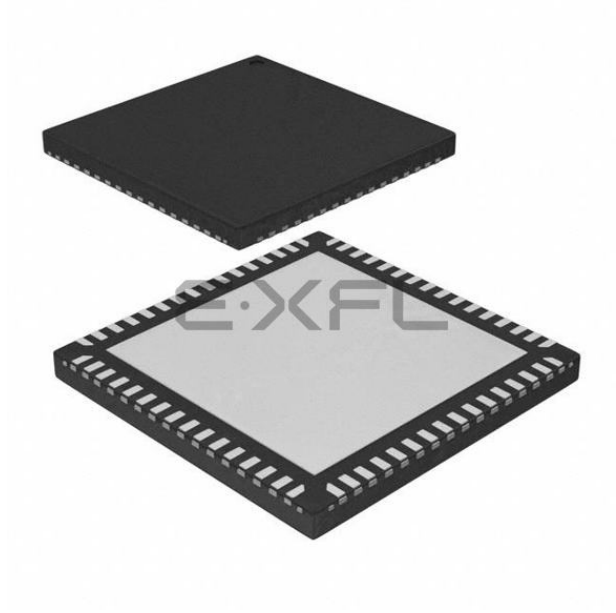
Microchip Technology - ATMEGA649-16MI Datasheet



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.

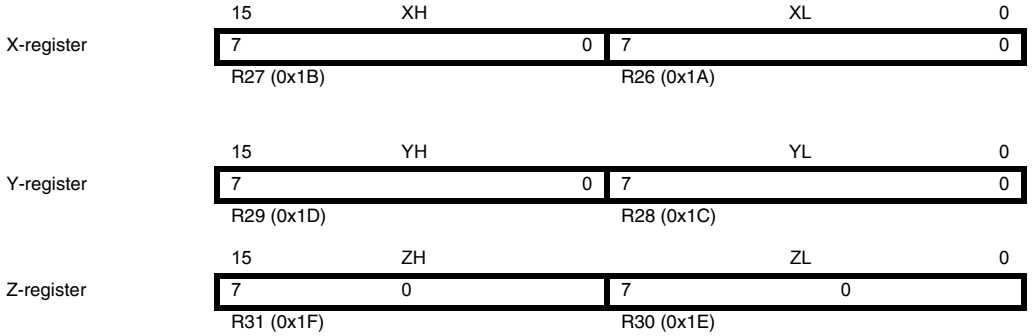
Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	SPI, UART/USART, USI
Peripherals	Brown-out Detect/Reset, LCD, POR, PWM, WDT
Number of I/O	53
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	2K x 8
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-QFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega649-16mi

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

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 6-3.

Figure 6-3. The X-, Y-, and Z-registers



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

6.6 Stack Pointer

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

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

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

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

Table 8-11. Clock Prescaler Select

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

PCINT15, Pin Change Interrupt source 15: The PB7 pin can serve as an external interrupt source.

- **OC1B/PCINT14, Bit 6**

OC1B, Output Compare Match B output: The PB6 pin can serve as an external output for the Timer/Counter1 Output Compare B. The pin has to be configured as an output (DDB6 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.

PCINT14, Pin Change Interrupt Source 14: The PB6 pin can serve as an external interrupt source.

- **OC1A/PCINT13, Bit 5**

OC1A, Output Compare Match A output: The PB5 pin can serve as an external output for the Timer/Counter1 Output Compare A. The pin has to be configured as an output (DDB5 set (one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.

PCINT13, Pin Change Interrupt Source 13: The PB5 pin can serve as an external interrupt source.

- **OC0A/PCINT12, Bit 4**

OC0A, Output Compare Match A output: The PB4 pin can serve as an external output for the Timer/Counter0 Output Compare A. The pin has to be configured as an output (DDB4 set (one)) to serve this function. The OC0A pin is also the output pin for the PWM mode timer function.

PCINT12, Pin Change Interrupt Source 12: The PB4 pin can serve as an external interrupt source.

- **MISO/PCINT11 – Port B, Bit 3**

MISO: Master Data input, Slave Data output pin for SPI. When the SPI is enabled as a Master, this pin is configured as an input regardless of the setting of DDB3. When the SPI is enabled as a Slave, the data direction of this pin is controlled by DDB3. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB3 bit.

PCINT11, Pin Change Interrupt Source 11: The PB3 pin can serve as an external interrupt source.

- **MOSI/PCINT10 – Port B, Bit 2**

MOSI: SPI Master Data output, Slave Data input for SPI. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB2. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB2. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB2 bit.

