z13:Z7 must point to the page that is supposed to be written.

- **Perform a Page Write**

To execute page write, set up the address in the Z pointer, write "0101" to SPMCR and execute SPM within four clock cycles after writing SPMCR. The data in R1 and R0 are ignored. The page address must be written to Z13:Z7. During this operation, Z6:Z0 must be zero to ensure that the page is written correctly.

Addressing the FLASH During Self-programming

The Z pointer is used to address the SPM commands.

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

Z15:Z14 always ignored

Z13:Z7 page select, for page erase, page write

Z6:Z1 word select, for filling temp buffer (must be zero during page write operation)

Z0 should be zero for all SPM commands, byte select for the LPM instruction.

The only operation that does not use the Z pointer is Setting the Boot Loader Lock Bits. The content of the Z pointer is ignored and will have no effect on the operation.

Note that the page erase and page write operation is addressed independently. Therefore it is of major importance that the Boot Loader software addresses the same page in both the page erase and page write operation.

The LPM instruction does also use the Z pointer to store the address. Since this instruction does address the FLASH byte by byte, also the LSB (bit Z0) of the Z pointer is used. See page 16 for a detailed description.

Accidental writing into Flash program by the SPM instruction is prevented by setting up a "SPM enable time window". All accesses are executed by first setting I/O bits, and then executing SPM within four clock cycles. The I/O register that controls the SPM accesses is defined as follows:

Store Program Memory Control Register – SPMCR

The Store Program Memory Control Register contains the control bits needed to control the programming of the FLASH from internal code execution.

Bit	7	6	5	4	3	2	1	0	
$37 ($57)	**-**	**-**			**BLBSET**	**PGWRT**	**PGERS**	**SPMEN**	**SPMCR**
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7..4 - Res: Reserved Bits**

These bits are reserved bits in the ATmega161 and always read as zero.

- **Bit 3 - BLBSET: Boot Lock Bit set**

If this bit is set at the same time as SPMEN, the next SPM instruction within four clock cycles sets Boot Lock bits, according to the data in R0. The data in R1 and the address in the Z pointer are ignored. The BLBSET bit will auto-clear upon completion of lock bit set, or if no SPM instruction is executed within four clock cycles. The CPU is halted during lock bit setting. Only a chip erase can clear the Lock Bits.

An LPM instruction within four cycles after BLBSET and SPMEN are set in the SPMCR register, will put either the Lock-bits or the Fuse-bits (depending od the Z0 in the Z-pointer) into the destination register. See “Reading the Fuse- and Lock Bits from Software” on page 99 for details.

- **Bit 2 - PGWRT: Page write**

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

- **Bit 1 - PGERS: Page Erase**

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

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

This bit enables the SPM instruction for the next four clock cycles. If set together with either BLBSET, PGWRT or PGERS, the following SPM instruction will have a special meaning, see description above. If only SPMEN is set, the following SPM instruction will store the value in R1:R0 in the temporary page buffer addressed by the Z pointer. The LSB of the Z pointer is ignored. The SPMEN bit will auto-clear upon completion of an SPM instruction, or if no SPM instruction is executed within four clock cycles.

Writing any other combination that "1001", "0101", "0011" or "0001" in the lower four bits, or writing to the I/O register when any bits are set, will have no effect.

EEPROM Write Prevents Writing to SPMCR

Note that an EEPROM write operation will block all software programming to Flash. Reading the Fuses and Lockbits from software will also be prevented during the EEPROM write operation. It is recommended that the user checks the status bit (EEWE) in the EECR register and verifies that the bit is cleared before writing to the SPMCR register.

Reading the Fuse- and Lock Bits from Software

It is possible to read both the Fuse and Lock bits from software. To read the Lock bits, load the Z-pointer with $0001 and set the BLBSET and SPMEN bits in SPMCR. If an LPM instruction is executed within three CPU cycles after the BLBSET and SPMEN bits are set in SPMCR, the Lock bits will be written to the destination register. The BLBSET and SPMEN bits will auto-clear upon completion of reading the Lock bits or if no LPM/SPM instruction is executed within three/four CPU cycles. When BLBSET and SPMEN are cleared, LPM will work as described in “Constant Addressing Using the LPM Instruction” on page 16 and in the Instruction set Manual.

Bit	7	6	5	4	3	2	1	0	
	-	-	BLB12	BLB11	BLB02	BLB01	LB2	LB1	R0/Rd

The algorithm for reading the Fuse bits is similar to the one described above for reading the Lock bits. But when reading the Fuse bits, load $0000 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SPMEN bits are set in the SPMCR, the Fuse-bits can be read in the destination register as shown below.

Bit	7	6	5	4	3	2	1	0	
	-	BOOTRST	SPIEN	BODLEVEL	BODEN	CKSEL[2]	CKSEL[1]	CKSEL[0]	R0/Rd

Fuse- and lock bits that are programmed, will be read as zero.

Program Memory Lock Bits

The ATmega161 MCU provides six Lock bits which can be left unprogrammed (‘1’) or can be programmed (‘0’) to obtain the additional features listed in Table 39. The Lock bits can only be erased to ‘1’ with the Chip Erase command.

Table 39. Lock Bit Protection Modes

Memory Lock Bits			Protection Type
LB mode	**LB1**	**LB2**	
1	1	1	No memory lock features enabled
2	0	1	Further programming of the Flash and EEPROM is disabled in parallel and serial programming mode. The Fuse bits are locked in both serial and parallel programming mode.(1)
3	0	0	Further programming and verification of the FLASH and EEPROM is disabled in parallel and serial programming mode. The Fuse bits are locked in both serial and parallel programming mode.(1).
BLB0 mode	**BLB01**	**BLB02**	
1	1	1	No memory lock features on SPM and LPM in Application code section (address $0000 - $1DFF).
2	0	1	SPM prohibited in the Application code section (address $0000 - $1DFF)
3	0	0	LPM and SPM prohibited in the Application code section (address $0000 - $1DFF)
4	1	0	LPM prohibited in the Application code section (address $0000 - $1DFF)
BLB1 mode	**BLB11**	**BLB12**	
1	1	1	No memory lock features on SPM and LPM in Boot Loader section (address $1E00 - $1FFF).
2	0	1	SPM prohibited in the Boot Loader section (address $1E00 - $1FFF)
3	0	0	LPM and SPM prohibited in the Boot Loader section (address $1E00 - $1FFF)
4	1	0	LPM prohibited in the Boot Loader section (address $1E00 - $1FFF)

Note: 1. Program the Fuse bits before programming the Lock bits.

Fuse Bits

The ATmega161 has seven fuse bits, BOOTRST, SPIEN, BODLEVEL, BODEN and CKSEL [2:0].

- When BOOTRST is programmed (‘0’), the reset vector is set to address $1E00 which is the first address location in the Boot Loader section of the FLASH. If the BOOTRST is unprogrammed (‘1’), the reset vector is set to address $0000. Default value is unprogrammed (‘1’).
- When the SPIEN Fuse is programmed (‘0’), Serial Program and Data Downloading is enabled. Default value is programmed (‘0’). The SPIEN Fuse is not accessible in serial programming mode.
- The BODLEVEL Fuse selects the Brown-out Detection Level and changes the Start-up times. See “Brown-out Detection” on page 27. Default value is unprogrammed (‘1’).
- When the BODEN Fuse is programmed (‘0’), the Brown-out Detector is enabled. See “Brown-out Detection” on page 27. Default value is unprogrammed (‘1’).
- CKSEL2..0: See Table 4, “Reset Delay Selections,” on page 25, for which combination of CKSEL2..0 to use. Default value is ‘010’.

The status of the Fuse bits is not affected by Chip Erase. Note that the Fuse bits are locked if lock bit1 (LB1) or lock bit2 (LB2) is programmed. Program the Fuse bits before programming the Lock bits.

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, and for the ATmega161 they are:

1. $000: $1E (indicates manufactured by Atmel)
2. $001: $94 (indicates 16KB Flash memory)
3. $002: $01 (indicates ATmega161 device when $001 is $94)

Programming the Flash and EEPROM

Atmel's ATmega161 offers 16K bytes of in-system reprogrammable Flash Program memory and 512 bytes of EEPROM Data memory.

The ATmega161 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 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 ATmega161 inside the user's system.

The Program memory array on the ATmega161 is organized as 128 pages of 128 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 in either programming mode.

The EEPROM Data memory array on the ATmega161 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.

During programming, the supply voltage must be in accordance with Table .

Table 40. Supply voltage during programming

Part	Serial programming	Parallel programming
ATmega161L	2.7 - 5.5V	4.5 - 5.5V
ATmega161	4.0 - 5.5V	4.5 - 5.5V

Parallel Programming

This section describes how to parallel program and verify Flash Program memory, EEPROM Data memory, Lock bits and Fuse bits in the ATmega161. Pulses are assumed to be at least 500ns unless otherwise noted.

Signal Names

In this section, some pins of the ATmega161 are referenced by signal names describing their functionality during parallel programming, see Figure 76 and Table 41. Pins not described in the following table are referenced by pin names.

The XA1/XA0 pins determine the action executed when the XTAL1 pin is given a positive pulse. The bit coding are shown in Table 42.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The Command is a byte where the different bits are assigned functions as shown in Table 43.

Figure 76. Parallel Programming

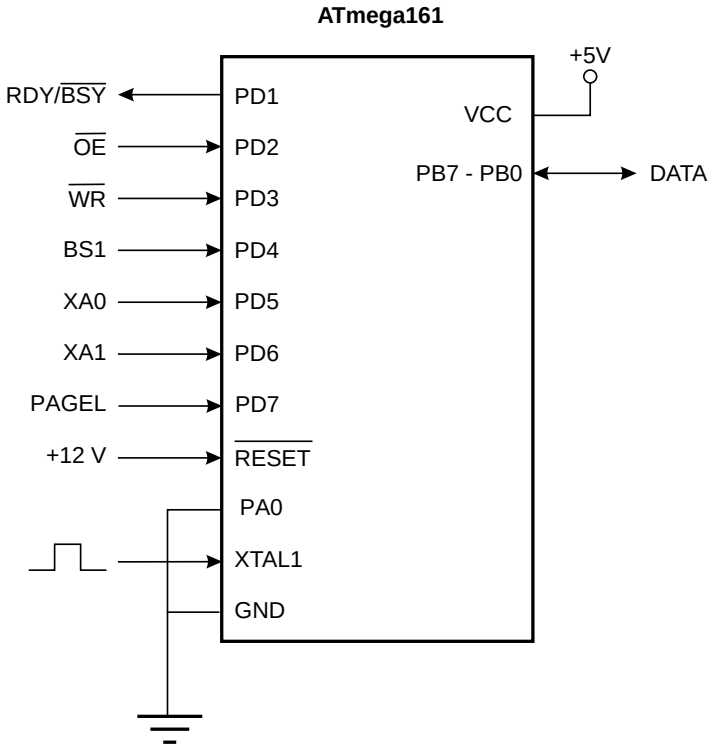


Table 41. Pin Name Mapping

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 1 (‘0’ selects low byte, ‘1’ selects high byte)
XA0	PD5	I	XTAL Action Bit 0
XA1	PD6	I	XTAL Action Bit 1
PAGEL	PD7	I	Program Memory Page Load
BS2	PA0	I	Byte Select 2 (Always low)
DATA	PB7-0	I/O	Bidirectional Databus (Output when $\overline{\text{OE}}$ is low)

Table 42. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash or EEPROM Address (High or low address byte 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

Table 43. Command Byte Bit Coding

Command Byte	Command Executed
1000 0000	Chip Erase
0100 0000	Write Fuse Bits
0010 0000	Write Lock Bits
0001 0000	Write Flash
0001 0001	Write EEPROM
0000 1000	Read Signature Bytes
0000 0100	Read Fuse and Lock Bits
0000 0010	Read Flash
0000 0011	Read EEPROM

Enter Programming Mode

The following algorithm puts the device in parallel programming mode:

1. Apply 4.5 - 5.5V between V_{CC} and GND.
2. Set $\overline{\text{RESET}}$ and BS pins to ‘0’ and wait at least 500 ns.
3. Apply 11.5 - 12.5V to $\overline{\text{RESET}}$, and wait for at least 500 ns.

Chip Erase

The Chip Erase will erase the Flash and EEPROM memories plus Lock bits. The Lock bits are not reset until the program memory has been completely erased. The Fuse bits are not changed. A Chip Erase must be performed before the Flash is reprogrammed.

