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	Active
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	54
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	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atmega169p-15at

9.5 Register Description

9.5.1 MCUSR – MCU Status Register

The MCU status register provides information on which reset source caused an MCU reset.

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	–	–	–	JTRF	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	See Bit Description					

- **Bit 4 – JTRF: JTAG Reset Flag**

This bit is set if a reset is being caused by a logic one in the JTAG reset register selected by the JTAG instruction AVR_RESET. This bit is reset by a power-on reset, or by writing a logic zero to the flag.

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

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

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

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

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

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

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

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

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

9.5.2 WDTCR – Watchdog Timer Control Register

Bit	7	6	5	4	3	2	1	0	
(0x60)	–	–	–	WDCE	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 and will always read as zero.

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

This bit must be set when the WDE bit is written to logic zero. Otherwise, the watchdog will not be disabled. Once written to one, hardware will clear this bit after four clock cycles. Refer to the description of the WDE bit for a watchdog disable procedure. This bit must also be set when changing the prescaler bits.
See Section 9.4.1 “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 44

14.3 Accessing 16-bit Registers

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

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

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

The following code examples show how to access the 16-bit timer registers assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCR1A/B and ICR1 registers. Note that when using “C”, the compiler handles the 16-bit access.

Assembly Code Examples(1)

```
...
; Set TCNT1 to 0x01FF
ldi    r17,0x01
ldi    r16,0xFF
out    TCNT1H,r17
out    TCNT1L,r16
; Read TCNT1 into r17:r16
in     r16,TCNT1L
in     r17,TCNT1H
...
```

C Code Examples(1)

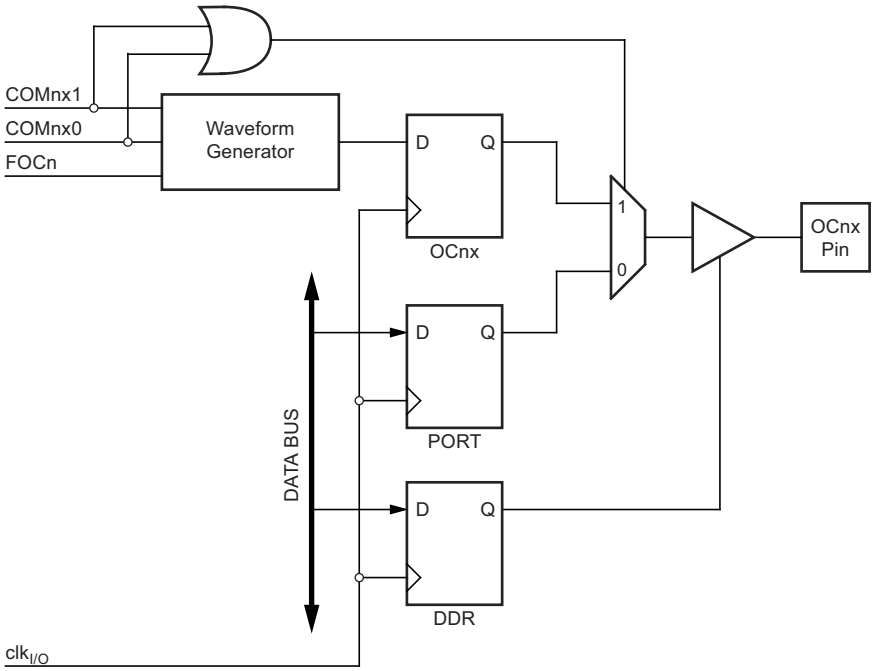
```
unsigned int i;
...
/* Set TCNT1 to 0x01FF */
TCNT1 = 0x1FF;
/* Read TCNT1 into i */
i = TCNT1;
...
```

Note: 1. See Section 4. “About Code Examples” on page 8.

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

It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit timer registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

Figure 14-5. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the output compare (OC1x) from the waveform generator if either of the COM1x1:0 bits are set. However, the OC1x pin direction (input or output) is still controlled by the *data direction register* (DDR) for the port pin. The data direction register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the waveform generation mode, but there are some exceptions. Refer to Table 14-2 on page 111, Table 14-3 on page 111 and Table 14-4 on page 112 for details.

The design of the output compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x1:0 bit settings are reserved for certain modes of operation.
See Section 14.11 “16-bit Timer/Counter Register Description” on page 111

The COM1x1:0 bits have no effect on the input capture unit.

