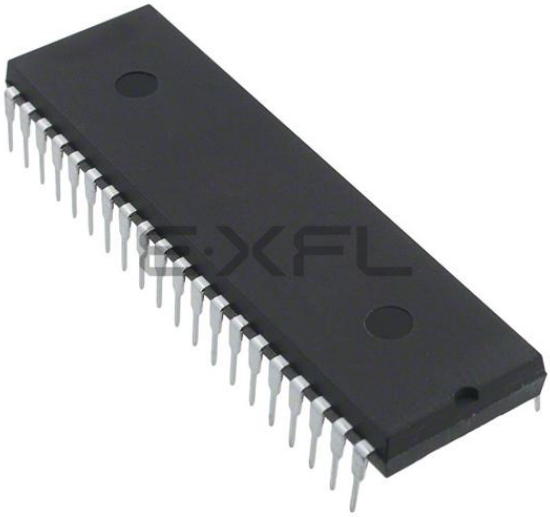


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

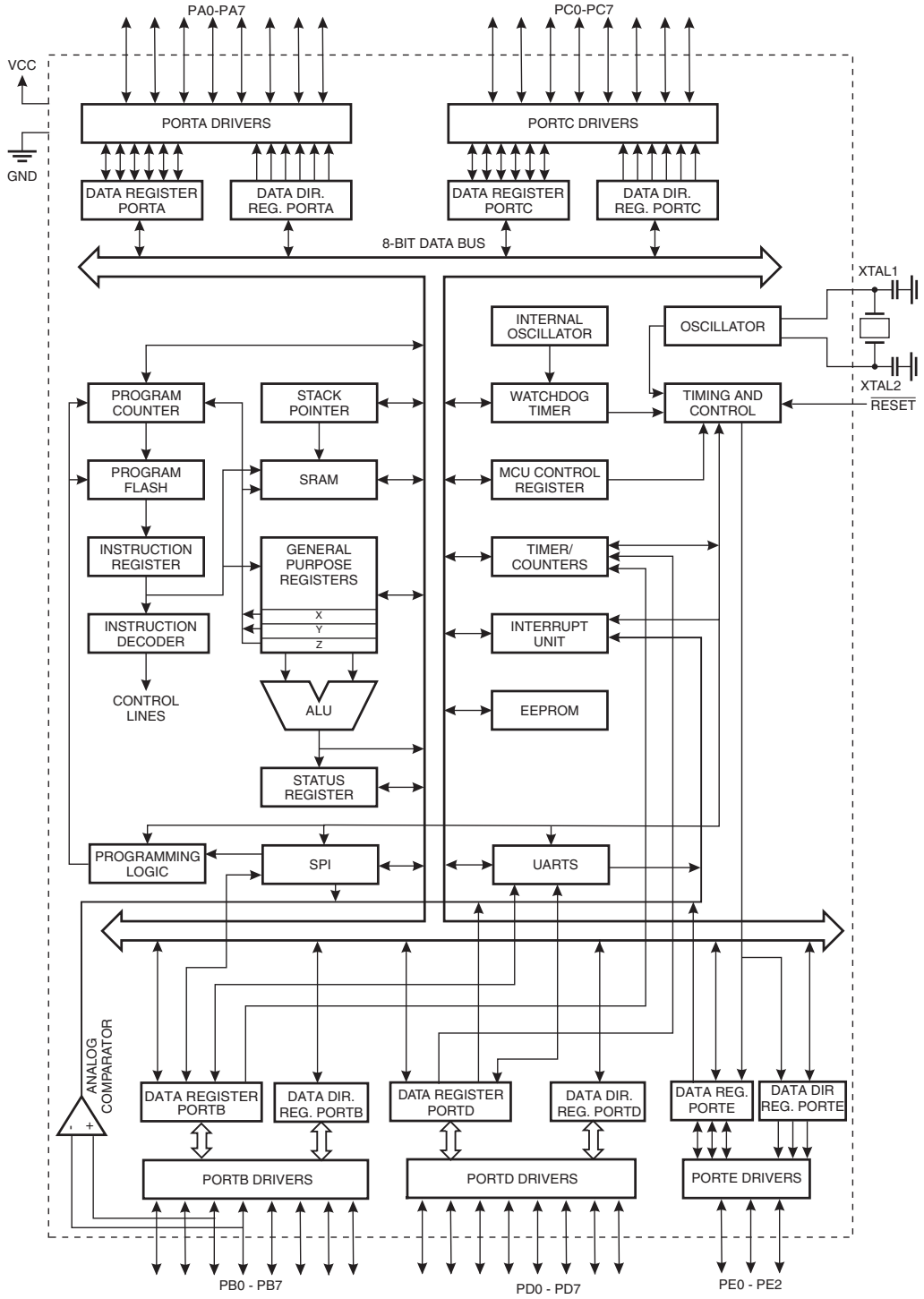
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	4MHz
Connectivity	SPI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	35
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega161l-4pc