Load Command “Chip Erase”

1. Set XA1, XA0 to ‘10’. This enables command loading.
2. Set BS1 to ‘0’.
3. Set DATA to ‘1000 0000’. This is the command for Chip Erase.
4. Give $\overline{\text{WR}}$ a negative pulse. This starts the Chip Erase. RDY/$\overline{\text{BSY}}$ goes low.
5. Wait until RDY/$\overline{\text{BSY}}$ goes high before loading a new command.

Programming the Flash

The Flash is organized as 128 pages of 128 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 "Write Flash"

1. Set XA1, XA0 to '10'. This enables command loading.
2. Set BS1 to '0'.
3. Set DATA to '0001 0000'. This is the command for Write Flash.
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 DATA = Address low byte ($00 - $FF).
4. Give XTAL1 a positive pulse. This loads the address low byte.

C. Load Data Low Byte

1. Set BS1 to ’0’. This selects low data byte.
2. Set XA1, XA0 to ’01’. This enables data loading.
3. Set DATA = Data low byte ($00 - $FF).
4. Give XTAL1 a positive pulse. This loads the data byte.

D. Latch Data Low Byte

Give PAGEL a positive pulse. This latches the data low byte.
(See Figure 77 for signal waveforms)

E. Load Data High Byte

1. Set BS1 to ’1’. This selects high data byte.
2. Set XA1, XA0 to ’01’. This enables data loading.
3. Set DATA = Data high byte ($00 - $FF).
4. Give XTAL1 a positive pulse. This loads the data byte.

F. Latch Data High Byte

Give PAGEL a positive pulse. This latches the data high byte.

G. Repeat B through F 64 times to fill the page buffer.

To address a page in the FLASH, 7 bits are needed (128 pages). The 5 most significant bits are read from address high byte as described in section ‘H’ below. The two least significant page address bits however, are the two most significant bits (bit7 and bit6) of the latest loaded address low byte as described in section ‘B’.

H. Load Address High byte

1. 1. Set XA1, XA0 to '00'. This enables address loading.
2. Set BS1 to '1'. This selects high address.
3. Set DATA = Address high byte ($00 - $1F).
4. Give XTAL1 a positive pulse. This loads the address high byte.

I. Program Page

1. Give $\overline{WR}$ a negative pulse. This starts programming of the entire page of data. RDY/$\overline{BSY}$goes low.
2. Wait until RDY/$\overline{BSY}$ goes high.
   (See Figure 78 for signal waveforms)

J. End Page Programming

1. 1. Set XA1, XA0 to '10'. This enables command loading.
2. Set DATA to '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 128 times or until all data have been programmed.

Figure 77. Programming the Flash waveforms

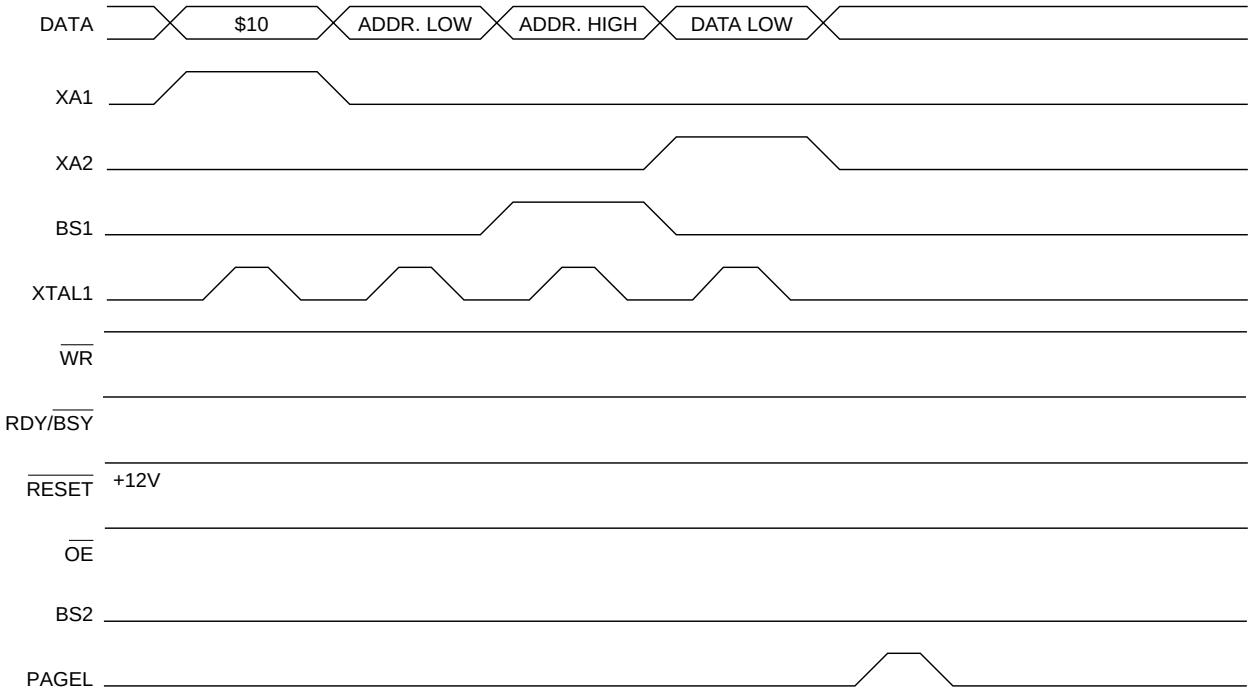
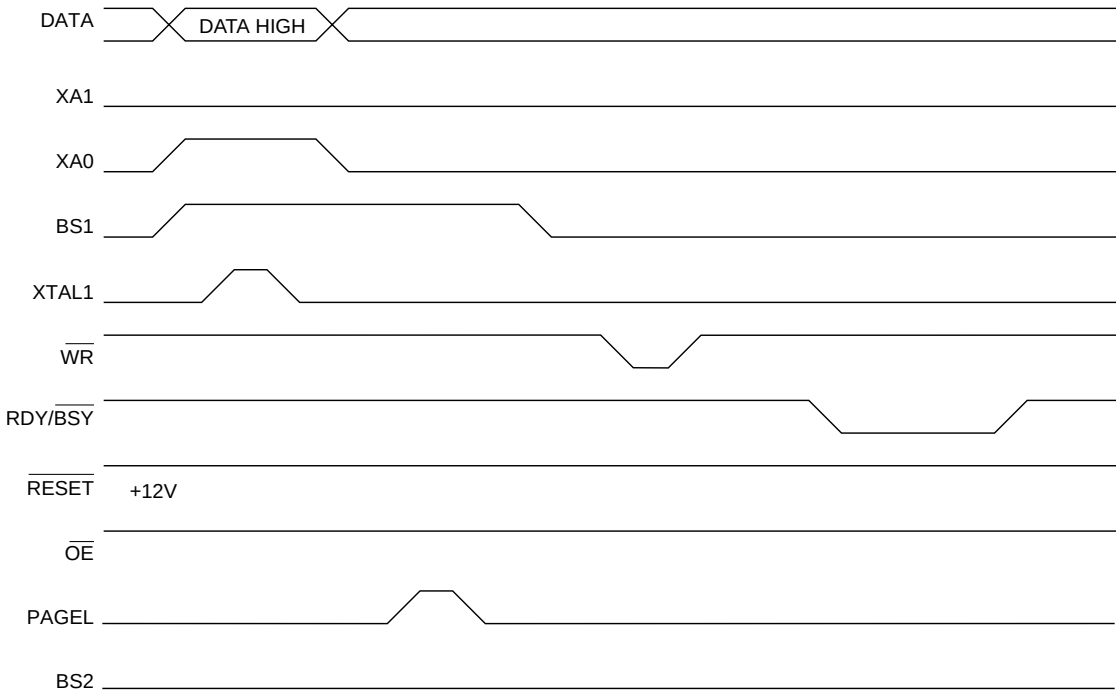


Figure 78. Programming the Flash waveforms (continued)



Programming the EEPROM

The programming algorithm for the EEPROM data memory is as follows (refer to “Programming the Flash” on page 103 for details on Command, Address and Data loading):

1. A: Load Command ‘0001 0001’.
2. H: Load Address High Byte ($00 - $01)
3. B: Load Address Low Byte ($00 - $FF)
4. E: Load Data Low Byte ($00 - $FF)

L: Write Data Low Byte

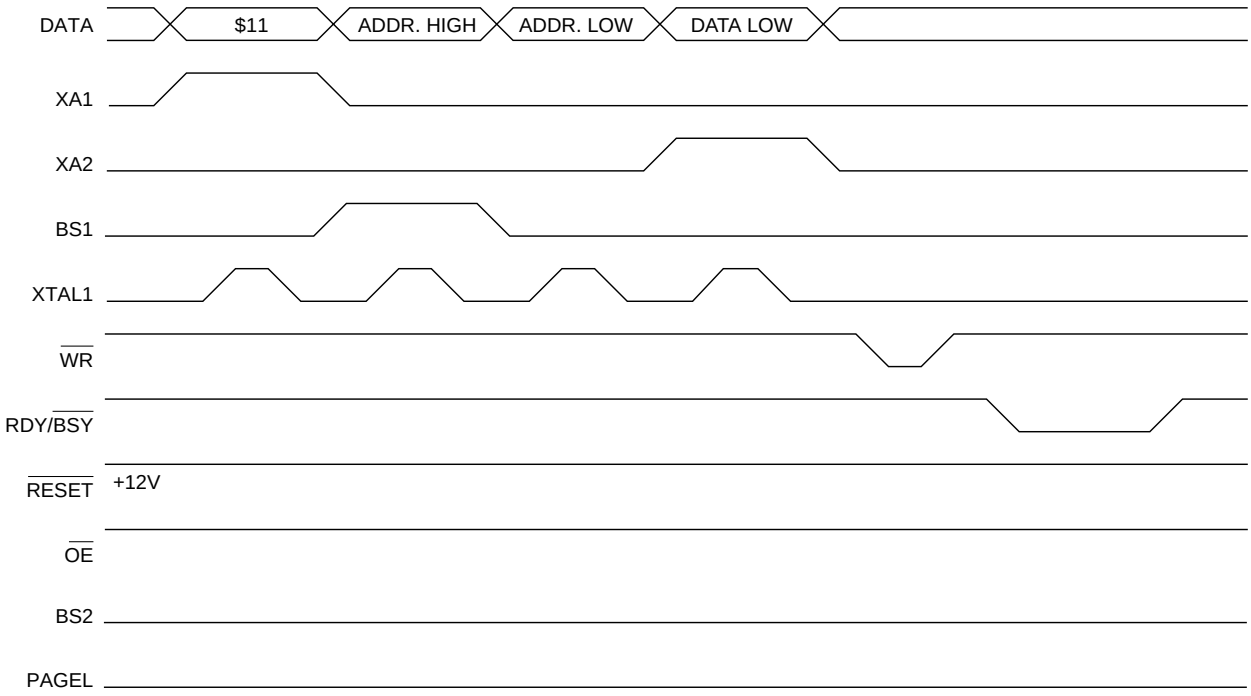
1. Set BS to ‘0’. This selects low data.
2. Give $\overline{WR}$ a negative pulse. This starts programming of the data byte. RDY/$\overline{BSY}$ goes low.
3. Wait until to RDY/$\overline{BSY}$ goes high before programming the next byte.
   (See Figure 79 for signal waveforms)

The loaded command and address are retained in the device during programming. For efficient programming, the following should be considered.

- The command needs only be loaded once when writing or reading multiple memory locations.
- Address high byte needs only be loaded before programming a new 256 word page in the EEPROM.
- Skip writing the data value $FF, that is the contents of the entire EEPROM after a Chip Erase.

These considerations also applies to Flash, EEPROM and Signature bytes reading.

Figure 79. Programming the EEPROM waveforms



Reading the Flash

The algorithm for reading the Flash memory is as follows (refer to “Programming the Flash” on page 103 for details on Command and Address loading):

1. A: Load Command ‘0000 0010’.
2. H: Load Address High Byte ($00 - $1F)
3. B: Load Address Low Byte ($00 - $FF)
4. Set $\overline{OE}$ to ‘0’, and BS1 to ‘0’. The Flash word low byte can now be read at DATA.
5. Set BS to ‘1’. The Flash word high byte can now be read at DATA.
6. Set $\overline{OE}$ to ‘1’.

Reading the EEPROM

The algorithm for reading the EEPROM memory is as follows (refer to “Programming the Flash” on page 103 for details on Command and Address loading):

1. A: Load Command ‘0000 0011’.
2. H: Load Address High Byte ($00 - $01)
3. B: Load Address ($00 - $FF)
4. Set $\overline{OE}$ to ‘0’, and BS1 to ‘0’. The EEPROM Data byte can now be read at DATA.
5. Set $\overline{OE}$ to ‘1’.