14.8.1 Compare Output Mode and Waveform Generation

The waveform generator uses the COM1x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x1:0 = 0 tells the waveform generator that no action on the OC1x register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 14-2 on page 111. For fast PWM mode refer to Table 14-3 on page 111, and for phase correct and phase and frequency correct PWM refer to Table 14-4 on page 112.

A change of the COM1x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC1x strobe bits.

14.9 Modes of Operation

The mode of operation, i.e., the behavior of the timer/counter and the output compare pins, is defined by the combination of the waveform generation mode (WGM13:0) and compare output mode (COM1x1:0) bits. The compare output mode bits do not affect the counting sequence, while the waveform generation mode bits do. The COM1x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM1x1:0 bits control whether the output should be set, cleared or toggle at a compare match
(see Section 14.8 “Compare Match Output Unit” on page 101).

For detailed timing information refer to Section 14.10 “Timer/Counter Timing Diagrams” on page 109.

The extreme values for the OCR1x register represents special cases when generating a PWM waveform output in the phase and frequency correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be set to high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values. If OCR1A is used to define the TOP value (WGM13:0 = 9) and COM1A1:0 = 1, the OC1A output will toggle with a 50% duty cycle.

14.10 Timer/Counter Timing Diagrams

The timer/counter is a synchronous design and the timer clock (clk_{T1}) is therefore shown as a clock enable signal in the following figures. The figures include information on when interrupt flags are set, and when the OCR1x register is updated with the OCR1x buffer value (only for modes utilizing double buffering). Figure 14-10 shows a timing diagram for the setting of OCF1x.

Figure 14-10.Timer/Counter Timing Diagram, Setting of OCF1x, no Prescaling

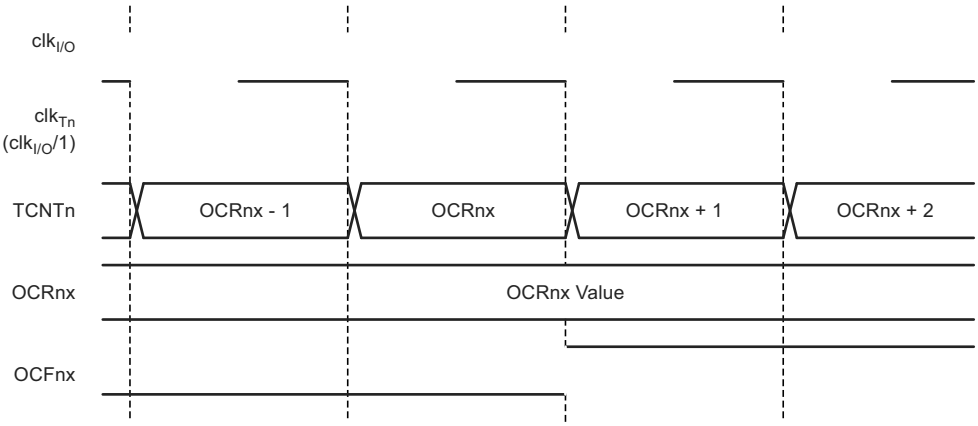
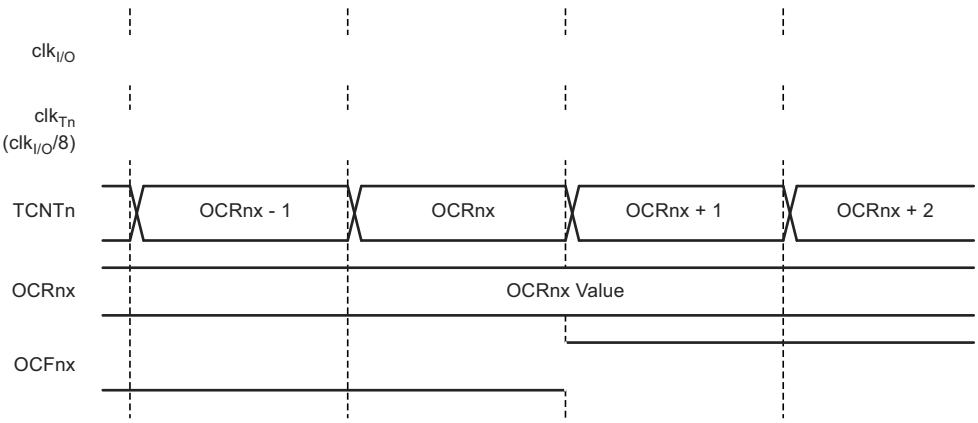


Figure 14-11 shows the same timing data, but with the prescaler enabled.