Block Diagram

Figure 1. The ATmega161 Block Diagram



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

Self-programmable Flash Program Memory

The ATmega161 contains 16K bytes of On-chip Self-programmable 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 1,000 write/erase cycles. The ATmega161 Program Counter (PC) is 13 bits wide, thus addressing the 8,192 Program memory locations.

See page 110 for a detailed description of Flash data downloading.

See page 13 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 60, specifying the EEPROM Address Registers, the EEPROM Data Register and the EEPROM Control Register.

For the SPI data downloading, see page 125 for a detailed description.

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 byte of internal data SRAM in the ATmega161 are all accessible through all these Addressing modes.

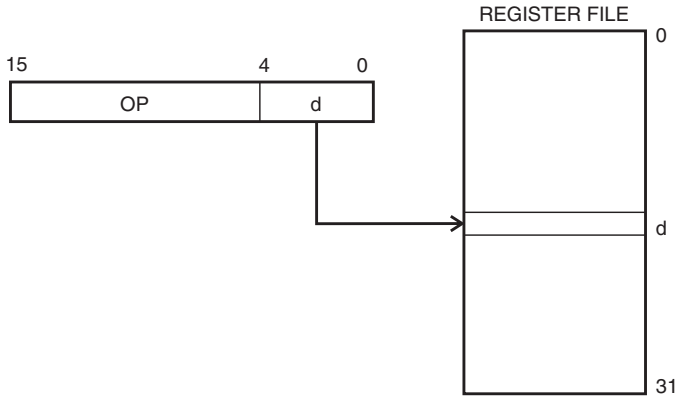
See the next section for a detailed description of the different Addressing modes.

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

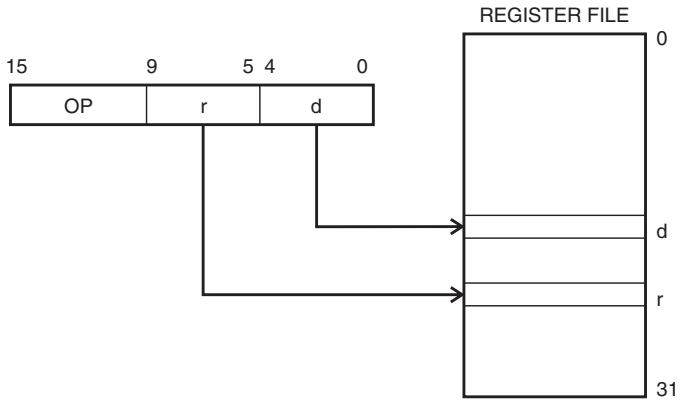
Figure 9. Direct Single Register Addressing



The operand is contained in register d (Rd).

Register Direct, Two Registers Rd and Rr

Figure 10. Direct Register Addressing, Two Registers



I/O Memory

The I/O space definition of the ATmega161 is shown in Table 1.

Table 1. ATmega161 I/O Space[1]