Programming the Fuse Bits

The algorithm for programming the Fuse bits is as follows (refer to “Programming the Flash” on page 103 for details on Command and Data loading):

1. A: Load Command ‘0100 0000’.
2. C: Load Data Low Byte. Bit n = ‘0’ programs and bit n = ‘1’ erases the Fuse bit.
   Bit 6 = BOOTRST Fuse bit
   Bit 5 = SPIEN Fuse bit
   Bit 4 = BODLEVEL Fuse bit
   Bit 3 = BODEN Fuse bit
   Bit 2-0 = CKSEL2..0 Fuse bits
   Bit 7 = ‘1’. This bit is reserved and should be left unprogrammed (‘1’).
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 “Programming the Flash” on page 103 for details on Command and Data loading):

1. A: Load Command ‘0010 0000’.
2. D: Load Data Low Byte. Bit n = ‘0’ programs the Lock bit.
   Bit 5 = Boot Lock Bit12
   Bit 4 = Boot Lock Bit11
   Bit 3 = Boot Lock Bit02
   Bit 2 = Boot Lock Bit01
   Bit 1 = Lock Bit2
   Bit 0 = Lock Bit1
   Bit 7-6 = ‘1’. These bits are reserved and should be left unprogrammed (‘1’).
3. L: Write Data Low Byte.

The Lock bits can only be cleared by executing Chip Erase.

Reading the Fuse And Lock Bits

The algorithm for reading the Fuse and Lock bits is as follows (refer to “Programming the Flash” on page 103 for details on Command loading):

1. A: Load Command ‘0000 0100’.
2. Set $\overline{OE}$ to ‘0’, and BS to ‘0’. The status of the Fuse bits can now be read at DATA (‘0’ means programmed).
   Bit 6 = BOOTRST Fuse bit
   Bit 5 = SPIEN Fuse bit
   Bit 4 = BODLEVEL Fuse bit
   Bit 3 = BODEN Fuse bit
   Bit 2-0 = CKSEL2..0 Fuse bits
3. Set $\overline{OE}$ to ‘0’, and BS to ‘1’. The status of the Lock bits can now be read at DATA (‘0’ means programmed).
   Bit 5 = Boot Lock Bit12
   Bit 4 = Boot Lock Bit11
   Bit 3 = Boot Lock Bit02
   Bit 2 = Boot Lock Bit01
   Bit 1 = Lock Bit2
   Bit 0 = Lock Bit1
4. Set $\overline{OE}$ to ‘1’.

Reading the Signature Bytes

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

1. A: Load Command ‘0000 1000’.
2. C: Load Address Low Byte ($00 - $02).
   Set $\overline{OE}$ to ‘0’, and BS to ‘0’. The selected Signature byte can now be read at DATA.
3. Set $\overline{OE}$ to ‘1’.

Parallel Programming Characteristics

Figure 80. Parallel Programming Timing

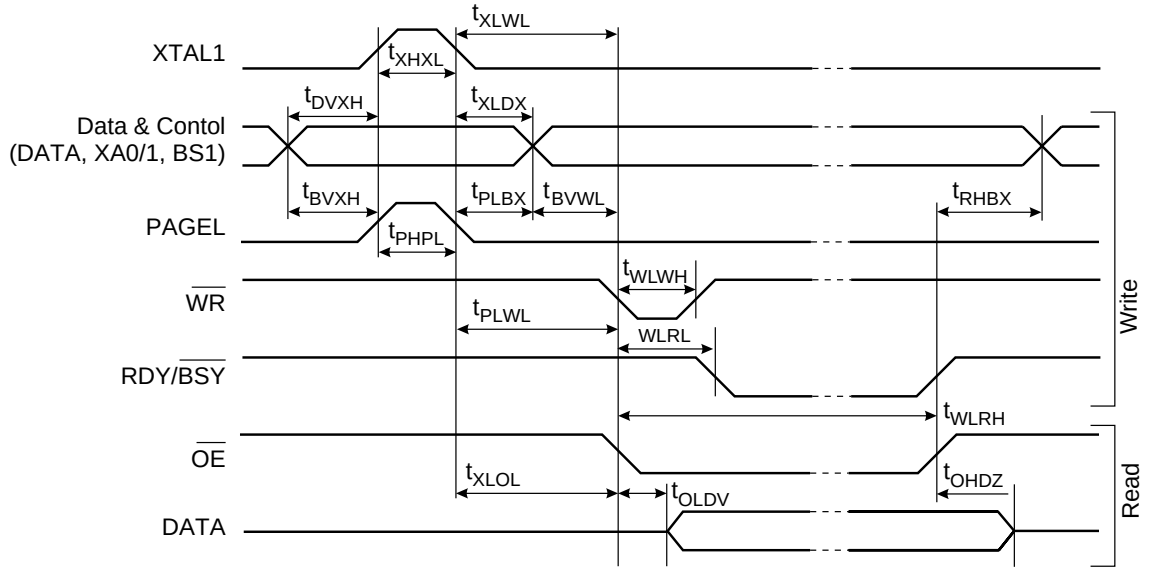


Table 44. Parallel Programming Characteristics, T_A = 25°C ± 10%, V_{CC} =5V ± 10%

Symbol	Parameter	Min	Typ	Max	Units
V_{PP}	Programming Enable Voltage	11.5		12.5	V
I_{PP}	Programming Enable Current			250	µA
t_{DVXH}	Data and Control Valid before XTAL1 High	67			ns
t_{XHXL}	XTAL1 Pulse Width High	67			ns
t_{XLDX}	Data and Control Hold after XTAL1 Low	67			ns
t_{XLWL}	XTAL1 Low to $\overline{WR}$ Low	67			ns
t_{BVXH}	BS1 Valid before XTAL1 High	67			ns
t_{PHPL}	PAGEL Pulse Width High	67			ns
t_{PLBX}	BS1 Hold after PAGEL Low	67			ns
t_{PLWL}	PAGEL Low to $\overline{WR}$ Low	67			ns
t_{BVWL}	BS1 Valid to $\overline{WR}$ Low	67			ns
t_{RHBX}	BS1 Hold after RDY/$\overline{BSY}$ High	67			ns
t_{WLWH}	$\overline{WR}$ Pulse Width Low	67			ns
t_{WLRL}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ Low	0		2.5	µs
t_{WLRH}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High(1)	0.5	0.7	0.9	ms
t_{WLRH_CE}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High for Chip Erase(2)	10	14	18	ms
t_{WLRH_FLASH}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High for Write Flash(3)	5	7	9	ms
t_{XLOL}	XTAL1 Low to $\overline{OE}$ Low	67			ns
t_{OLDV}	$\overline{OE}$ Low to DATA Valid		20		ns
t_{OHDZ}	$\overline{OE}$ High to DATA Tri-stated			20	ns

Notes: 1. t_{WLRH} is valid for the Write EEPROM, Write Fuse Bits and Write Lock Bits commands.
2. t_{WLRH_CE} is valid for the Chip Erase command.
3. t_{WLRH_FLASH} is valid for the Write Flash command.

Serial Downloading

Both the Flash and EEPROM memory arrays can be programmed using the serial SPI bus while $\overline{RESET}$ is pulled to GND. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After $\overline{RESET}$ 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:

$0000 to $1FFF for Program memory and $0000 to $01FF 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 cycle

High:> 2 XTAL1 clock cycles

Serial Programming Algorithm

When writing serial data to the ATmega161, data is clocked on the rising edge of SCK.

When reading data from the ATmega161, data is clocked on the falling edge of SCK. See Figure 81, Figure 82 and Table 47 for timing details.

To program and verify the ATmega161 in the serial programming mode, the following sequence is recommended (See four byte instruction formats in Table 46):

1. Power-up sequence:

   Apply power between V_{CC} 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 MOSI/PB5.
3. The serial programming instructions will not work if the communication is out of synchronization. When in sync. the second byte ($53), will echo back when issuing the third byte of the Programming Enable instruction. Whether the echo is correct or not, all 4 bytes of the instruction must be transmitted. If the $53 did not echo back, give $\overline{\text{RESET}}$ a positive pulse and issue a new Programming Enable command.
4. If a chip erase is performed (must be done to erase the Flash), 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 6 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 7 MSB of the address. If polling is not used, the user must wait at least t_{WD_FLASH} before issuing the next page. (Please refer to Table 45). Accessing the serial programming interface before the Flash write operation completes can result in incorrect programming.
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. If polling is not used, the user must wait at least t_{WD_EEPROM} before issuing the next byte. (Please refer to Table 45). In a chip erased device, no $FFs in the data file(s) need to be programmed.
7. Any memory location can be verified by using the Read instruction which returns the content at the selected address at serial output MISO/PB6.
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

Data Polling FLASH

When a page is being programmed into the Flash, reading an address location within the page being programmed will give the value $FF. At the time the device is ready for a new page, the programmed value will read correctly. This is used to determine when the next page can be written. Note that the entire page is written simultaneously and any address within the page can be used for polling. Data polling of the FLASH will not work for the value $FF, so when programming this value, the user will have to wait for at least t_{WD_FLASH} before programming the next page. As a chip-erased device contains $FF in all locations, programming of addresses that are meant to contain $FF, can be skipped. See Table 45 for t_{WD_FLASH} value.

Data Polling EEPROM

When a new byte has been written and is being programmed into EEPROM, reading the address location being programmed will give the value $FF. At the time the device is ready for a new byte, the programmed value will read correctly. This is used to determine when the next byte can be written. This will not work for the value $FF, but the user should have the following in mind: As a chip-erased device contains $FF in all locations, programming of addresses that are meant to contain $FF, can be skipped. This does not apply if the EEPROM is re-programmed without chip-erasing the device. In this case, data polling cannot be used for the value $FF, and the user will have to wait at least t_{WD_EEPROM} before programming the next byte. See Table 45 for t_{WD_EEPROM} value.

Table 45. Minimum wait delay before writing the next Flash or EEPROM location

Symbol	3.2V	3.6V	4.0V	5.0V
t_{WD_FLASH}	28 ms	22 ms	18 ms	11 ms
t_{WD_EEPROM}	9 ms	7 ms	6 ms	4 ms

Figure 81. Serial Programming Waveforms

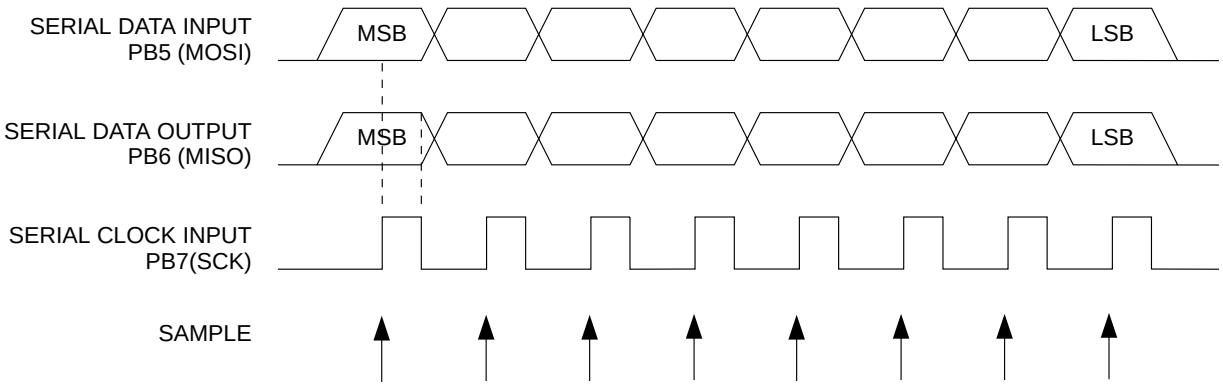


Table 46. Serial Programming Instruction Set

Instruction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte4	
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	xxxa **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	xx**bb bbbb**	**iiii iiii**	Write **H** (high or low) data **i** to Program Memory page at word address **b**.
Write Program Memory Page	0100 1100	xxxa **aaaa**	**bb**xx xxxx	iiii iiii	Write Program Memory Page at address **a**:**b**.
Read EEPROM Memory	1010 0000	xxxx xxx**a**	**bbbb bbbb**	**oooo oooo**	Read data **o** from EEPROM memory at address **a**:**b**.
Write EEPROM Memory	1100 0000	xxxx xxx**a**	**bbbb bbbb**	**iiii iiii**	Write data **i** to EEPROM memory at address **a**:**b**.
Read Lock Bits	0101 1000	xxxx xxxx	xxxx xxxx	xx***65 4321***	Read Lock bits. ‘0’ = programmed, ‘1’ = unprogrammed.
Write Lock Bits	1010 1100	111x xxxx	xxxx xxxx	11***65 4321***	Write Lock bits. Set bits ***6*** - ***1*** = ’0’ to program Lock bits.
Read Signature Byte	0011 0000	xxxx xxxx	xxxx xx**bb**	**oooo oooo**	Read Signature Byte **o** at address **b**.
Write Fuse Bits	1010 1100	101x xxxx	xxxx xxxx	1**DCB A987**	Set bits **D** - **A**, **9** - **7** = ’0’ to program, ‘1’ to unprogram
Read Fuse Bits	1010 0000	xxxx xxxx	xxxx xxxx	x**DCB A987**	Read Fuse bits. ’0’ = programmed, ‘1’ = unprogrammed