Figure 14-11.Timer/Counter Timing Diagram, Setting of OCF1x, with Prescaler ($f_{clk\_I/O}/8$)



16.5.1 Compare Output Mode and Waveform Generation

The waveform generator uses the COM2A1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM2A1:0 = 0 tells the waveform generator that no action on the OC2A register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 16-3 on page 132. For fast PWM mode, refer to Table 16-4 on page 132, and for phase correct PWM refer to Table 16-5 on page 133.

A change of the COM2A1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC2A strobe bits.

16.6 Modes of Operation

The mode of operation, i.e., the behavior of the timer/counter and the output compare pins, is defined by the combination of the waveform generation mode (WGM21:0) and compare output mode (COM2A1:0) bits. The compare output mode bits do not affect the counting sequence, while the waveform generation mode bits do. The COM2A1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM2A1:0 bits control whether the output should be set, cleared, or toggled at a compare match (see Section 16.5 “Compare Match Output Unit” on page 123).

For detailed timing information refer to Section 16.7 “Timer/Counter Timing Diagrams” on page 128.

16.6.1 Normal Mode

The simplest mode of operation is the normal mode (WGM21:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the timer/counter overflow flag (TOV2) will be set in the same timer clock cycle as the TCNT2 becomes zero. The TOV2 flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV2 flag, the timer resolution can be increased by software. There are no special cases to consider in the normal mode, a new counter value can be written anytime.

The output compare unit can be used to generate interrupts at some given time. Using the output compare to generate waveforms in normal mode is not recommended, since this will occupy too much of the CPU time.

16.6.2 Clear Timer on Compare Match (CTC) Mode

In clear timer on compare or CTC mode (WGM21:0 = 2), the OCR2A register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT2) matches the OCR2A. The OCR2A defines the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 16-5. The counter value (TCNT2) increases until a compare match occurs between TCNT2 and OCR2A, and then counter (TCNT2) is cleared.

Figure 16-5. CTC Mode, Timing Diagram

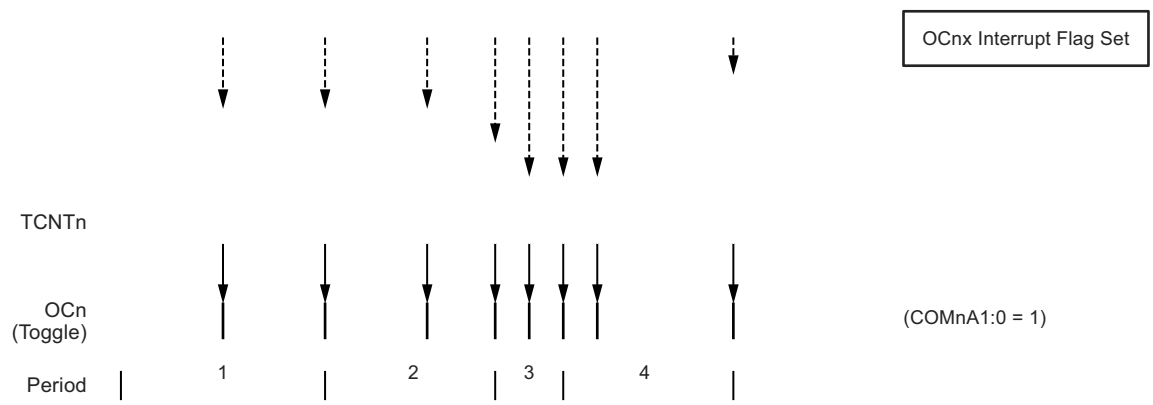


Table 16-5 shows the COM2A1:0 bit functionality when the WGM21:0 bits are set to phase correct PWM mode.

Table 16-5. Compare Output Mode, Phase Correct PWM Mode(1)

COM2A1	COM2A0	Description
0	0	Normal port operation, OC2A disconnected.
0	1	Reserved
1	0	Clear OC2A on compare match when up-counting. Set OC2A on compare match when downcounting.
1	1	Set OC2A on compare match when up-counting. Clear OC2A on compare match when downcounting.