PCINT10, Pin Change Interrupt Source 10: The PB2 pin can serve as an external interrupt source.

- **SCK/PCINT9 – Port B, Bit 1**

SCK: Master Clock output, Slave Clock input pin for SPI. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB1. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB1. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB1 bit.

PCINT9, Pin Change Interrupt Source 9: The PB1 pin can serve as an external interrupt source.

Table 16-3 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 16-3. Compare Output Mode, Fast PWM[1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 14 or 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on Compare Match, set OC1A/OC1B at BOTTOM (non-inverting mode).
1	1	Set OC1A/OC1B on Compare Match, clear OC1A/OC1B at BOTTOM (inverting mode).

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at BOTTOM. See “Fast PWM Mode” on page 124. for more details.

Table 16-4 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

Table 16-4. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM[1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 9 or 11: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on Compare Match when up-counting. Set OC1A/OC1B on Compare Match when counting down.
1	1	Set OC1A/OC1B on Compare Match when up-counting. Clear OC1A/OC1B on Compare Match when counting down.

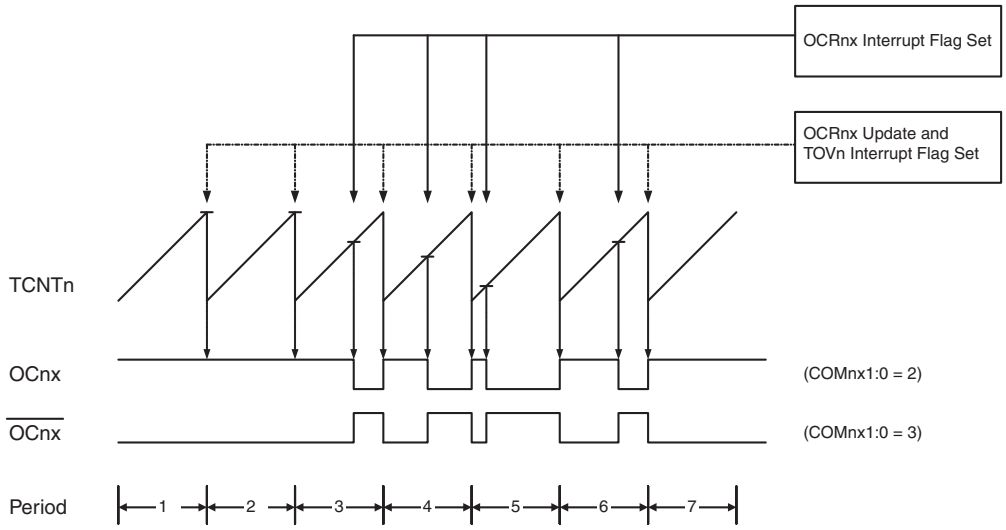
Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. See “Phase Correct PWM Mode” on page 126. for more details.

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

Combined with the WGM13:2 bits found in the TCCR1B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 16-5. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare match (CTC) mode, and three types of Pulse Width Modulation (PWM) modes. (See “Modes of Operation” on page 123.).

PWM mode is shown in Figure 17-6. The TCNT2 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT2 slopes represent compare matches between OCR2A and TCNT2.

Figure 17-6. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV2) is set each time the counter reaches MAX. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC2A pin. Setting the COM2A1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM2A1:0 to three (See Table 17-4 on page 154). The actual OC2A value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC2A Register at the compare match between OCR2A and TCNT2, and clearing (or setting) the OC2A Register at the timer clock cycle the counter is cleared (changes from MAX to BOTTOM).

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

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

The *N* variable represents the prescale factor (1, 8, 32, 64, 128, 256, or 1024).

The extreme values for the OCR2A Register represent special cases when generating a PWM waveform output in the fast PWM mode. If the OCR2A is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR2A equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM2A1:0 bits.)

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC2A to toggle its logical level on each compare match (COM2A1:0 = 1). The waveform generated will have a maximum frequency of $f_{oc2} = f_{clk\_I/O}/2$ when OCR2A is set to zero. This feature is similar to the OC2A toggle in CTC mode, except the double buffer feature of the Output Compare unit is enabled in the fast PWM mode.