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

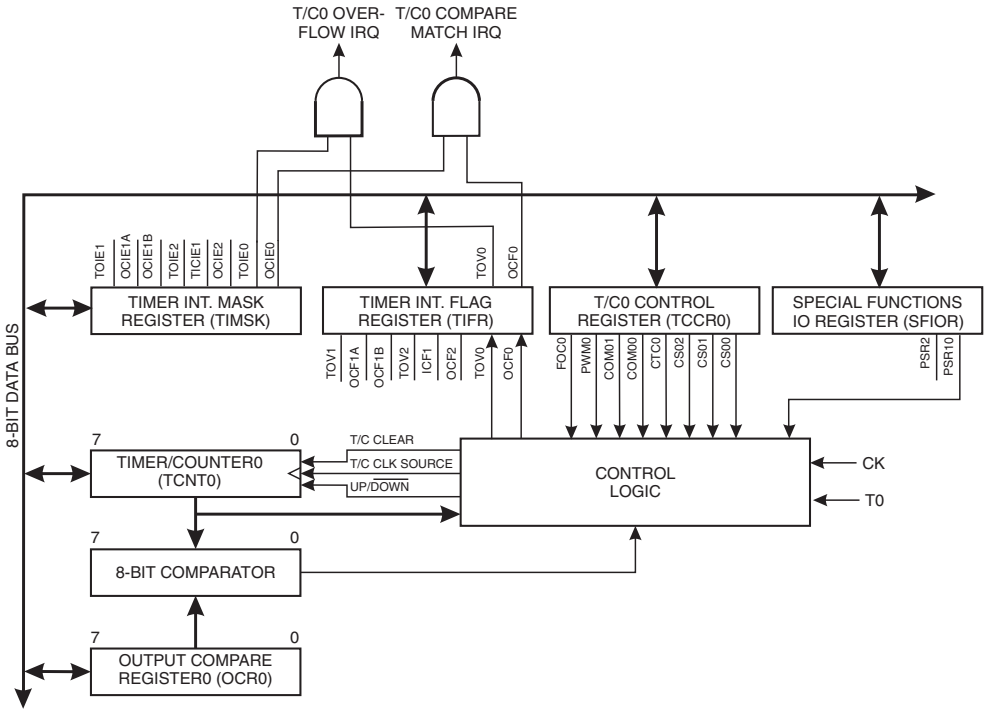
- **Bit 0 – PSR10: Prescaler Reset Timer/Counter1 and Timer/Counter0**

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 Timer/Counters T/C0 and T/C2

Figure 31 shows the block diagram for Timer/Counter0. Figure 32 shows the block diagram for Timer/Counter2.

Figure 31. Timer/Counter0 Block Diagram



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 also applies 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	

- **Bits 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 “0” 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 “0” 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 “0” 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.

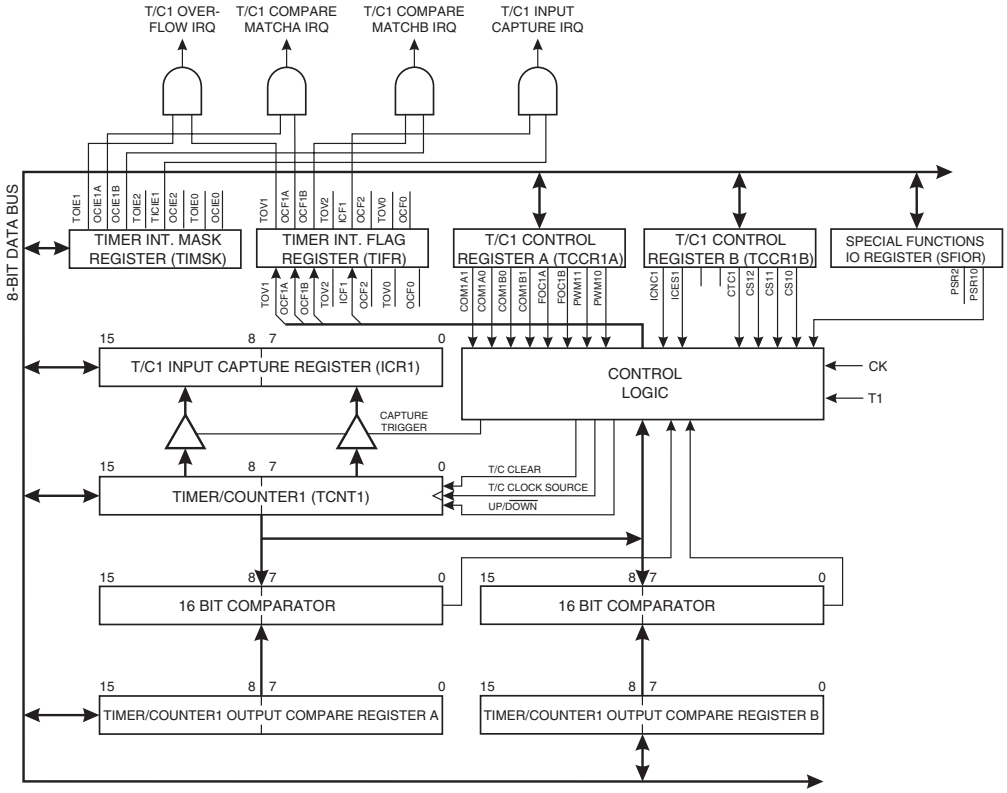
The mechanisms for reading TCNT2, OCR2 and TCCR2 are different. When reading TCNT2, the actual timer value is read. When reading OCR2 or TCCR2, the value in the temporary storage register is read.