Note: 1. A special case occurs when OCR2A equals TOP and COM2A1 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See Section 16.6.4 “Phase Correct PWM Mode” on page 126 for more details.

- **Bit 2:0 – CS22:0: Clock Select**

The three clock select bits select the clock source to be used by the timer/counter, see Table 16-6.

Table 16-6. Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (timer/counter stopped).
0	0	1	clk_{T2S}/(no prescaling)
0	1	0	clk_{T2S}/8 (from prescaler)
0	1	1	clk_{T2S}/32 (from prescaler)
1	0	0	clk_{T2S}/64 (from prescaler)
1	0	1	clk_{T2S}/128 (from prescaler)
1	1	0	clk_{T2S}/256 (from prescaler)
1	1	1	clk_{T2S}/1024 (from prescaler)

16.10.2 TCNT2 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
(0xB2)	**TCNT2[7:0]**								**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	

The timer/counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT2 register blocks (removes) the compare match on the following timer clock. Modifying the counter (TCNT2) while the counter is running, introduces a risk of missing a compare match between TCNT2 and the OCR2A register.

17.3 $\overline{SS}$ Pin Functionality

17.3.1 Slave Mode

When the SPI is configured as a slave, the slave select ($\overline{SS}$) 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 driven high.

The $\overline{SS}$ pin is useful for packet/byte synchronization to keep the slave bit counter synchronous with the master clock generator. When the $\overline{SS}$ pin is driven high, the SPI slave will immediately reset the send and receive logic, and drop any partially received data in the shift register.

17.3.2 Master Mode

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. Typically, the pin will be driving the $\overline{SS}$ pin of the SPI slave.

If $\overline{SS}$ is configured as an input, it must be held high to ensure master SPI operation. If the $\overline{SS}$ pin is driven low by peripheral circuitry when the SPI is configured as a 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. If the MSTR bit has been cleared by a slave select, it must be set by the user to re-enable SPI master mode.

17.5 Register Description

17.5.1 SPCR – SPI Control Register

Bit	7	6	5	4	3	2	1	0	
0x2C (0x4C)	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 written to 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 written to one, the LSB of the data word is transmitted first.

When the DORD bit is written to zero, the MSB of the data word is transmitted first.

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

This bit selects master SPI mode when written to one, and slave SPI mode when written logic 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 written to one, SCK is high when idle. When CPOL is written to zero, SCK is low when idle.
Refer to Figure 17-3 and Figure 17-4 for an example. The CPOL functionality is summarized below:

Table 17-3. 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 17-3 and Figure 17-4 on page 141 for an example. The CPOL functionality is summarized below:

Table 17-4. 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:

The following code demonstrates how to use the USI module as a SPI master with maximum speed (fsck = fck/4):

```
SPITransfer_Fast:

       sts    USIDR,r16
       ldi    r16,(1<<USIWM0)|(0<<USICS0)|(1<<USITC)
       ldi    r17,(1<<USIWM0)|(0<<USICS0)|(1<<USITC)|(1<<USICLK)

       sts    USICR,r16 ; MSB
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16
       sts    USICR,r17
       sts    USICR,r16 ; LSB
       sts    USICR,r17

       lds    r16,USIDR
       ret
```

19.2.3 SPI Slave Operation Example

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

```
init:
       ldi    r16,(1<<USIWM0)|(1<<USICS1)
       sts    USICR,r16
...
SlaveSPITransfer:
       sts    USIDR,r16
       ldi    r16,(1<<USIOIF)
       sts    USISR,r16
SlaveSPITransfer_loop:
       lds    r16, USISR
       sbrs   r16, USIOIF
       rjmp   SlaveSPITransfer_loop
       lds    r16,USIDR
       ret
```

The code is size optimized using only eight instructions (+ ret). The code example assumes that the DO is configured as output and USCK pin is configured as input in the DDR register. The value stored in register r16 prior to the function is called is transferred to the master device, and when the transfer is completed the data received from the master is stored back into the r16 register.

Note that the first two instructions is for initialization only and needs only to be executed once.These instructions sets three-wire mode and positive edge shift register clock. The loop is repeated until the USI counter overflow flag is set.

22. LCD Controller

22.1 Features

- Display capacity of 25 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

22.2 Overview

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

A simplified block diagram of the LCD controller/driver is shown in Figure 22-1 on page 198. For the actual placement of I/O pins, see Section 1-1 “Pinout ATmega169P” 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 Section 8.9.2 “PRR – Power Reduction Register” on page 38 must be written to zero to enable the LCD module.