- **Bit 3 – CPOL: Clock Polarity**

When this bit is written to one, SCK is high when idle. When CPOL is written to zero, SCK is low when idle. Refer to Figure 18-3 and Figure 18-4 for an example. The CPOL functionality is summarized below:

Table 18-3. CPOL Functionality

CPOL	Leading Edge	Trailing Edge
0	Rising	Falling
1	Falling	Rising

- **Bit 2 – CPHA: Clock Phase**

The settings of the Clock Phase bit (CPHA) determine if data is sampled on the leading (first) or trailing (last) edge of SCK. Refer to Figure 18-3 and Figure 18-4 for an example. The CPOL functionality is summarized below:

Table 18-4. CPHA Functionality

CPHA	Leading Edge	Trailing Edge
0	Sample	Setup
1	Setup	Sample

- **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_{osc} is shown in the following table:

Table 18-5. Relationship Between SCK and the Oscillator Frequency

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

The UPEn bit is set if the next character that can be read from the receive buffer had a Parity Error when received and the Parity Checking was enabled at that point (UPMn1 = 1). This bit is valid until the receive buffer (UDRn) is read.

19.7.6 Disabling the Receiver

In contrast to the Transmitter, disabling of the Receiver will be immediate. Data from ongoing receptions will therefore be lost. When disabled (i.e., the RXENn is set to zero) the Receiver will no longer override the normal function of the RxD port pin. The Receiver buffer FIFO will be flushed when the Receiver is disabled. Remaining data in the buffer will be lost

19.7.7 Flushing the Receive Buffer

The receiver buffer FIFO will be flushed when the Receiver is disabled, i.e., the buffer will be emptied of its contents. Unread data will be lost. If the buffer has to be flushed during normal operation, due to for instance an error condition, read the UDRn I/O location until the RXCn Flag is cleared. The following code example shows how to flush the receive buffer.

Assembly Code Example[1]

```
USART_Flush:
  sbis UCSR0A, RXC0
  ret
  in   r16, UDR0
  rjmp USART_Flush
```

C Code Example[1]

```
void USART_Flush( void )
{
  unsigned char dummy;
  while ( UCSR0A & (1<<RXC0) ) dummy = UDR0;
}
```

Note: 1. See “About Code Examples” on page 9.

19.8 Asynchronous Data Reception

The USART includes a clock recovery and a data recovery unit for handling asynchronous data reception. The clock recovery logic is used for synchronizing the internally generated baud rate clock to the incoming asynchronous serial frames at the RxD pin. The data recovery logic samples and low pass filters each incoming bit, thereby improving the noise immunity of the Receiver. The asynchronous reception operational range depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size in number of bits.

19.8.1 Asynchronous Clock Recovery

The clock recovery logic synchronizes internal clock to the incoming serial frames. Figure 19-5 illustrates the sampling process of the start bit of an incoming frame. The sample rate is 16 times the baud rate for Normal mode, and eight times the baud rate for Double Speed mode. The horizontal arrows illustrate the synchronization variation due to the sampling process. Note the larger time variation when using the Double Speed mode (U2Xn = 1) of operation. Samples denoted zero are samples done when the RxD line is idle (i.e., no communication activity).

19.9 Multi-processor Communication Mode

Setting the Multi-processor Communication mode (MPCMn) bit in UCSRnA enables a filtering function of incoming frames received by the USART Receiver. Frames that do not contain address information will be ignored and not put into the receive buffer. This effectively reduces the number of incoming frames that has to be handled by the CPU, in a system with multiple MCUs that communicate via the same serial bus. The Transmitter is unaffected by the MPCMn setting, but has to be used differently when it is a part of a system utilizing the Multi-processor Communication mode.

If the Receiver is set up to receive frames that contain 5 to 8 data bits, then the first stop bit indicates if the frame contains data or address information. If the Receiver is set up for frames with nine data bits, then the ninth bit (RXB8n) is used for identifying address and data frames. When the frame type bit (the first stop or the ninth bit) is one, the frame contains an address. When the frame type bit is zero the frame is a data frame.

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

19.9.1 Using MPCM

For an MCU to act as a master MCU, it can use a 9-bit character frame format (UCSZ = 7). The ninth bit (TXB8n) must be set when an address frame (TXB8n = 1) or cleared when a data frame (TXB = 0) is being transmitted. The slave MCUs must in this case be set to use a 9-bit character frame format.

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 frame, and all slaves receive and read this frame. 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 and keeps the MPCMn setting.
4. The addressed MCU will receive all data frames until a new address frame is received. The other Slave MCUs, which still have the MPCMn bit set, will ignore the data frames.
5. When the last data frame is received by the addressed MCU, the addressed MCU sets the MPCMn bit and waits for a new address frame from master. The process then repeats from 2.