- 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 1 before the processor can read the counter value. The Interrupt Flags are updated three 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 three processor cycles plus one timer cycle. The timer is therefore advanced by at least 1 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 35 shows the block diagram for Timer/Counter1.

Figure 35. 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 “Timer/Counter1 Control Register B – TCCR1B”. 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.

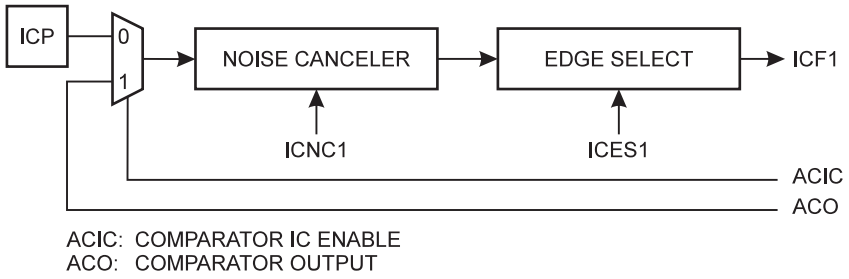
The 16-bit Timer/Counter1 features both a high-resolution and a high-accuracy usage with the lower prescaling opportunities. Similarly, the high-prescaling opportunities make the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

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 an 8-, 9-, or 10-bit Pulse Width 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 55 for a detailed description of 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 36.

Figure 36. ICP Pin Schematic Diagram



If the noise canceler function is enabled, the actual trigger condition for the Capture Event is monitored over four samples, and all four 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.

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	X	0	

- **Bits 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 “1” 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 “1” to the EEMWE bit in EECR (to be able to write a logical “1” to the EEMWE bit, the EEWE bit must be written to zero in the same cycle).
5. Within four clock cycles after setting EEMWE, write a logical “1” 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 four last steps to avoid these problems.

When the write access time 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.

An RC Oscillator is used to time EEPROM write access. The table below lists the typical programming time listed for EEPROM access from CPU.

Table 21. EEPROM Access Time from CPU"See "Typical Characteristics" on page 138 to find RC Oscillator frequency." on page 62[1]

Symbol	No. of RC Oscillator Cycles	Min Programming Time	Max Programming Time
EEPROM write (from CPU)	2048	2.0 ms	3.4 ms

Note: 1. See "Typical Characteristics" on page 138 to find RC Oscillator frequency.

Prevent EEPROM Corruption

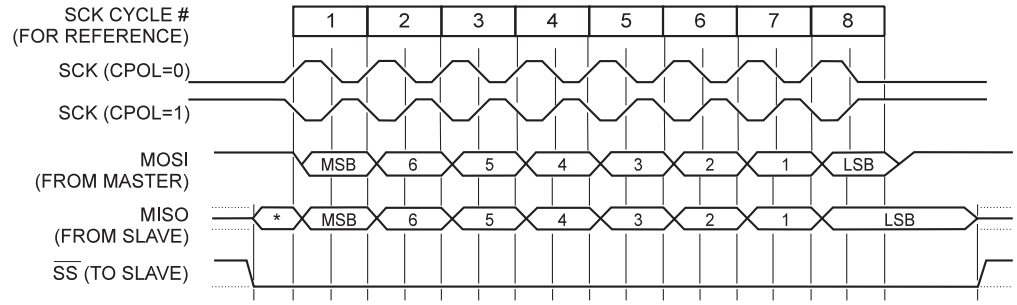
During periods of low V_{CC}, the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. These issues are the same as for board-level systems using the EEPROM, and the same design solutions should be applied.

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. 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 cannot be updated by the CPU unless the boot loader software supports writing to the Flash and the Boot Lock bits are configured so that writing to the Flash memory from the CPU is allowed. See "Boot Loader Support" on page 110 for details.

Figure 43. 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 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 42 and Figure 43 for additional information.

- **Bit 2 – CPHA: Clock Phase**

Refer to Figure 42 or Figure 43 for the functionality of this bit.

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 “1” 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.