22.2.1 Definitions

Several terms are used when describing LCD. The definitions in Table 22-1 are used throughout this document.

Table 22-1. 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.

24. IEEE 1149.1 (JTAG) Boundary-scan

24.1 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

24.2 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.

24.3 Data Registers

The data registers relevant for boundary-scan operations are:

- Bypass register
- Device identification register
- Reset register
- Boundary-scan chain

24.3.1 Bypass Register

The bypass register consists of a single Shift register stage. When the bypass register is selected as path between TDI and TDO, the register is reset to 0 when leaving the capture-DR controller state. The bypass register can be used to shorten the scan chain on a system when the other devices are to be tested.

Table 24-1 summarizes the scan registers for the external clock pin XTAL1, oscillators with XTAL1/XTAL2 connections as well as 32kHz timer oscillator.

Table 24-1. Scan Signals for the Oscillator(1)(2)(3)

Enable Signal	Scanned Clock Line	Clock Option	Scanned Clock Line when not Used
EXTCLKEN	EXTCLK (XTAL1)	External clock	0
OSCON	OSCCK	External crystal External ceramic resonator	1
OSC32EN	OSC32CK	Low freq. external crystal	1

Notes:
1. Do not enable more than one clock source as main clock at a time.
2. Scanning an oscillator output gives unpredictable results as there is a frequency drift between the internal oscillator and the JTAG TCK clock. If possible, scanning an external clock is preferred.
3. The clock configuration is programmed by fuses. As a fuse is not changed run-time, the clock configuration is considered fixed for a given application. The user is advised to scan the same clock option as to be used in the final system. The enable signals are supported in the scan chain because the system logic can disable clock options in sleep modes, thereby disconnecting the oscillator pins from the scan path if not provided.

24.5.4 Scanning the Analog Comparator

The relevant comparator signals regarding boundary-scan are shown in Figure 24-7. The boundary-scan cell from Figure 24-8 on page 225 is attached to each of these signals. The signals are described in Table 24-2 on page 225.

The comparator need not be used for pure connectivity testing, since all analog inputs are shared with a digital port pin as well.