Using any of the 5- to 8-bit character frame formats is possible, but impractical since the Receiver must change between using n and n+1 character frame formats. This makes full-duplex operation difficult since the Transmitter and Receiver uses the same character size setting. If 5- to 8-bit character frames are used, the Transmitter must be set to use two stop bit (USBSn = 1) since the first stop bit is used for indicating the frame type.

- **Bit 0 – UCPOLn: Clock Polarity**

This bit is used for synchronous mode only. Write this bit to zero when asynchronous mode is used. The UCPOLn bit sets the relationship between data output change and data input sample, and the synchronous clock (XCK).

Table 19-12. UCPOLn Bit Settings

UCPOLn	Transmitted Data Changed (Output of TxD Pin)	Received Data Sampled (Input on RxD Pin)
0	Rising XCK Edge	Falling XCK Edge
1	Falling XCK Edge	Rising XCK Edge

19.11.5 UBRRnL and UBRRnH – USART Baud Rate Registers n

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

- **Bit 15:12 – Reserved Bits**

These bits are reserved for future use. For compatibility with future devices, these bit must be written to zero when UBRRnH is written.

- **Bit 11:0 – UBRR11:0: USART Baud Rate Register**

This is a 12-bit register which contains the USART baud rate. The UBRRnH contains the four most significant bits, and the UBRRnL contains the eight least significant bits of the USART baud rate. Ongoing transmissions by the Transmitter and Receiver will be corrupted if the baud rate is changed. Writing UBRRnL will trigger an immediate update of the baud rate prescaler.

20. USI – Universal Serial Interface

20.1 Features

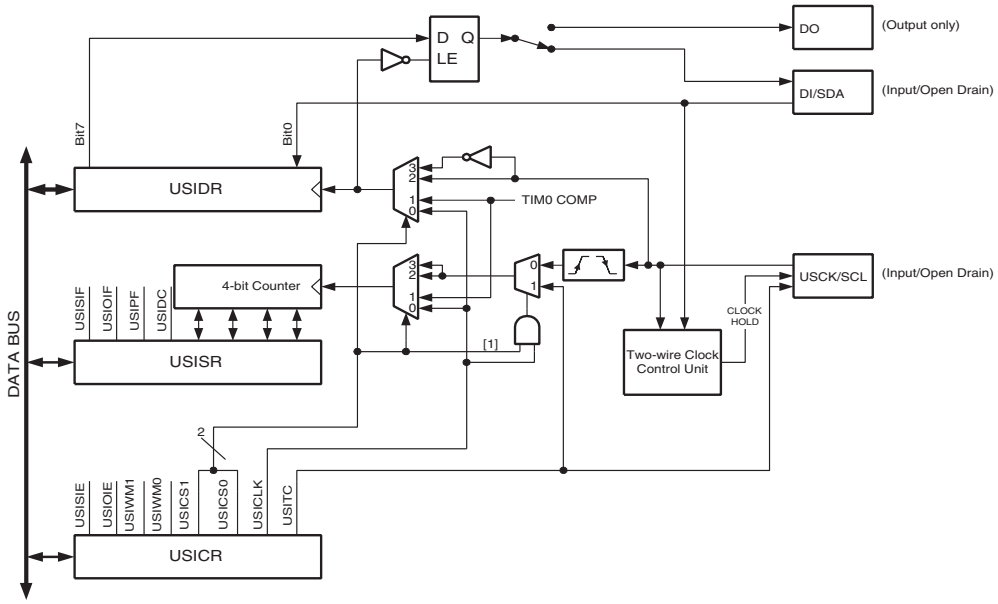
The Universal Serial Interface, or USI, provides the basic hardware resources needed for serial communication. Combined with a minimum of control software, the USI allows significantly higher transfer rates and uses less code space than solutions based on software only. Interrupts are included to minimize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

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

20.2 Overview

A simplified block diagram of the USI is shown on Figure 20-1. For the actual placement of I/O pins, refer to “Pinout ATmega3290/6490” on page 2 and “Pinout ATmega329/649” on page 3. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O Register and bit locations are listed in the “Register Descriptions” on page 203.

Figure 20-1. Universal Serial Interface, Block Diagram



The 8-bit Shift Register is directly accessible via the data bus and contains the incoming and outgoing data. The register has no buffering so the data must be read as quickly as possible to ensure that no data is lost. The most significant bit is connected to one of two output pins depending of the wire mode configuration. A transparent latch is inserted between the Serial Register Output and output pin, which delays the change of data output to the opposite clock edge of the data input sampling. The serial input is always sampled from the Data Input (DI) pin independent of the configuration.