Note: **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 = Boot Lock Bit01
4 = Boot Lock Bit02
5 = Boot Lock Bit11
6 = Boot Lock Bit12
7 = CKSEL0 Fuse
8 = CKSEL1 Fuse
9 = CKSEL2 Fuse
A = BODEN Fuse
B = BODLEVEL Fuse
C = SPIEN Fuse
D = BOOTRST Fuse

Serial Programming Characteristics

Figure 82. Serial Programming Timing

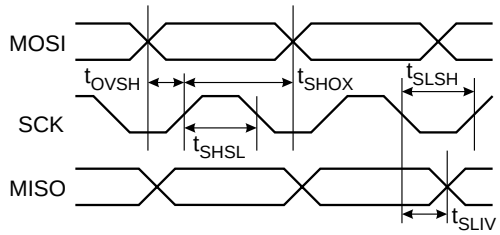


Table 47. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 2.7 - 5.5V (Unless otherwise noted)

Symbol	Parameter	Min	Typ	Max	Units
$1/t_{CLCL}$	Oscillator Frequency (V_{CC} = 2.7 - 5.5V)	0		4	MHz
t_{CLCL}	Oscillator Period (V_{CC} = 2.7 - 5.5V)	250			ns
$1/t_{CLCL}$	Oscillator Frequency (V_{CC} = 4.0 - 5.5V)	0		8	MHz
t_{CLCL}	Oscillator Period (V_{CC} = 4.0 - 5.5V)	125			ns
t_{SHSL}	SCK Pulse Width High	2 t_{CLCL}			ns
t_{SLSH}	SCK Pulse Width Low	2 t_{CLCL}			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2 t_{CLCL}			ns
t_{SLIV}	SCK Low to MISO Valid	10	16	32	ns

Electrical Characteristics

Absolute Maximum Ratings*

Operating Temperature ..... -55°C to +125°C

Storage Temperature ..... -65°C to +150°C

Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground .....-1.0V to V_{CC}+0.5V

Voltage on $\overline{\text{RESET}}$ with respect to Ground.....-1.0V to +13.0V

Maximum Operating Voltage 6.6V

DC Current per I/O Pin ..... 40.0 mA

DC Current V_{CC} and GND Pins..... 200.0 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage	(Except XTAL1)	-0.5		0.3 V_{CC}[1]	V
V_{IL1}	Input Low Voltage	(XTAL1)	-0.5		0.2 V_{CC}[1]	V
V_{IH}	Input High Voltage	(Except XTAL1, $\overline{RESET}$)	0.6 V_{CC}[2]		V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	(XTAL1)	0.8 V_{CC}[2]		V_{CC} + 0.5	V
V_{IH2}	Input High Voltage	($\overline{RESET}$)	0.9 V_{CC}[2]		V_{CC} + 0.5	V
V_{OL}	Output Low Voltage[3] (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[4] (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	Vcc = 5.5V, pin low (absolute value)			8.0	µA
I_{IH}	Input Leakage Current I/O pin	Vcc = 5.5V, pin high (absolute value)			980	nA
RRST	Reset Pull-up Resistor		100		500	kΩ
$R_{I/O}$	I/O Pin Pull-up Resistor		35		120	kΩ
I_{CC}	Power Supply Current	Active Mode, V_{CC} = 3V, 4MHz			3.0	mA
		Idle Mode V_{CC} = 3V, 4MHz			1.2	mA
	Power-down Mode[5]	WDT enabled, V_{CC} = 3V		9	15.0	µA
		WDT disabled, V_{CC} = 3V		<1	2.0	µA
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V			40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	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. "Max" means the highest value where the pin is guaranteed to be read as low
2. "Min" means the lowest value where the pin is guaranteed to be read as high
3. Although each I/O port can sink more than the test conditions (20 mA at Vcc = 5V, 10 mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOL, for all ports, should not exceed 200 mA.
   2] The sum of all IOL, for ports B0 - B7, D0 - D7 and XTAL2, should not exceed 100 mA.
   3] The sum of all IOL, for ports A0 - A7, ALE, OC1B and C0 - C7 should not exceed 100 mA.
   If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (3 mA at Vcc = 5V, 1.5 mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOH, for all ports, should not exceed 200 mA.
   2] The sum of all IOH, for ports B0 - B7, D0 - D7 and XTAL2, should not exceed 100 mA.
   3] The sum of all IOH, for ports A0 - A7, ALE, OC1B and C0 - C7 should not exceed 100 mA.
   If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.
5. Minimum V_{CC} for Power-down is 2V.

External Clock Drive Waveforms

Figure 83. External Clock

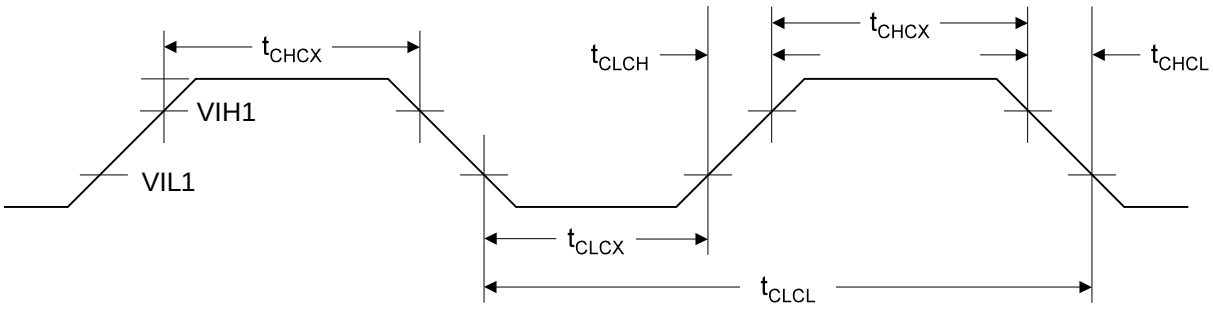


Table 48. External Clock Drive

Symbol	Parameter	V_{CC} = 2.7V to 5.5V		V_{CC} = 4.0V to 5.5V		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	4	0	8	MHz
t_{CLCL}	Clock Period	250		125		ns
t_{CHCX}	High Time	100		50		ns
t_{CLCX}	Low Time	100		50		ns
t_{CLCH}	Rise Time		1.6		0.5	µs
t_{CHCL}	Fall Time		1.6		0.5	µs

Note: See “External Data Memory Timing” on page 116. for a description of how the duty cycle influences the timing for the External Data Memory

External Data Memory Timing

Table 49. External Data Memory Characteristics, 4.0 - 5.5V, 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	TBD		t_{CLCL} - TBD		ns
2	t_{AVLL}	Address Valid A to ALE Low	TBD		$0.5t_{CLCL}$ - TBD(1)		ns
3a	t_{LLAX_ST}	Address Hold After ALE Low, ST/STD/STS Instructions	TBD		TBD		ns
3b	t_{LLAX_LD}	Address Hold after ALE Low, LD/LDD/LDS Instructions	TBD		TBD		ns
4	t_{AVLLC}	Address Valid C to ALE Low	TBD		$0.5t_{CLCL}$ - TBD(1)		ns
5	t_{AVRL}	Address Valid to RD Low	TBD		$1.0t_{CLCL}$ - TBD		ns
6	t_{AVWL}	Address Valid to WR Low	TBD		$1.0t_{CLCL}$ - TBD		ns
7	t_{LLWL}	ALE Low to WR Low	TBD	TBD	$0.5t_{CLCL}$ - TBD(2)	$0.5t_{CLCL}$ - TBD(2)	ns
8	t_{LLRL}	ALE Low to RD Low	TBD	TBD	$0.5t_{CLCL}$ - TBD(2)	$0.5t_{CLCL}$ - TBD(2)	ns
9	t_{DVRH}	Data Setup to RD High	TBD		TBD		ns
10	t_{RLDV}	Read Low to Data Valid		TBD		TBD	ns
11	t_{RHDX}	Data Hold After RD High	TBD		TBD		ns
12	t_{RLRH}	RD Pulse Width	TBD		$1.0t_{CLCL}$ - TBD		ns
13	t_{DVWL}	Data Setup to WR Low	TBD		$0.5t_{CLCL}$ - TBD(1)		ns
14	t_{WHDX}	Data Hold After WR High	TBD		$0.5t_{CLCL}$ - TBD(1)		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$1.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$1.0t_{CLCL}$ - TBD		ns

Notes: 1. This assumes 50% clock duty cycle. The half period is actually the high time of the external clock, XTAL1.
2. This assumes 50% clock duty cycle. The half period is actually the low time of the external clock, XTAL1.

Table 50. External Data Memory Characteristics, 4.0 - 5.5V, 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		TBD		$2.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$2.0t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$2.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$2.0t_{CLCL}$ - TBD		ns

Table 51. External Data Memory Characteristics, 4.0 - 5.5V, SRWn1 = 1, SRWn0 = 0

			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		TBD		$3.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$3.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns

Table 52. External Data Memory Characteristics, 4.0 - 5.5V, SRWn1 = 1, SRWn0 = 1

			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		TBD		$3.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns
14	t_{WHDX}	Data Hold After WR High	TBD		$1.5t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$3.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns

Table 53. External Data Memory Characteristics, 2.7 - 5.5V, No Wait State

			4 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	TBD		t_{CLCL} - TBD		ns
2	t_{AVLL}	Address Valid A to ALE Low	TBD		$0.5t_{CLCL}$ - TBD		ns
3a	t_{LLAX_ST}	Address Hold After ALE Low, ST/STD/STS Instructions	TBD		TBD		ns
3b	t_{LLAX_LD}	Address Hold after ALE Low, LD/LDD/LDS Instructions	TBD		TBD		ns
4	t_{AVLLC}	Address Valid C to ALE Low	TBD		$0.5t_{CLCL}$ - TBD		ns
5	t_{AVRL}	Address Valid to RD Low	TBD		$1.0t_{CLCL}$ - TBD		ns
6	t_{AVWL}	Address Valid to WR Low	TBD		$1.0t_{CLCL}$ - TBD		ns
7	t_{LLWL}	ALE Low to WR Low	TBD	TBD	$0.5t_{CLCL}$ - TBD	$0.5t_{CLCL}$ - TBD	ns
8	t_{LLRL}	ALE Low to RD Low	TBD	TBD	$0.5t_{CLCL}$ - TBD	$0.5t_{CLCL}$ - TBD	ns
9	t_{DVRH}	Data Setup to RD High	TBD		TBD		ns
10	t_{RLDV}	Read Low to Data Valid		TBD		TBD	ns
11	t_{RHDX}	Data Hold After RD High	TBD		TBD		ns

Table 53. External Data Memory Characteristics, 2.7 - 5.5V, No Wait State (Continued)

			4 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
12	t_{RLRH}	RD Pulse Width	TBD		$1.0t_{CLCL}$ - TBD		ns
13	t_{DVWL}	Data Setup to WR Low	TBD		$0.5t_{CLCL}$ - TBD		ns
14	t_{WHDX}	Data Hold After WR High	TBD		$0.5t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$1.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$1.0t_{CLCL}$ - TBD		ns

Notes: 1. This assumes 50% clock duty cycle. The half period is actually the high time of the external clock, XTAL1.
2. This assumes 50% clock duty cycle. The half period is actually the low time of the external clock, XTAL1.