Figure 24-7. Analog Comparator

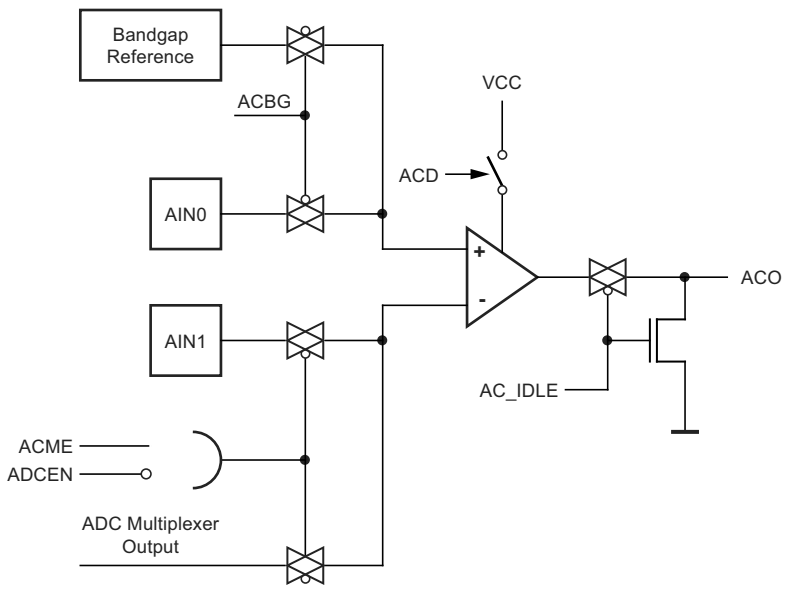


Figure 24-8. General Boundary-scan cell Used for Signals for Comparator and ADC

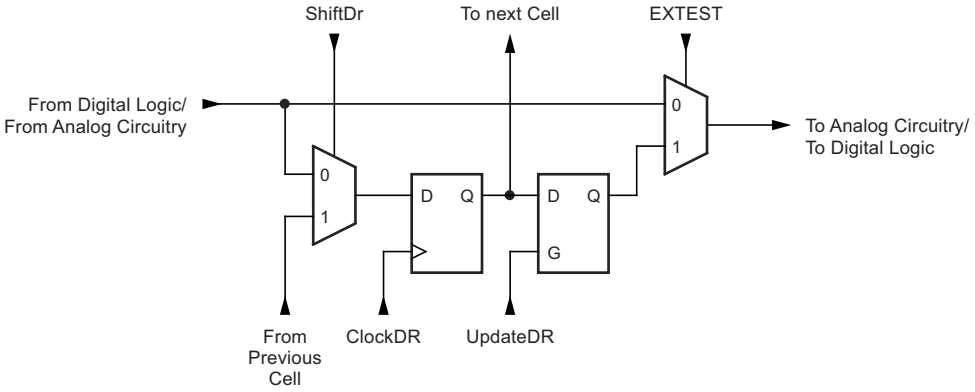


Table 24-2. Boundary-scan Signals for the Analog Comparator

Signal Name	Direction as Seen from the Comparator	Description	Recommended Input when Not in Use	Output Values when Recommended Inputs are Used
AC_IDLE	input	Turns off analog comparator when true	1	Depends upon µC code being executed
ACO	output	Analog comparator output	Will become input to µC code being executed	0
ACME	input	Uses output signal from ADC mux when true	0	Depends upon µC code being executed
ACBG	input	Bandgap reference enable	0	Depends upon µC code being executed

When reading the extended fuse byte, load 0x0002 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SPMEN bits are set in the SPMCSR, the value of the extended fuse byte (EFB) will be loaded in the destination register as shown below. Refer to Table 26-3 on page 251 for detailed description and mapping of the extended fuse byte.

Bit	7	6	5	4	3	2	1	0
Rd	–	–	–	–	**EFB3**	**EFB2**	**EFB1**	**EFB0**

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

25.8.10 Preventing Flash Corruption

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

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

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

1. If there is no need for a boot loader update in the system, program the boot loader lock bits to prevent any boot loader software updates.
2. Keep the AVR® RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal brown-out detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.
3. Keep the AVR core in power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR register and thus the flash from unintentional writes.

25.8.11 Programming Time for Flash when Using SPM

The calibrated RC oscillator is used to time flash accesses. Table 25-5 shows the typical programming time for flash accesses from the CPU.

Table 25-5. SPM Programming Time

Symbol	Min Programming Time	Max Programming Time
Flash write (Page Erase, Page Write, and write Lock bits by SPM)	3.7ms	4.5ms

Table 26-16. Serial Programming Instruction Set (Continued)

Instruction/Operation	Instruction Format			
	Byte 1	Byte 2	Byte 3	Byte4
Write Instructions[6]				
Write program memory page	$4C	adr MSB	adr LSB	$00
Write EEPROM memory	$C0	0000 00aa	aaaa aaaa	data byte in
Write EEPROM memory page (Page access)	$C2	0000 00aa	aaaa aa00	$00
Write lock bits	$AC	$E0	$00	data byte in
Write fuse bits	$AC	$A0	$00	data byte in
Write fuse high bits	$AC	$A8	$00	data byte in
Write extended fuse bits	$AC	$A4	$00	data byte in

Note:
1. Not all instructions are applicable for all parts
2. a = address
3. Bits are programmed ‘0’, unprogrammed ‘1’.
4. To ensure future compatibility, unused fuses and lock bits should be unprogrammed (‘1’).
5. Refer to the correspondig section for fuse and lock bits, calibration and signature bytes and page size.
6. Instructions accessing program memory use a word address. This address may be random within the page range.
7. See htt://www.atmel.com/avr for Application Notes regarding programming and programmers.

If the LSB in RDY/BSY data byte out is ‘1’, a programming operation is still pending. Wait until this bit returns ‘0’ before the next instruction is carried out.

Within the same page, the low data byte must be loaded prior to the high data byte.

After data is loaded to the page buffer, program the EEPROM page, see Figure 26-12 on page 268.