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

Table 29. DDAn Effects on Port A Pins[1]

DDAn	PORTAn	I/O	Pull-up	Comment
0	0	Input	No	Tri-state (high-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

Note: 1. 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 55. Port A Schematic Diagrams (Pins PA0 - PA7)

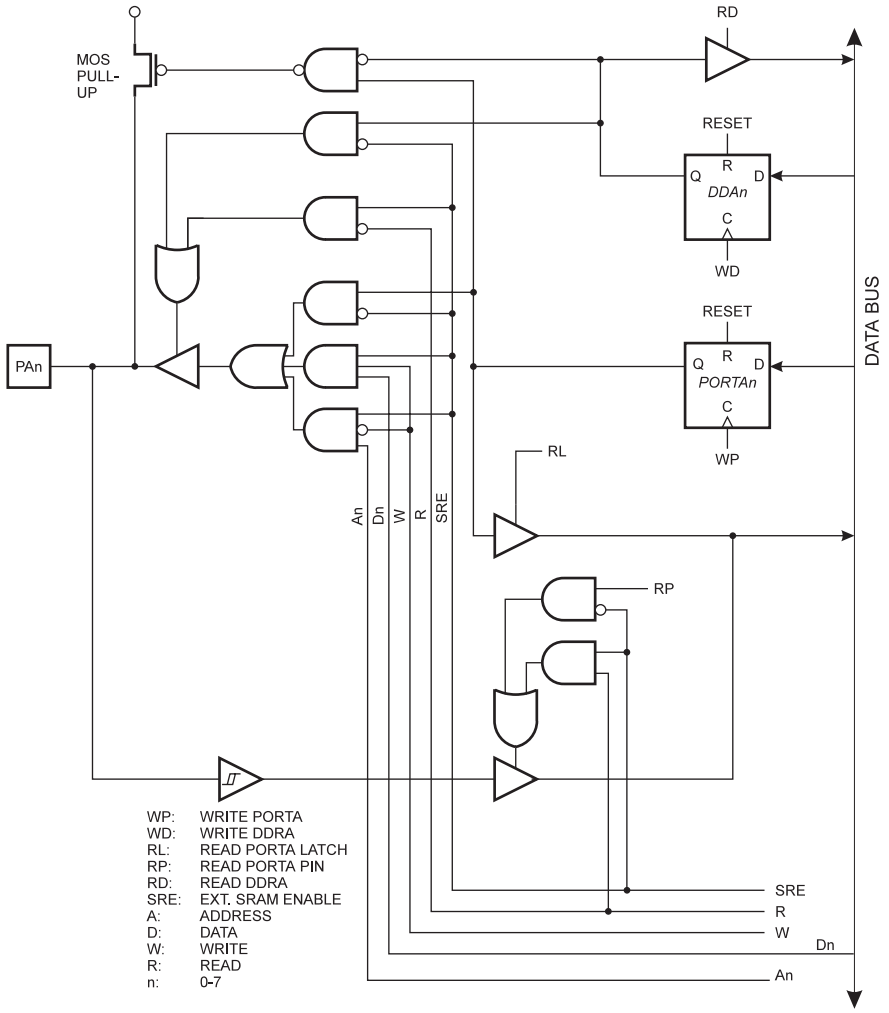


Figure 59. Port B Schematic Diagram (Pin PB4)