Table 54. External Data Memory Characteristics, 2.7 - 5.5V, SRWn1 = 0, SRWn0 = 1

			4 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	4.0	MHz
10	t_{RLDV}	Read Low to Data Valid		TBD		$2.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$2.0t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$2.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$2.0t_{CLCL}$ - TBD		ns

Table 55. External Data Memory Characteristics, 2.7 - 5.5V, SRWn1 = 1, SRWn0 = 0

			4 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	4.0	MHz
10	t_{RLDV}	Read Low to Data Valid		TBD		$3.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$3.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns

Table 56. External Data Memory Characteristics, 2.7 - 5.5V, SRWn1 = 1, SRWn0 = 1

			4 MHz Oscillator		Variable Oscillator		
	Symbol	**Parameter**	**Min**	**Max**	**Min**	**Max**	**Unit**
0	$1/t_{CLCL}$	Oscillator Frequency			0.0	4.0	MHz
10	t_{RLDV}	Read Low to Data Valid		TBD		$3.0t_{CLCL}$ - TBD	ns
12	t_{RLRH}	RD Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns
14	t_{WHDX}	Data Hold After WR High	TBD		$1.5t_{CLCL}$ - TBD		ns
15	t_{DVWH}	Data Valid to WR High	TBD		$3.0t_{CLCL}$ - TBD		ns
16	t_{WLWH}	WR Pulse Width	TBD		$3.0t_{CLCL}$ - TBD		ns

Figure 84. External Memory Timing (SRWn1 = 0, SRWn0 = 0

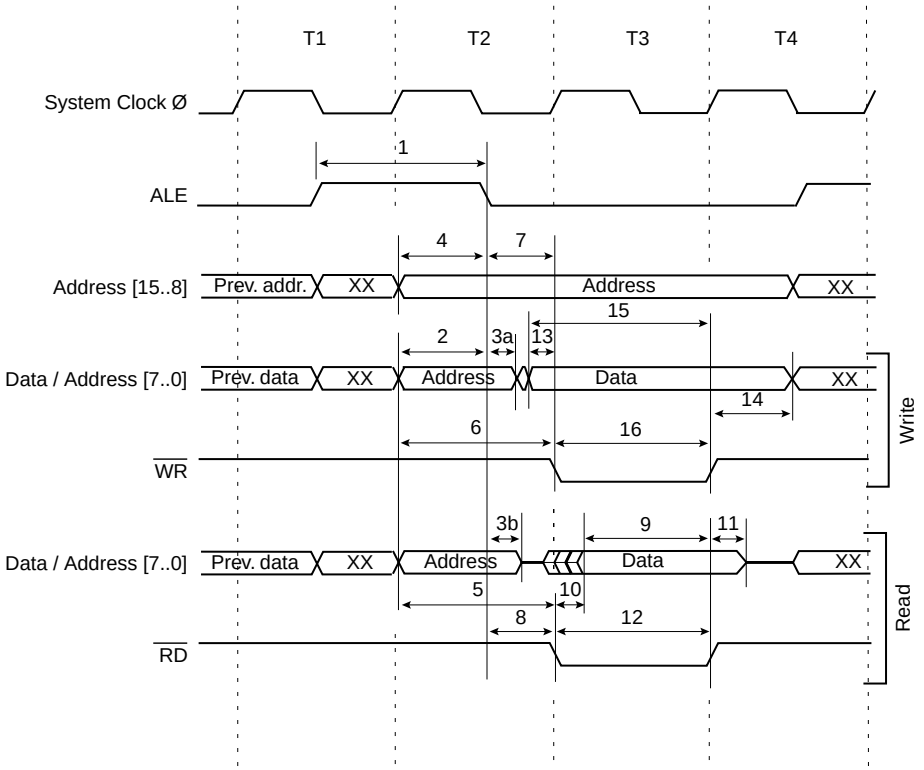


Figure 85. External Memory Timing (SRWn1 = 0, SRWn0 = 1)

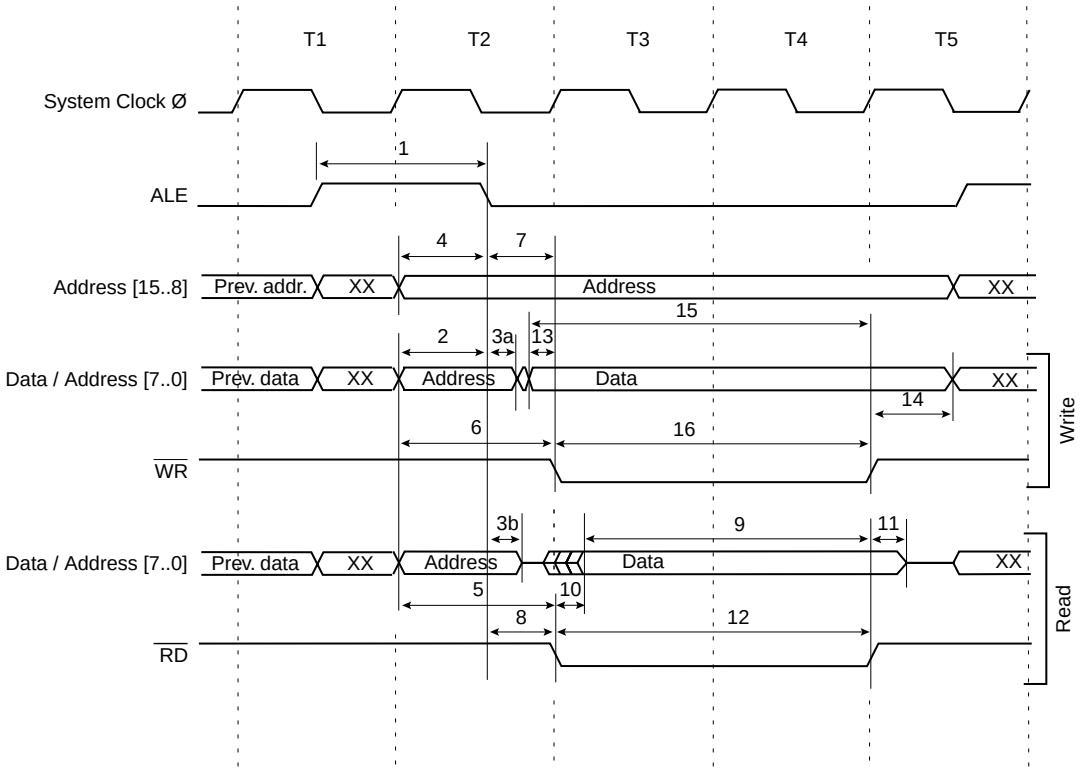


Figure 86. External Memory Timing (SRWn1 = 1, SRWn0 = 0)

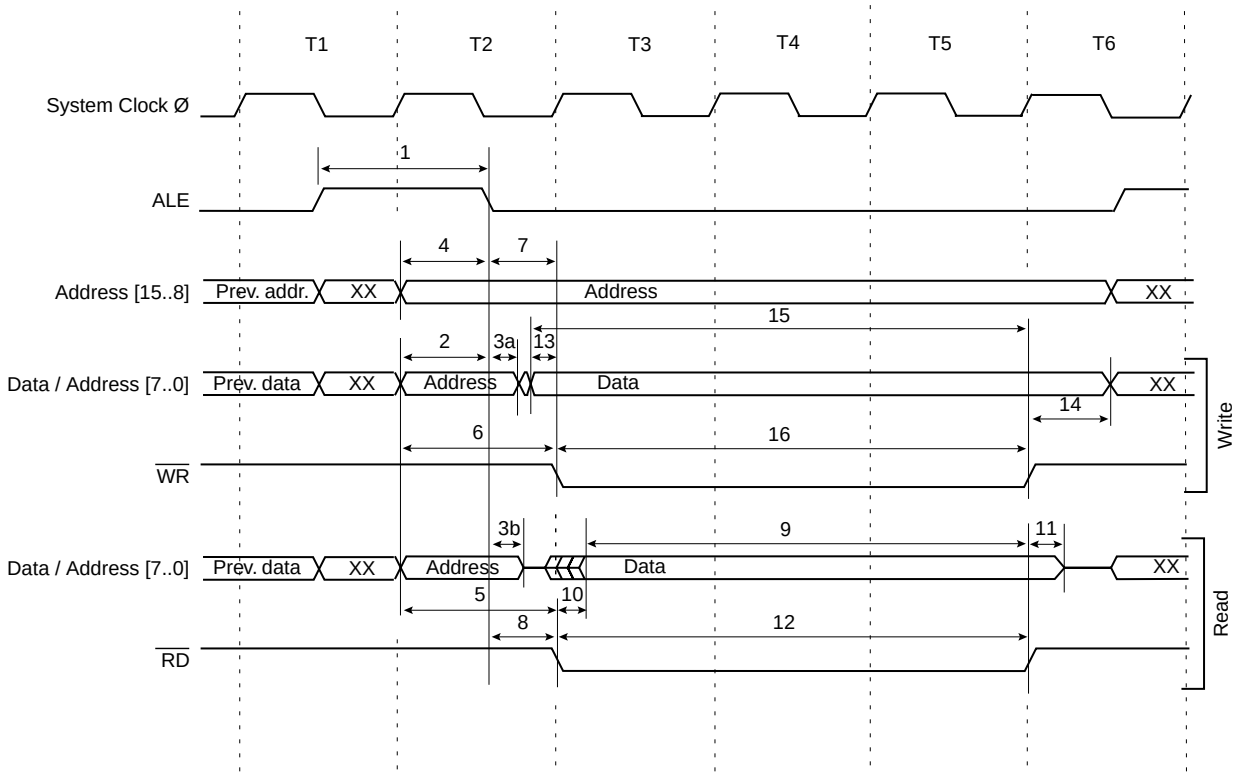
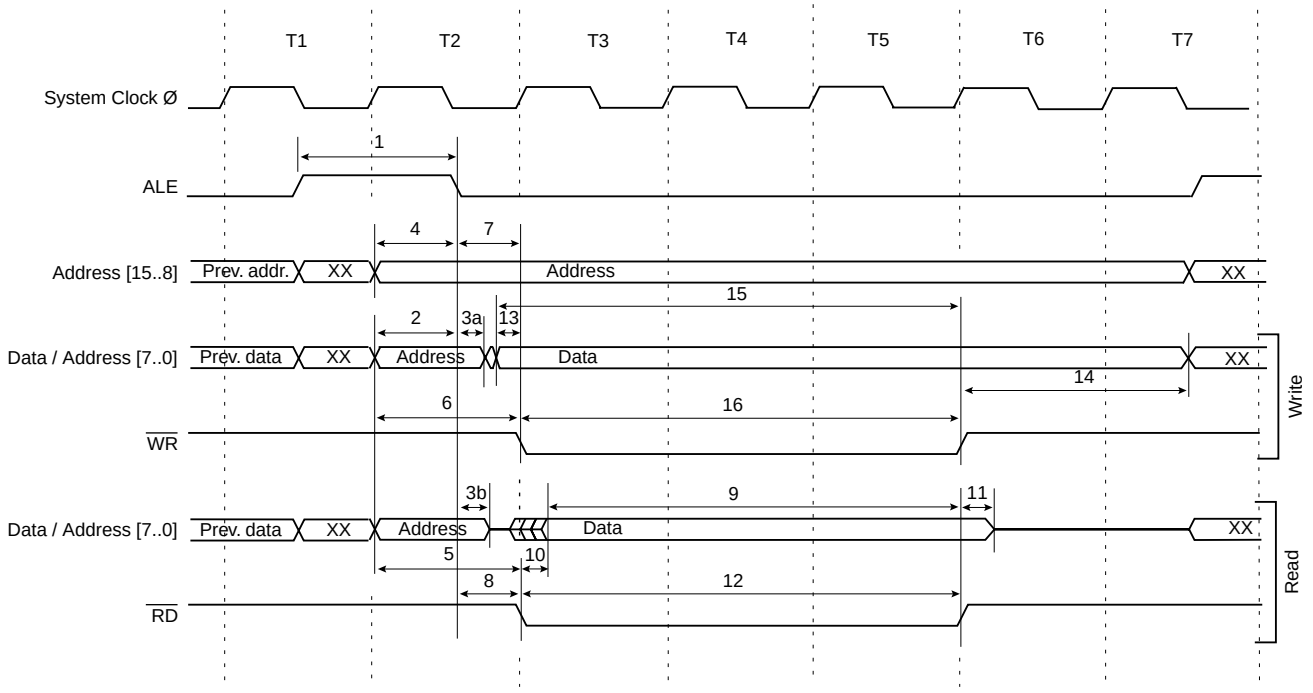


Figure 87. External Memory Timing (SRWn1 = 1, SRWn0 = 1)



Note: The ALE pulse in the last period (T4 - T7) is only present if the next instruction accesses the RAM (internal or external). The Data and Address will only change in T4 - T7 if ALE is present (the next instruction accesses the RAM).