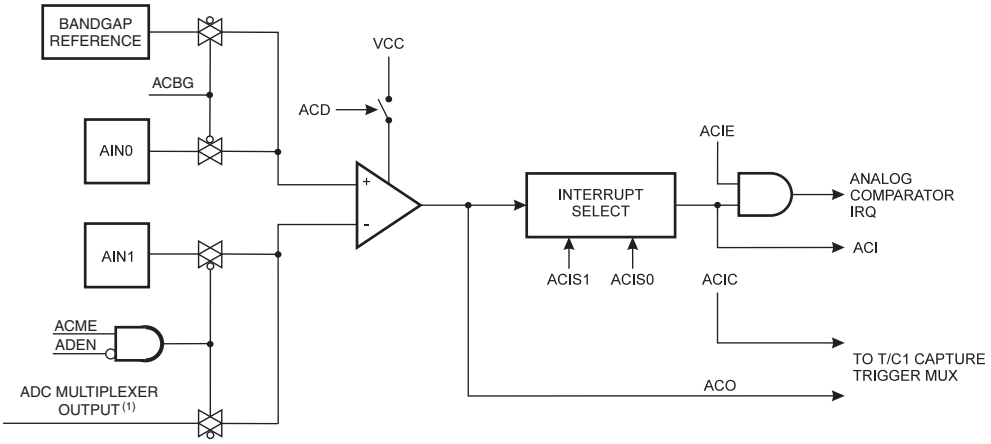
21. Analog Comparator

21.1 Overview

The Analog Comparator compares the input values on the positive pin AIN0 and negative pin AIN1. When the voltage on the positive pin AIN0 is higher than the voltage on the negative pin AIN1, the Analog Comparator output, ACO, is set. The comparator'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 21-1.

The PRADC, in "Power Reduction Register - PRR" on page 35 must be written to zero to use the ADC input MUX.

Figure 21-1. Analog Comparator Block Diagram(2)

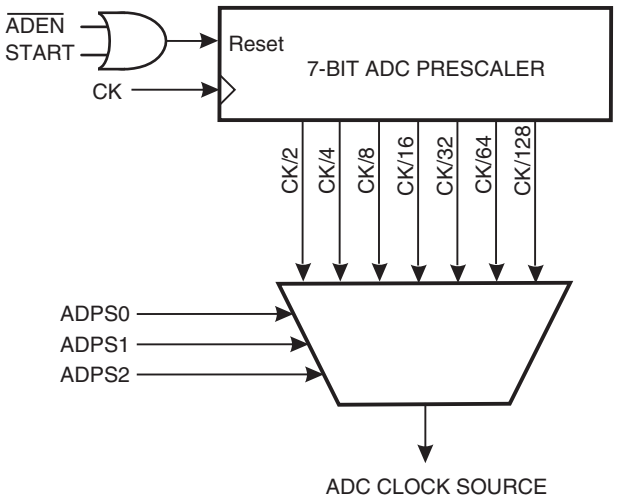


Notes: 1. See Table 1 on page 208.
2. Refer to Figure 1-1 on page 2 and Table 13-5 on page 68 for Analog Comparator pin placement.

If Auto Triggering is enabled, single conversions can be started by writing ADSC in ADCSRA to one. ADSC can also be used to determine if a conversion is in progress. The ADSC bit will be read as one during a conversion, independently of how the conversion was started.

22.4 Prescaling and Conversion Timing

Figure 22-3. ADC Prescaler



By default, the successive approximation circuitry requires an input clock frequency between 50kHz and 200kHz to get maximum resolution. If a lower resolution than 10 bits is needed, the input clock frequency to the ADC can be higher than 200kHz to get a higher sample rate.

The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100kHz. The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSRA. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a single ended conversion by setting the ADSC bit in ADCSRA, the conversion starts at the following rising edge of the ADC clock cycle.

A normal conversion takes 13 ADC clock cycles. The first conversion after the ADC is switched on (ADEN in ADCSRA is set) takes 25 ADC clock cycles in order to initialize the analog circuitry.

When the bandgap reference voltage is used as input to the ADC, it will take a certain time for the voltage to stabilize. If not stabilized the first value read after the first conversion may be wrong.