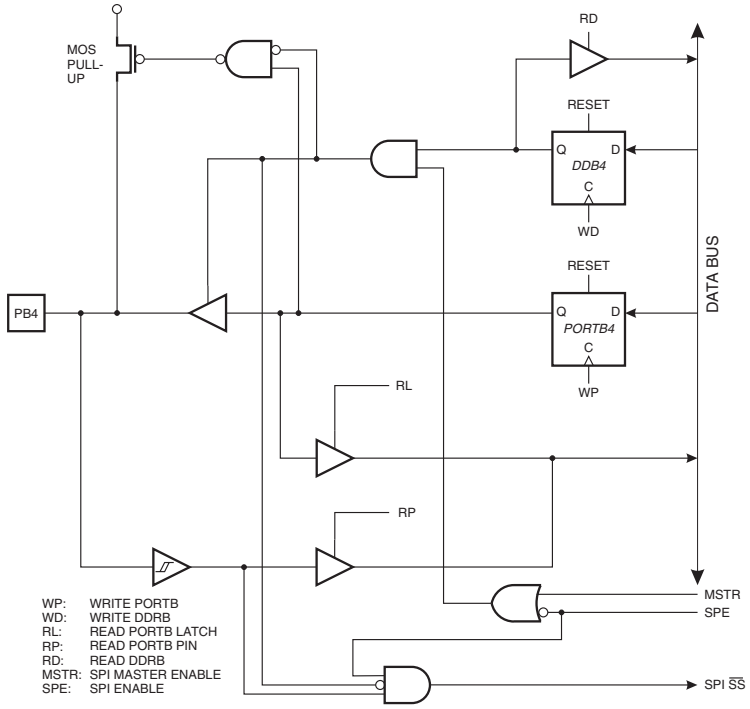
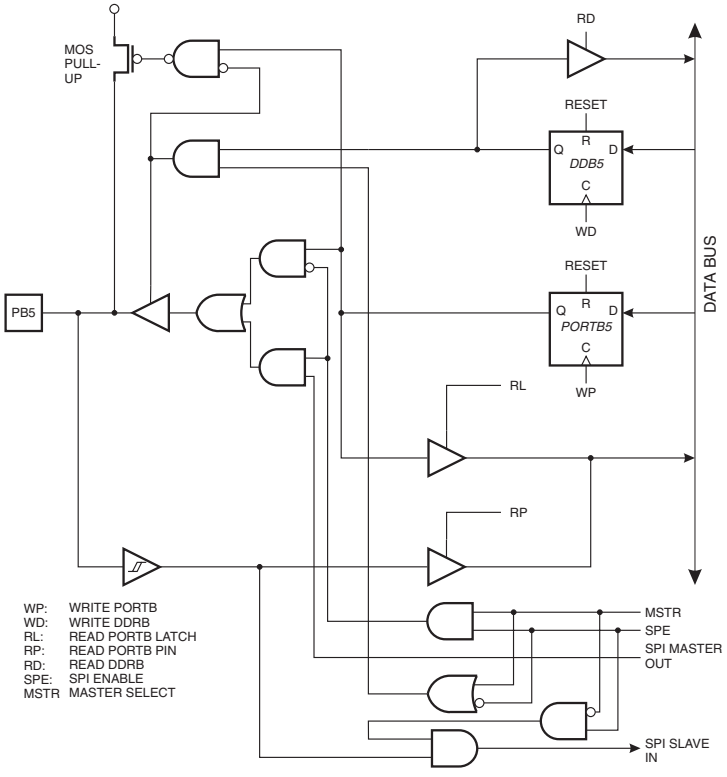


Figure 60. Port B Schematic Diagram (Pin PB5)



Memory Programming

Boot Loader Support

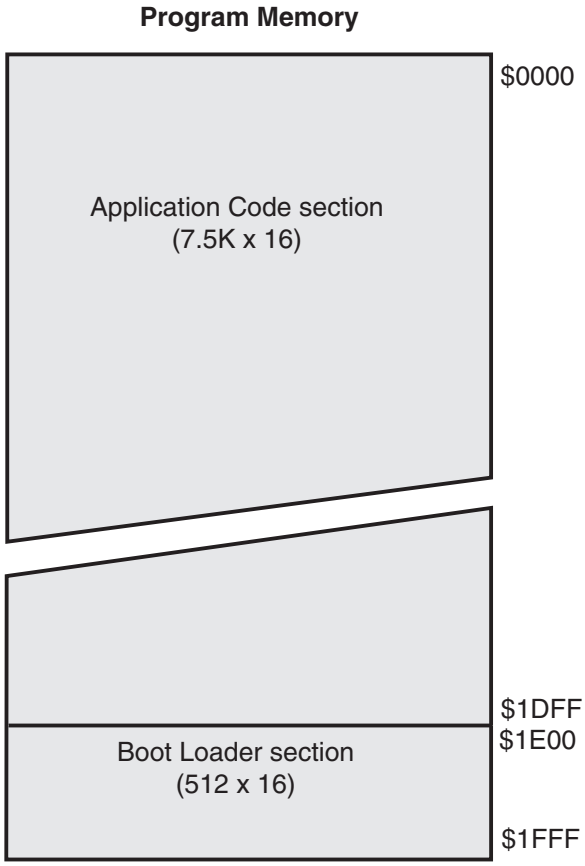
The ATmega161 provides a mechanism for downloading and uploading program code by the MCU itself. This feature allows flexible application software updates, controlled by the MCU using a Flash-resident Boot Loader program.