Typical Characteristics – Preliminary Data

Analog comparator offset voltage is measured as absolute offset

Figure 88. Analog Comparator Offset Voltage vs, Common Mode Voltage

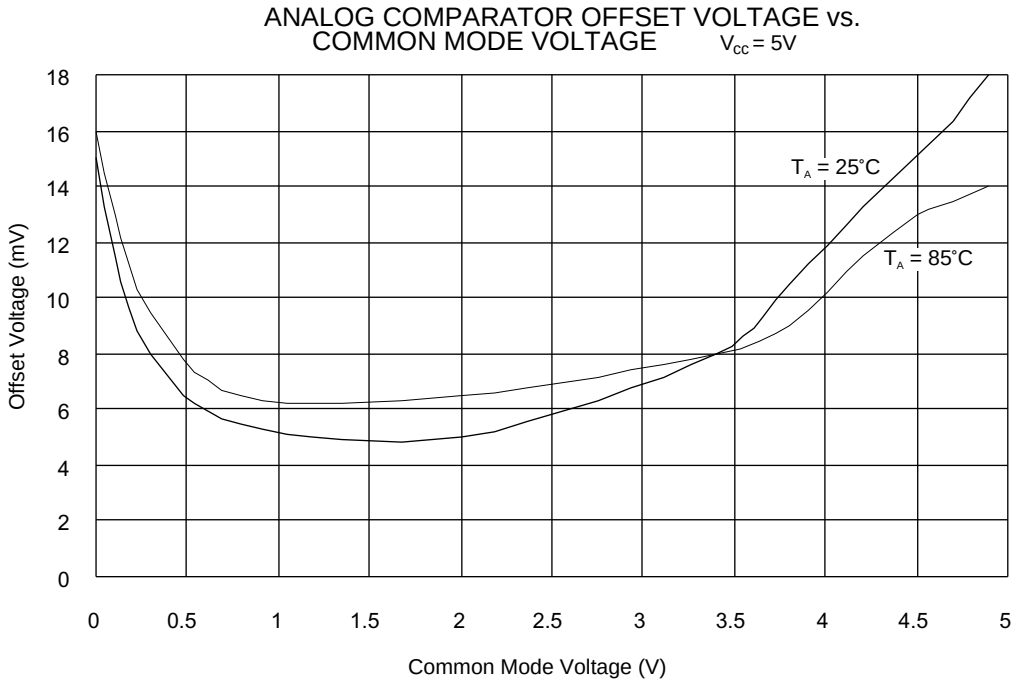


Figure 89. Analog Comparator Offset Voltage vs. Common Mode Voltage

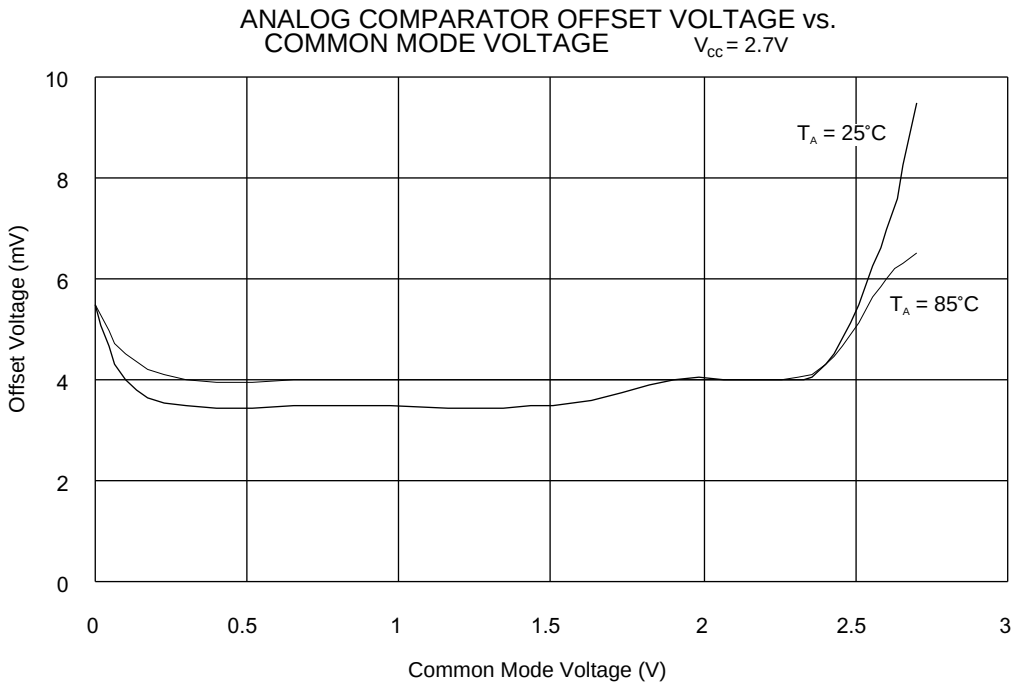


Figure 90. Analog Comparator Input Leakage Current

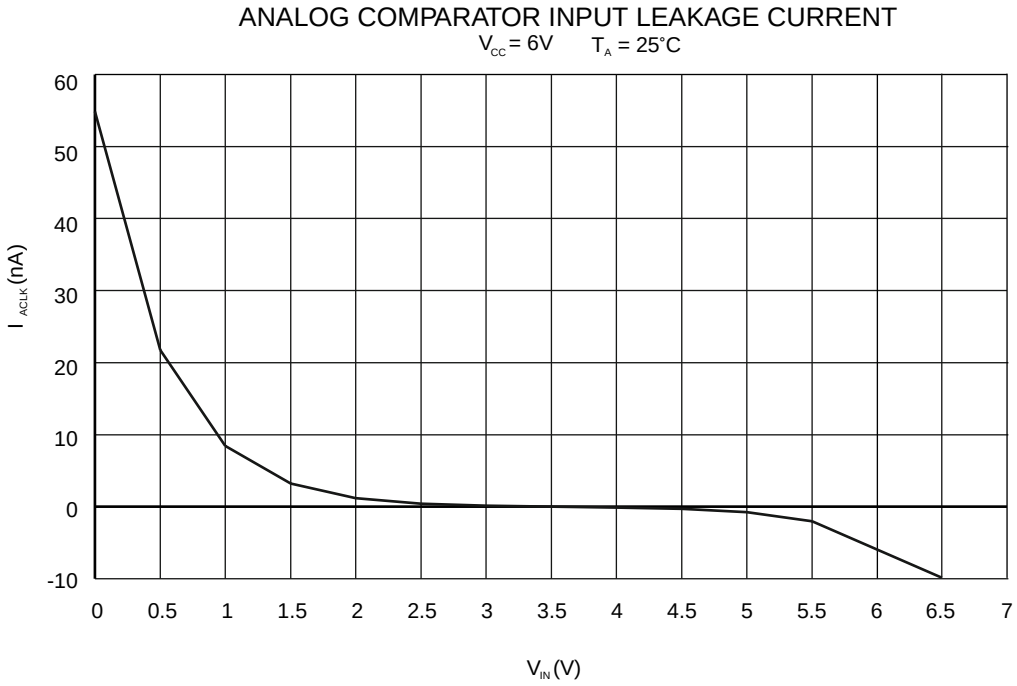
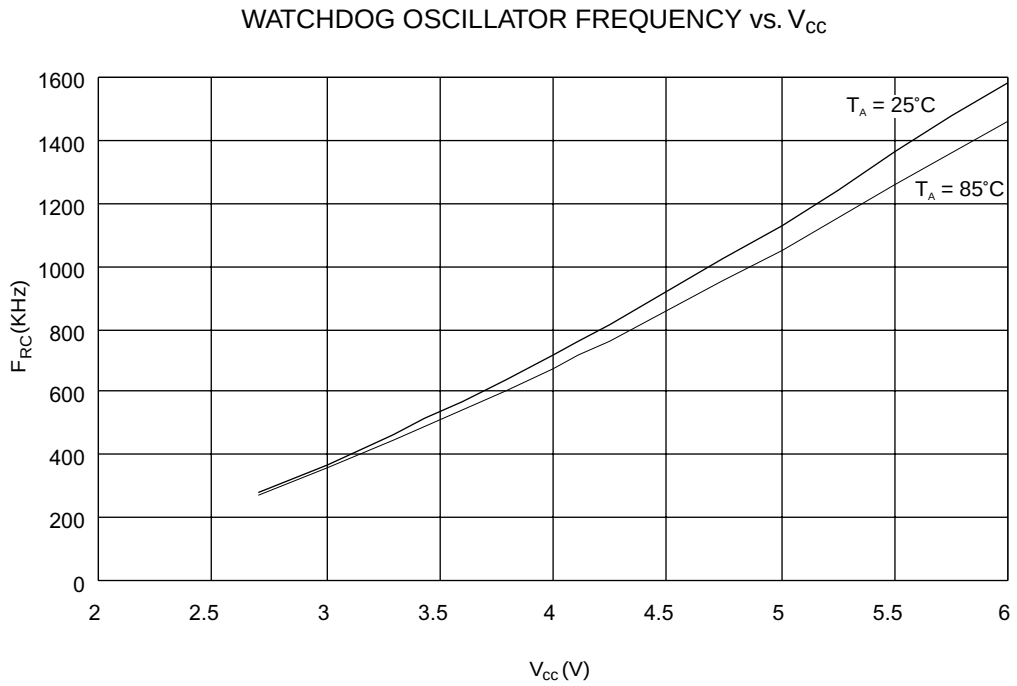


Figure 91. Watchdog Oscillator Frequency vs. V_{CC}



Sink and source capabilities of I/O ports are measured on one pin at a time.