The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 13.5 ADC clock cycles after the start of an first conversion. When a conversion is complete, the result is written to the ADC Data Registers, and ADIF is set. In Single Conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

When Auto Triggering is used, the prescaler is reset when the trigger event occurs. This assures a fixed delay from the trigger event to the start of conversion. In this mode, the sample-and-hold takes place two ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic. When using Differential mode, along

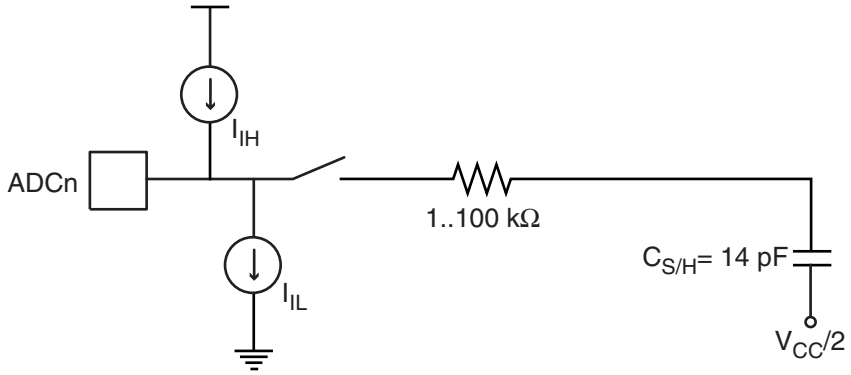
22.5.4 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 22-8. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

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

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

Figure 22-8. Analog Input Circuitry



22.5.5 Analog Noise Canceling Techniques

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

1. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.
2. The AVCC pin on the device should be connected to the digital V_{CC} supply voltage via an LC network as shown in Figure 22-9.
3. Use the ADC noise canceler function to reduce induced noise from the CPU.
4. If any ADC port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

23. LCD Controller

The LCD Controller/driver is intended for monochrome passive liquid crystal display (LCD) with up to four common terminals and up to 25/40 segment terminals.

23.1 Features

- **Display Capacity of 25/40 Segments and Four Common Terminals**
- **Support Static, 1/2, 1/3 and 1/4 Duty**
- **Support Static, 1/2, 1/3 Bias**
- **On-chip LCD Power Supply, only One External Capacitor needed**
- **Display Possible in Power-save Mode for Low Power Consumption**
- **Software Selectable Low Power Waveform Capability**
- **Flexible Selection of Frame Frequency**
- **Software Selection between System Clock or an External Asynchronous Clock Source**
- **Equal Source and Sink Capability to maximize LCD Life Time**
- **LCD Interrupt Can be Used for Display Data Update or Wake-up from Sleep Mode**
- **Segment and Common Pins not Needed for Driving the Display Can be Used as Ordinary I/O Pins**
- **Latching of Display Data gives Full Freedom in Register Update**

23.1.1 Overview

A simplified block diagram of the LCD Controller/Driver is shown in Figure 23-1. For the actual placement of I/O pins, refer to "Pinout ATmega3290/6490" on page 2 and "Pinout ATmega329/649" on page 3.

An LCD consists of several segments (pixels or complete symbols) which can be visible or non visible. A segment has two electrodes with liquid crystal between them. When a voltage above a threshold voltage is applied across the liquid crystal, the segment becomes visible.

The voltage must alternate to avoid an electrophoresis effect in the liquid crystal, which degrades the display. Hence the waveform across a segment must not have a DC-component.

The PRLCD bit in "Power Reduction Register" on page 37 must be written to zero to enable the LCD module.

23.1.2 Definitions

Several terms are used when describing LCD. The definitions in Table 23-1 are used throughout this document.

Table 23-1. Definitions

LCD	A passive display panel with terminals leading directly to a segment
Segment	The least viewing element (pixel) which can be on or off
Common	Denotes how many segments are connected to a segment terminal
Duty	1/(Number of common terminals on a actual LCD display)
Bias	1/(Number of voltage levels used driving a LCD display -1)
Frame Rate	Number of times the LCD segments is energized per second.

Table 23-3. LCD Port Mask (Values in bold are only available in ATmega3290/6490)

LCDPM3	LCDPM2	LCDPM1	LCDPM0	I/O Port in Use as Segment Driver	Maximum Number of Segments
1	**1**	**0**	**1**	**SEG0:36**	**37**
1	**1**	**1**	**0**	**SEG0:38**	**39**
1	**1**	**1**	**1**	**SEG0:39**	**40**

Note: 1. LCDPM3 is reserved and will always read as zero in ATmega329/649.

23.4.3 LCDFRR – LCD Frame Rate Register

Bit	7	6	5	4	3	2	1	0	
(0xE6)	–	LCDPS2	LCDPS1	LCDPS0	–	LCDCD2	LCDCD1	LCDCD0	LCDFRR
Read/Write	R	R/W	R/W	R/W	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – Reserved Bit**

This bit is reserved bit in the ATmega329/3290/649/6490 and will always read as zero.