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 74. Memory Sections



The 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 each contain 128 bytes. The Boot Loader Flash section occupies eight 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 that can be set independently. This gives the user a unique flexibility to select different levels of protection. The user can elect to:

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 76 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, seven bits are needed (128 pages). The five 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. 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

Programming the EEPROM

The programming algorithm for the EEPROM Data memory is as follows (refer to “Programming the Flash” 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

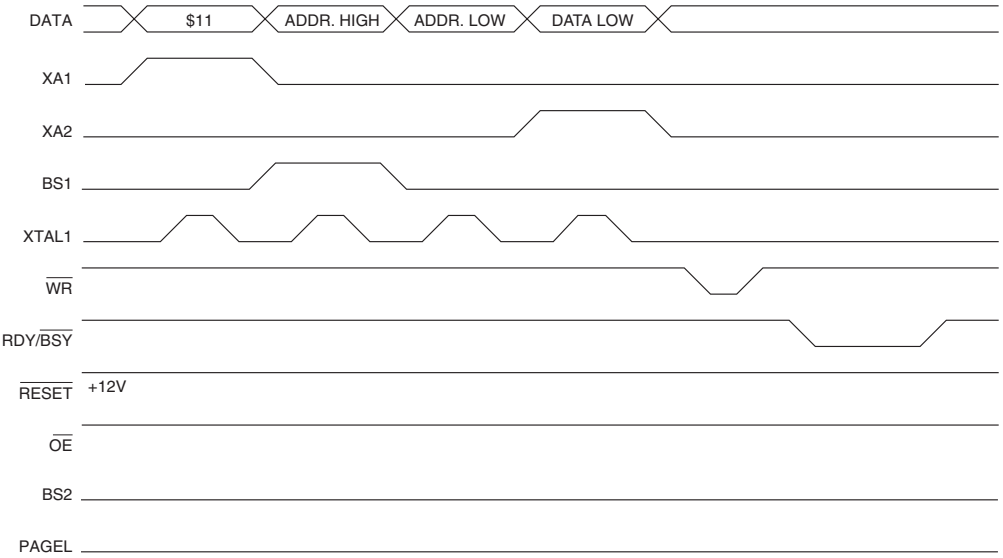
1. Set BS to “0”. This selects low data.
2. Give $\overline{WR}$ a negative pulse. This starts programming of the data byte. RDY/$\overline{BSY}$ goes low.
3. Wait until to RDY/$\overline{BSY}$ goes high before programming the next byte. (See Figure 78 for signal waveforms.)

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

- The command needs to be loaded only once when writing or reading multiple memory locations.
- Address High byte only needs to 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 apply to Flash, EEPROM and Signature bytes reading.

Figure 78. Programming the EEPROM Waveforms



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 46 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 keep 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 reprogrammed 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 46 for t_{WD_EEPROM} value.

Table 46. Minimum Wait Delay before Writing the Next Flash or EEPROM Location

Symbol	Minimum Wait Delay
t_{WD_FLASH}	14 ms
t_{WD_EEPROM}	3.4 ms

Table 47. Minimum Wait Delay after a Chip Erase Command

Symbol	Minimum Wait Delay
t_{WD_ERASE}	28 ms

Figure 80. Serial Programming Waveforms

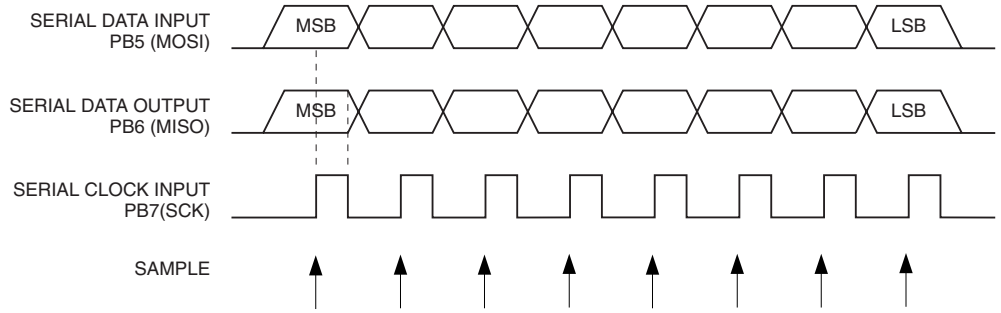


Table of Contents

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

1228D–AVR–02/07