Figure 92. Pull-up Resistor Current vs. Input Voltage

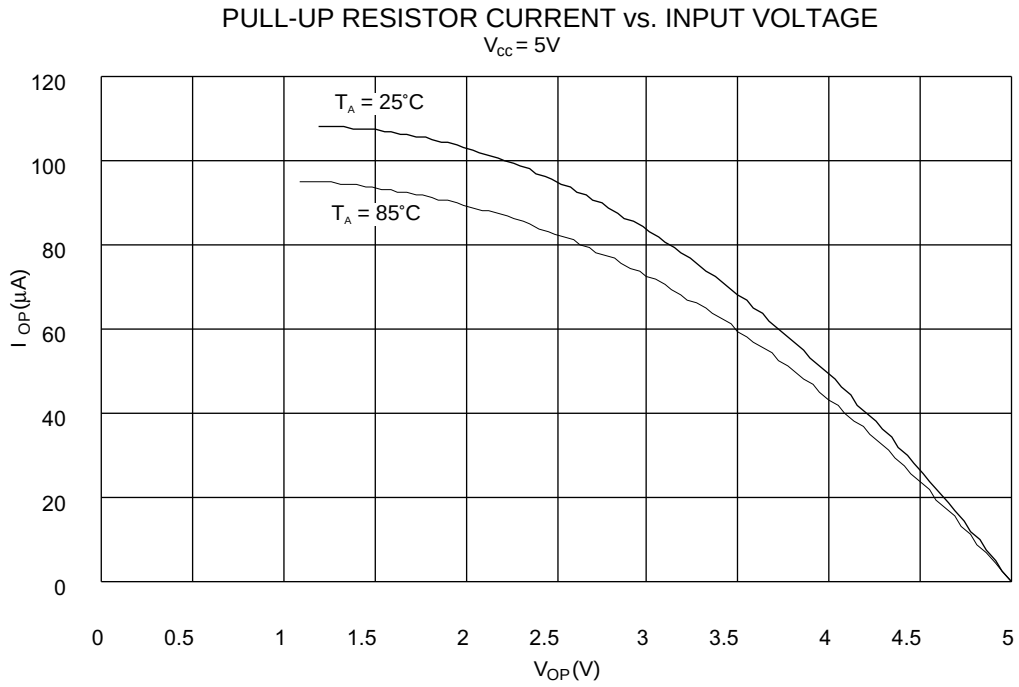


Figure 93. Pull-up Resistor Current vs. Input Voltage

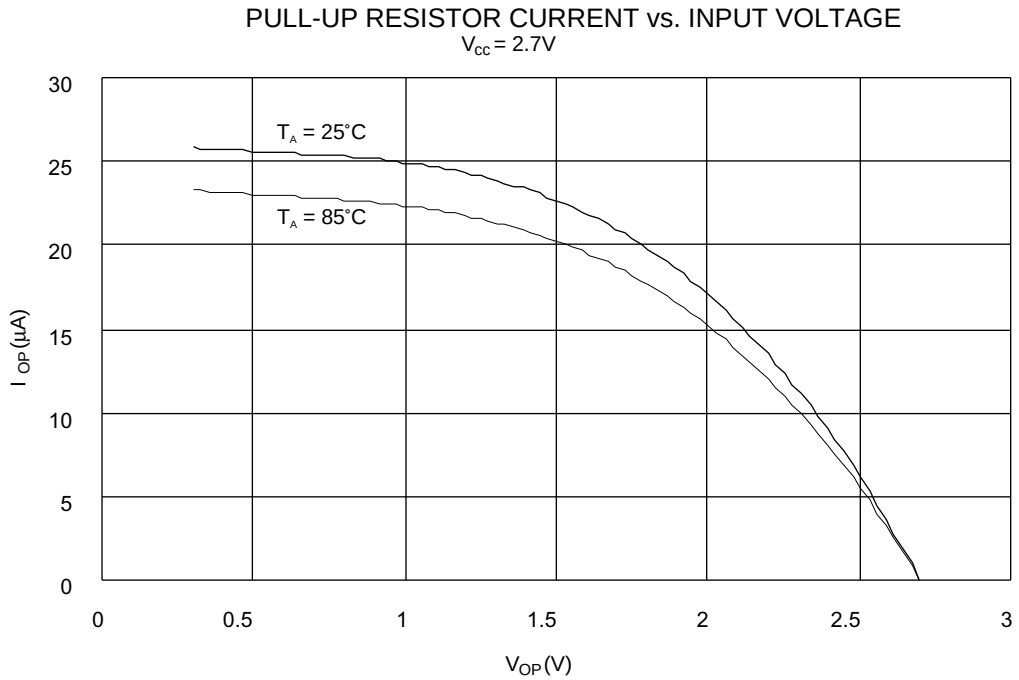


Figure 94. I/O Pin Sink Current vs. Output Voltage

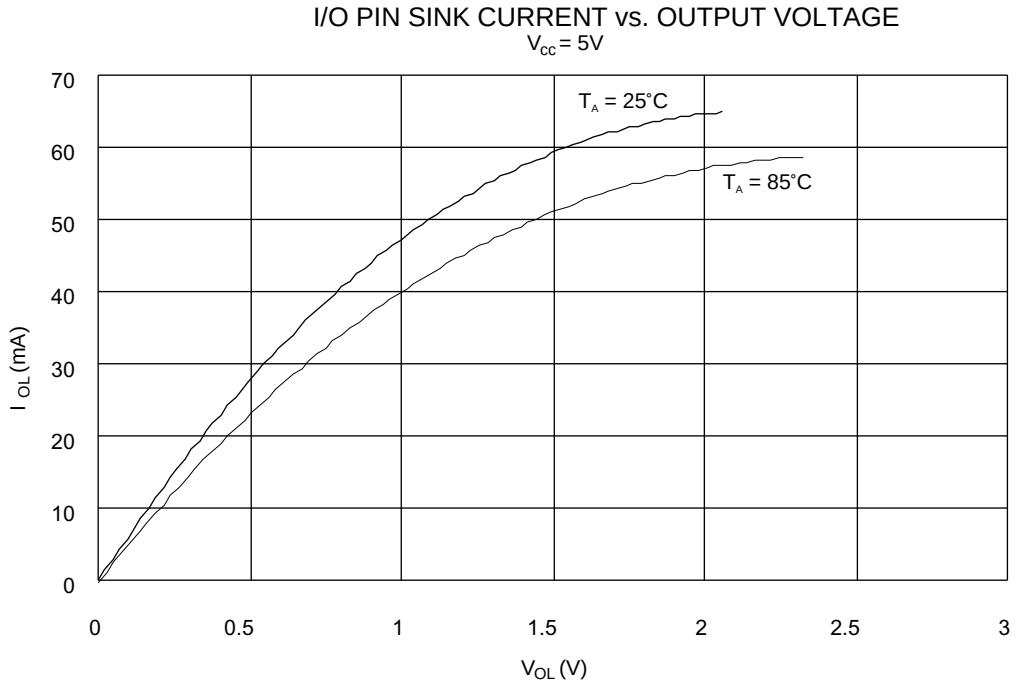


Figure 95. I/O Pin Source Current vs. Output Voltage

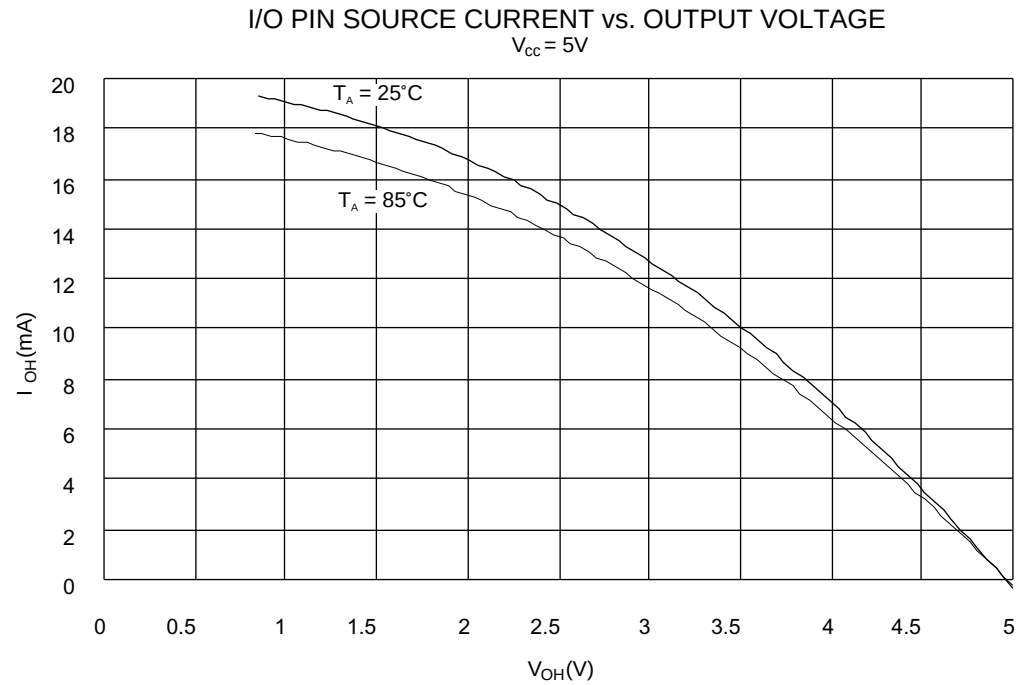


Figure 96. I/O Pin Sink Current vs. Output Voltage

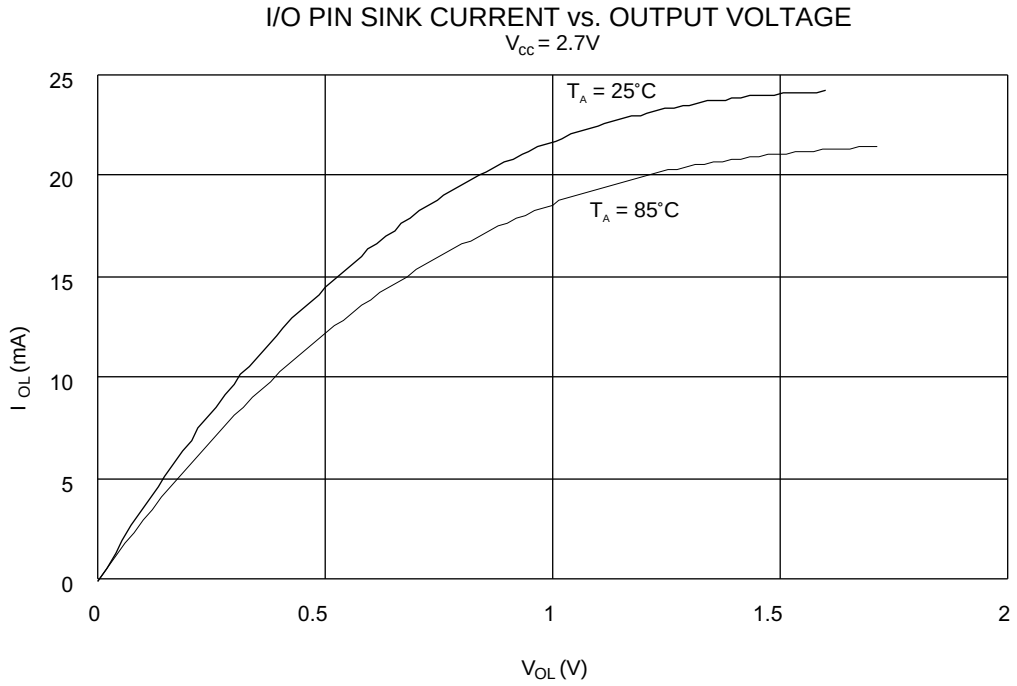


Figure 97. I/O Pin Source Current vs. Output Voltage

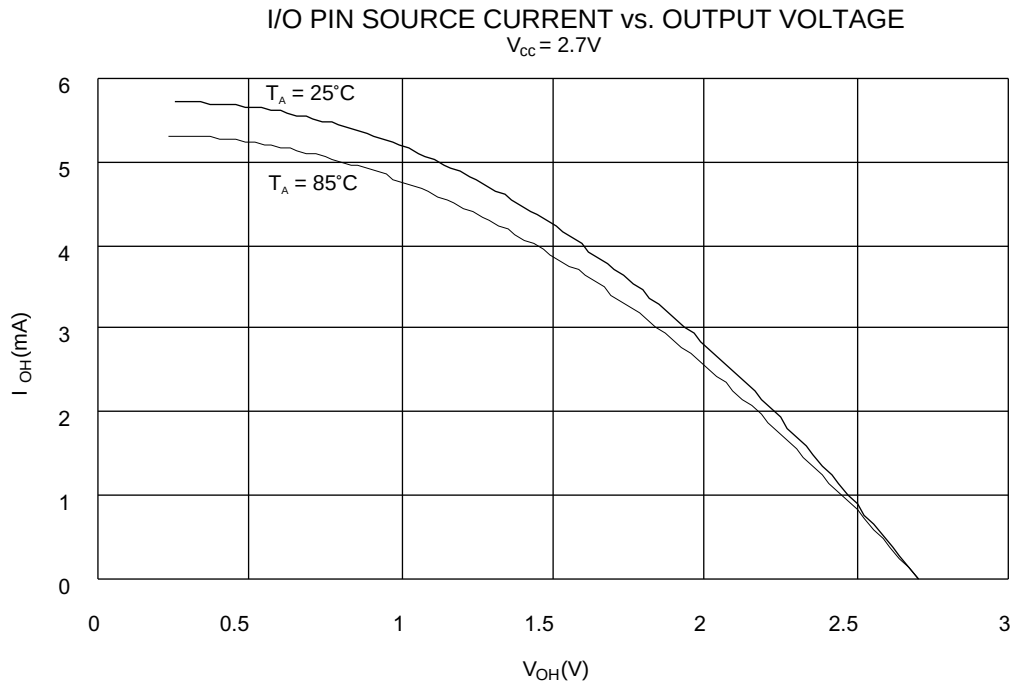


Figure 98. I/O Pin Input Threshols vs. V_{CC}

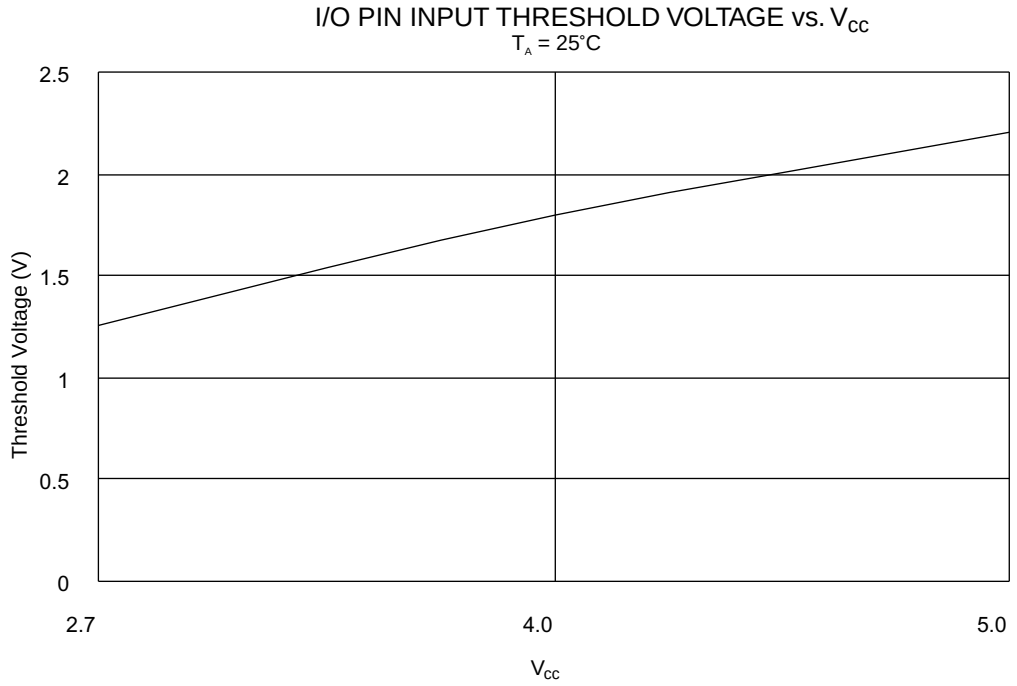
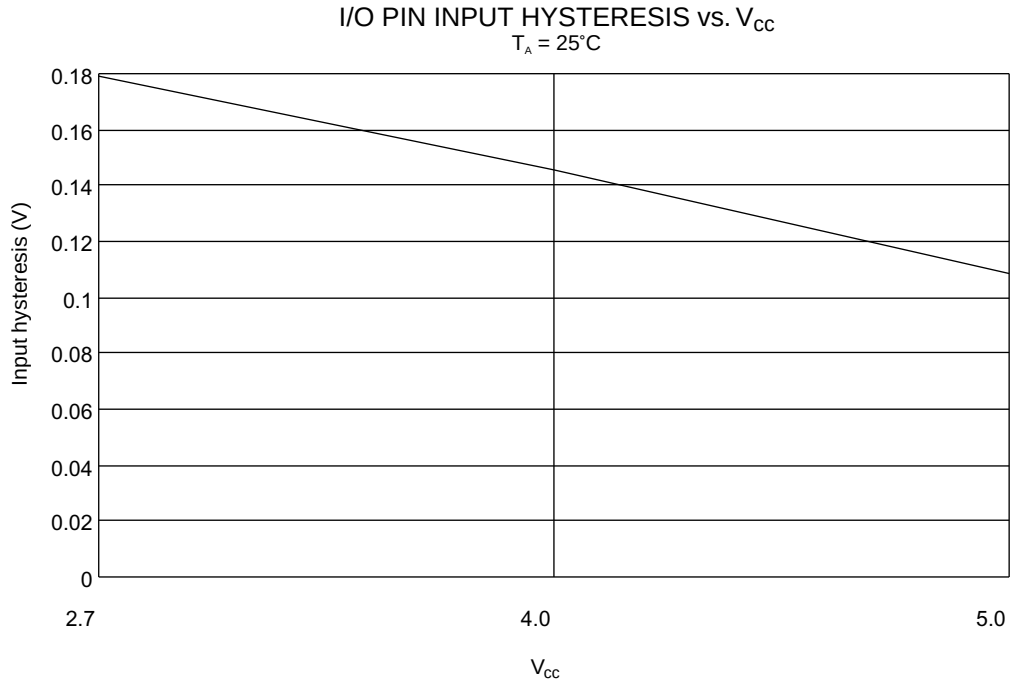


Figure 99. I/O Pin Input Hysteresis vs. V_{CC}



Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	21
$3E ($5E)	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	22
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	22
$3C ($5C)	Reserved									
$3B ($5B)	GIMSK	INT1	INT0	INT2	-	-	-	-	-	29
$3A ($5A)	GIFR	INTF1	INTF0	INTF2						30
$39 ($59)	TIMSK	TOIE1	OCIE1A	OCIE1B	OCIE2	TICIE1	TOIE2	TOIE0	OCIE0	30
$38 ($58)	TIFR	TOV1	OCF1A	OCF1B	OCFI2	ICF1	TOV2	TOV0	OCIF0	31
$37 ($57)	SPMCR	-	-	-	-	LBSET	PGWRT	PGERS	SPMEN	98
$36 ($56)	EMCUCR	SM0	SRL2	SRL1	SRL0	SRW01	SRW00	SRW11	ISC2	33
$35 ($55)	MCUCR	SRE	SRW10	SE	SM1	ISC11	ISC10	ISC01	ISC00	32
$34 ($54)	MCUSR	-	-	-	-	WDRF	BORF	EXTRF	PORF	28
$33 ($53)	TCCR0	FOC0	PWM0	COM01	COM00	CTC0	CS02	CS01	CS00	38
$32 ($52)	TCNT0	Timer/Counter0 Counter Register								40
$31 ($51)	OCR0	Timer/Counter0 Output Compare Register								40
$30 ($50)	SFIOR	-	-	-	-	-	-	PSR2	PSR10	36
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	PWM11	PWM10	46
$2E ($4E)	TCCR1B	ICNC1	ICES1	-	-	CTC1	CS12	CS11	CS10	47
$2D ($4D)	TCNT1H	Timer/Counter1 - Counter Register High Byte								48
$2C ($4C)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								48
$2B ($4B)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								49
$2A ($4A)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								49
$29 ($49)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								49
$28 ($48)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								49
$27 ($47)	TCCR2	FOC2	PWM2	COM21	COM20	CTC2	CS22	CS21	CS20	38
$26 ($46)	ASSR	-	-	-	-	AS20	TCON2UB	OCR2UB	TCR2UB	43
$25 ($45)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								49
$24 ($44)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								49
$23 ($43)	TCNT2	Timer/Counter2 Counter Register								40
$22 ($42)	OCR2	Timer/Counter2 Output Compare Register								40
$21 ($41)	WDTCR	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0	53
$20 ($40)	UBRRHI	UBRR1[11:8]				UBRR0[11:8]				67
$1F ($3F)	EEARH	-	-	-	-	-	-	-	EEAR8	54
$1E ($3E)	EEARL	EEPROM Address Register Low Byte								54
$1D ($3D)	EEDR	EEPROM Data Register								54
$1C ($3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE	54
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	77
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	77
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	77
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	79
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	79
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	79
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	85
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	85
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	85
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	87