- **Bits 6:4 – LCDPS2:0: LCD Prescaler Select**

The LCDPS2:0 bits selects tap point from a prescaler. The prescaled output can be further divided by setting the clock divide bits (LCDCD2:0). The different selections are shown in Table 23-4. Together they determine the prescaled LCD clock (clk_{LCD_PS}), which is clocking the LCD module.

Table 23-4. LCD Prescaler Select

LCDPS2	LCDPS1	LCDPS0	Output from Prescaler clk_{LCD}/N	Applied Prescaled LCD Clock Frequency when LCDCD2:0 = 0, Duty = 1/4, and Frame Rate = 64 Hz
0	0	0	$clk_{LCD}/16$	8.1kHz
0	0	1	$clk_{LCD}/64$	33kHz
0	1	0	$clk_{LCD}/128$	66kHz
0	1	1	$clk_{LCD}/256$	130kHz
1	0	0	$clk_{LCD}/512$	260kHz
1	0	1	$clk_{LCD}/1024$	520kHz
1	1	0	$clk_{LCD}/2048$	1MHz
1	1	1	$clk_{LCD}/4096$	2MHz

- **Bit 3 – Reserved Bit**

This bit is reserved bit in the ATmega329/3290/649/6490 and will always read as zero.

- **Bits 2:0 – LCDCD2:0: LCD Clock Divide 2, 1, and 0**

The LCDCD2:0 bits determine division ratio in the clock divider. The various selections are shown in Table 23-5. This Clock Divider gives extra flexibility in frame rate selection.

25. IEEE 1149.1 (JTAG) Boundary-scan

25.1 Features

- **JTAG (IEEE std. 1149.1 compliant) Interface**
- **Boundary-scan Capabilities According to the JTAG Standard**
- **Full Scan of all Port Functions as well as Analog Circuitry having Off-chip Connections**
- **Supports the Optional IDCODE Instruction**
- **Additional Public AVR_RESET Instruction to Reset the AVR**

25.2 System Overview

The Boundary-scan chain has the capability of driving and observing the logic levels on the digital I/O pins, as well as the boundary between digital and analog logic for analog circuitry having off-chip connections. At system level, all ICs having JTAG capabilities are connected serially by the TDI/TDO signals to form a long Shift Register. An external controller sets up the devices to drive values at their output pins, and observe the input values received from other devices. The controller compares the received data with the expected result. In this way, Boundary-scan provides a mechanism for testing interconnections and integrity of components on Printed Circuits Boards by using the four TAP signals only.

The four IEEE 1149.1 defined mandatory JTAG instructions IDCODE, BYPASS, SAMPLE/PRELOAD, and EXTEST, as well as the AVR specific public JTAG instruction AVR_RESET can be used for testing the Printed Circuit Board. Initial scanning of the Data Register path will show the ID-Code of the device, since IDCODE is the default JTAG instruction. It may be desirable to have the AVR device in reset during test mode. If not reset, inputs to the device may be determined by the scan operations, and the internal software may be in an undetermined state when exiting the test mode. Entering reset, the outputs of any port pin will instantly enter the high impedance state, making the HIGHZ instruction redundant. If needed, the BYPASS instruction can be issued to make the shortest possible scan chain through the device. The device can be set in the reset state either by pulling the external $\overline{\text{RESET}}$ pin low, or issuing the AVR_RESET instruction with appropriate setting of the Reset Data Register.

The EXTEST instruction is used for sampling external pins and loading output pins with data. The data from the output latch will be driven out on the pins as soon as the EXTEST instruction is loaded into the JTAG IR-Register. Therefore, the SAMPLE/PRELOAD should also be used for setting initial values to the scan ring, to avoid damaging the board when issuing the EXTEST instruction for the first time. SAMPLE/PRELOAD can also be used for taking a snapshot of the external pins during normal operation of the part.

The JTAGEN Fuse must be programmed and the JTD bit in the I/O Register MCUCR must be cleared to enable the JTAG Test Access Port.

When using the JTAG interface for Boundary-scan, using a JTAG TCK clock frequency higher than the internal chip frequency is possible. The chip clock is not required to run.