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	87
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	87
$0F ($2F)	SPDR	SPI Data Register								60
$0E ($2E)	SPSR	SPIF	WCOL	-	-	-	-	-	SPI2X	59
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	59
$0C ($2C)	UDR0	UART0 I/O Data Register								64
$0B ($2B)	UCSR0A	RXC0	TXC0	UDRE0	FE0	OR0	-	U2X0	MPCM0	64
$0A ($2A)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	CHR90	RXB80	TXB80	65
$09 ($29)	UBRR0	UART0 Baud Rate Register								68
$08 ($28)	ACSR	ACD	AINBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	70
$07 ($27)	PORTE	-	-	-	-	-	PORTE2	PORTE1	PORTE0	92

Register Summary (Continued)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$06 ($26)	DDRE	-	-	-	-	-	DDE2	DDE1	DDE0	92
$05 ($25)	PINE	-	-	-	-	-	PINE2	PINE1	PINE0	92
$04 ($24)	Reserved									
$03 ($23)	UDR1	UART1 I/O Data Register								64
$02 ($22)	UCSR1A	RXC1	TXC1	UDRE1	FE1	OR1	-	U2X1	MPCM1	64
$01 ($21)	UCSR1B	RXCIE1	TXCIE1	UDRIE1	RXEN1	TXEN1	CHR91	RXB81	TXB81	65
$00 ($20)	UBRR1	UART1 Baud Rate Register								68

Notes: 1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

2. Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

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
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
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	1/2/3
CP	Rd,Rr	Compare	Rd – Rr	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	Rd – K	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2

Instruction Set Summary (Continued)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
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
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
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

Instruction Set Summary (Continued)

Mnemonics	Operands	Description	Operation	Flags	#Clocks
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 WDR/timer)	None	1

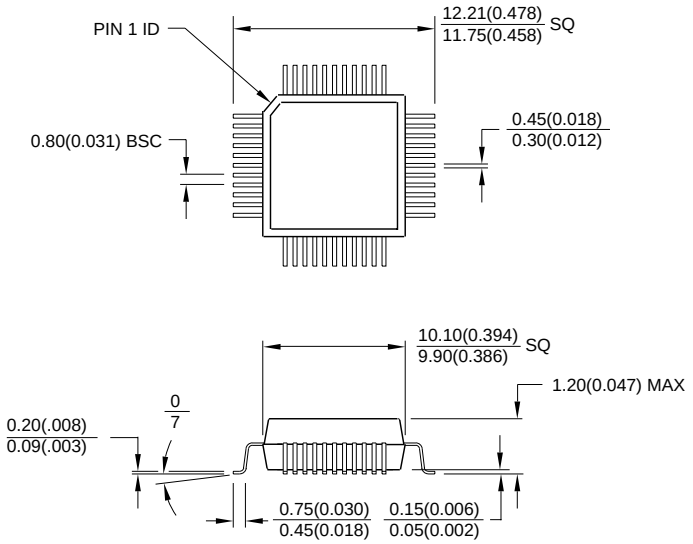
Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
4	2.7 - 5.5V	ATmega161-4AC ATmega161-4JC ATmega161-4PC	44A 44J 40P6	Commercial (0°C to 70°C)
		ATmega161-4AI ATmega161-4JI ATmega161-4PI	44A 44J 40P6	Industrial (-40°C to 85°C)
8	4.0 - 5.5V	ATmega161-8AC ATmega161-8JC ATmega161-8PC	44A 44J 40P6	Commercial (0°C to 70°C)
		ATmega161-8AI ATmega161-8JI ATmega161-8PI	44A 44J 40P6	Industrial (-40°C to 85°C)

Package Type	
44A	44-lead, Thin (1.0 mm) Plastic Gull-Wing Quad Flat Package (TQFP)
44J	44-lead, Plastic J-leaded Chip Carrier (PLCC)
40P6	40-lead, 0.600” Wide, Plastic Dual-in-line Package (PDIP)

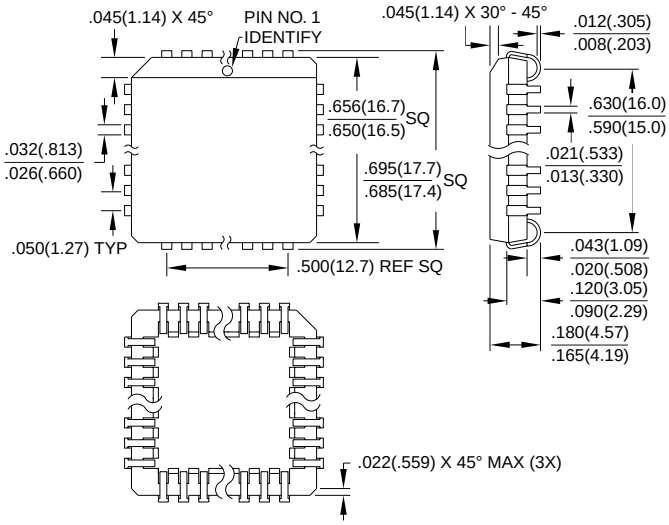
Packaging Information

44A, 44-lead, Thin (1.0 mm) Plastic Gull-Wing Quad Flat Package (TQFP)
Dimensions in Millimeters and (Inches)*

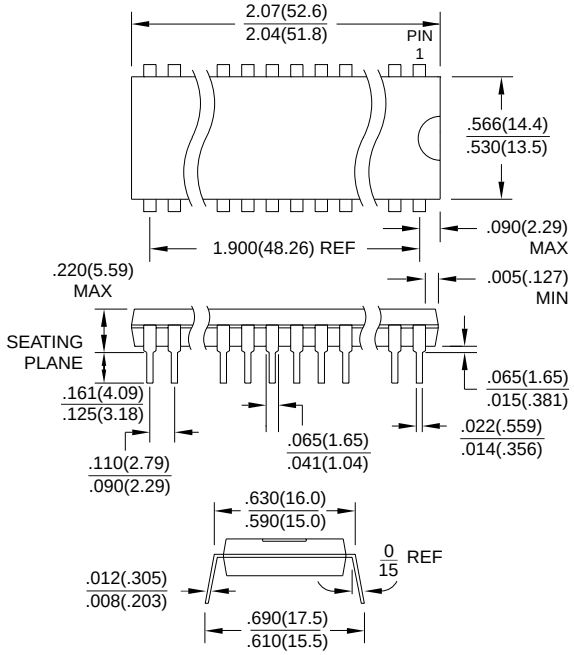


*Controlling dimension: millimeters

44J, 44-lead, Plastic J-leaded Chip Carrier (PLCC)
Dimensions in Inches and (Millimeters)



40P6, 40-lead, 0.600" Wide,
Plastic Dual-in-line Package (PDIP)
Dimensions in Inches and (Millimeters)
JEDEC STANDARD MS-011 AC



Atmel Headquarters

Corporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe
Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686-677
FAX (44) 1276-686-697

Asia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan
Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Fax-on-Demand
North America:
1-(800) 292-8635

International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

1228A–08/99/xM