25.3 Data Registers

The Data Registers relevant for Boundary-scan operations are:

- Bypass Register
- Device Identification Register
- Reset Register
- Boundary-scan Chain

Table 25-7. ATmega329/649 Boundary-scan Order, 64-pin (Continued)

Bit Number	Signal Name	Module
133	PB0.Data	Port B
132	PB0.Control	
131	PB0.Pull-up_Enable	
130	PB1.Data	
129	PB1.Control	
128	PB1.Pull-up_Enable	
127	PB2.Data	
126	PB2.Control	
125	PB2.Pull-up_Enable	
124	PB3.Data	
123	PB3.Control	
122	PB3.Pull-up_Enable	
121	PB4.Data	
120	PB4.Control	
119	PB4.Pull-up_Enable	
118	PB5.Data	
117	PB5.Control	
116	PB5.Pull-up_Enable	
115	PB6.Data	
114	PB6.Control	
113	PB6.Pull-up_Enable	
112	PB7.Data	
111	PB7.Control	
110	PB7.Pull-up_Enable	
109	PG3.Data	Port G
108	PG3.Control	
107	PG3.Pull-up_Enable	
106	PG4.Data	
105	PG4.Control	
104	PG4.Pull-up_Enable	
103	PG5	(Observe Only)
102	RSTT	Reset Logic (Observe-only)
101	RSTHV	

27.6 Parallel Programming

27.6.1 Enter Programming Mode

The following algorithm puts the device in Parallel (High-voltage) Programming mode:

1. Set Prog_enable pins listed in Table 27-7 on page 297 to “0000”, RESET pin and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND.
3. Ensure that V_{CC} reaches at least 1.8V within the next 20 µs.
4. Wait 20 - 60 µs, and apply 11.5 - 12.5V to RESET.
5. Keep the Prog_enable pins unchanged for at least 10µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
6. Wait at least 300 µs before giving any parallel programming commands.
7. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

If the rise time of the V_{CC} is unable to fulfill the requirements listed above, the following alternative algorithm can be used.

1. Set Prog_enable pins listed in Table 27-7 on page 297 to “0000”, RESET pin to 0V and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND.
3. Monitor V_{CC}, and as soon as V_{CC} reaches 0.9 - 1.1V, apply 11.5 - 12.5V to RESET.
4. Keep the Prog_enable pins unchanged for at least 10µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
5. Wait until V_{CC} actually reaches 4.5 -5.5V before giving any parallel programming commands.
6. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

27.6.2 Considerations for Efficient Programming

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

- The command needs only be loaded once when writing or reading multiple memory locations.
- Skip writing the data value 0xFF, that is the contents of the entire EEPROM (unless the EESAVE Fuse is programmed) and Flash after a Chip Erase.
- Address high byte needs only be loaded before programming or reading a new 256 word window in Flash or 256 byte EEPROM. This consideration also applies to Signature bytes reading.

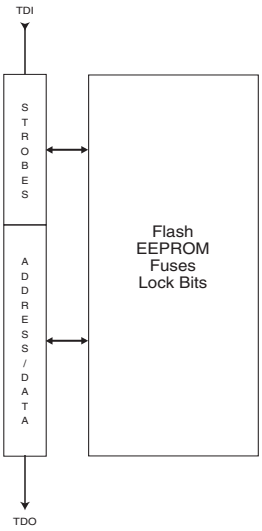
27.6.3 Chip Erase

The Chip Erase will erase the Flash and EEPROM[1] memories plus Lock bits. The Lock bits are not reset until the program memory has been completely erased. The Fuse bits are not changed. A Chip Erase must be performed before the Flash and/or EEPROM are reprogrammed.

Note: 1. The EEPRPOM memory is preserved during Chip Erase if the EESAVE Fuse is programmed.

Load Command “Chip Erase”

Figure 27-15. Programming Command Register



32.4 ATmega6490

Speed (MHz)[3]	Power Supply	Ordering Code[2]	Package Type[1]	Operational Range
8	1.8 - 5.5V	ATmega6490V-8AU ATmega6490V-8AUR[4]	100A 100A	Industrial (-40°C to 85°C)
16	2.7 - 5.5V	ATmega6490-16AU ATmega6490-16AUR[4]	100A 100A	Industrial (-40°C to 85°C)

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging alternative, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. For Speed Grades see Figure 28-1 on page 328 and Figure 28-2 on page 328.
4. Tape & Reel

Package Type	
64A	64-lead, 14 x 14 x 1.0 mm, Thin Profile Plastic Quad Flat Package (TQFP)
64M1	64-pad, 9 x 9 x 1.0 mm, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
100A	100-lead, 14 x 14 x 1.0 mm, 0.5 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)