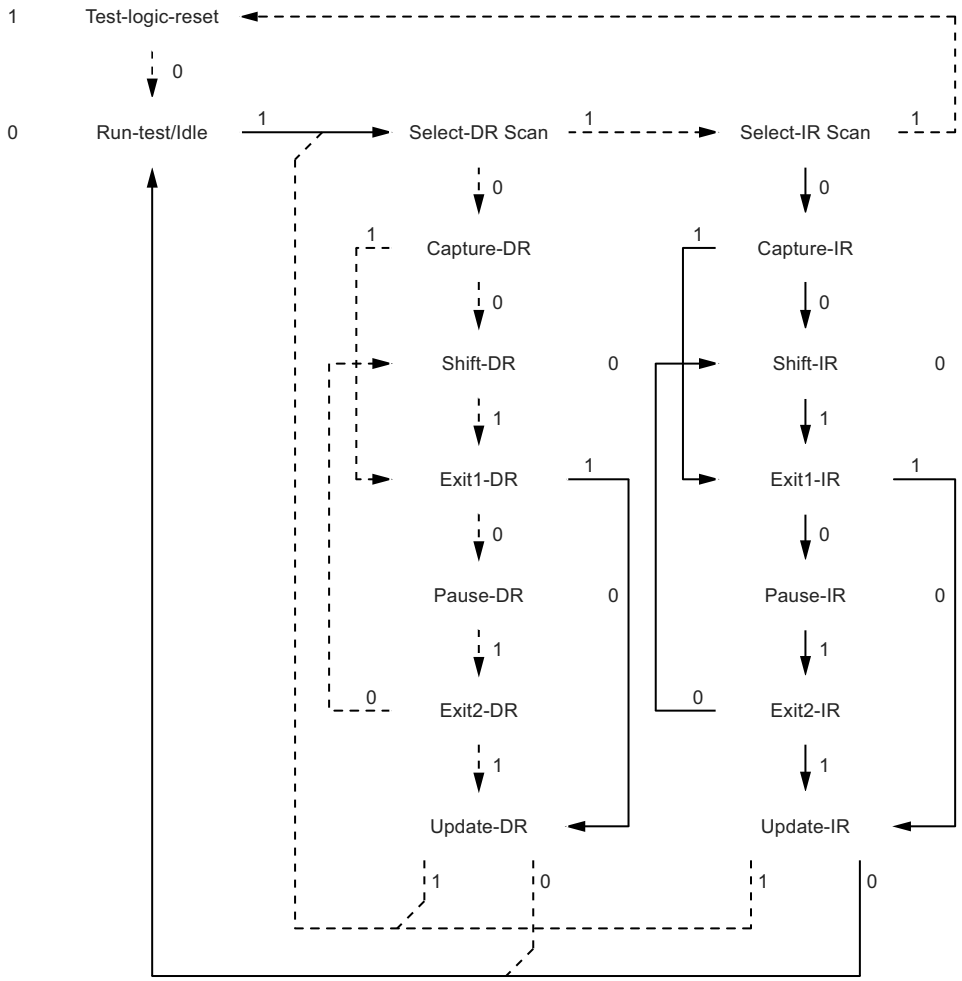
26.9.1 Programming Specific JTAG Instructions

The instruction register is 4-bit wide, supporting up to 16 instructions. The JTAG instructions useful for programming are listed below.

The OPCODE for each instruction is shown behind the instruction name in hex format. The text describes which data register is selected as path between TDI and TDO for each instruction.

The run-test/idle state of the TAP controller is used to generate internal clocks. It can also be used as an idle state between JTAG sequences. The state machine sequence for changing the instruction word is shown in Figure 26-13.

Figure 26-13. State Machine Sequence for Changing the Instruction Word



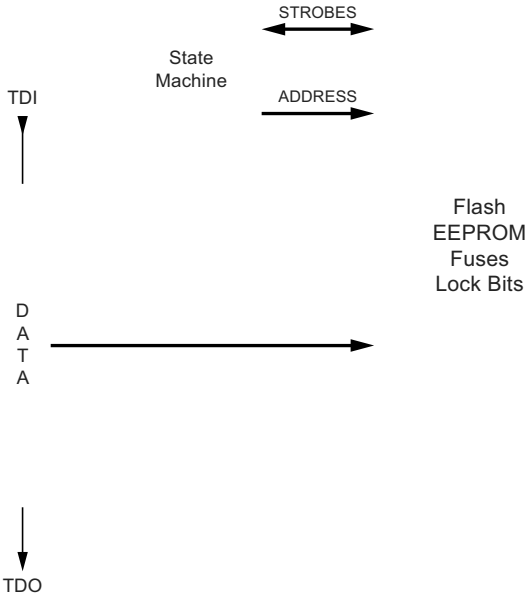
26.9.2 AVR_RESET (0xC)

The AVR® specific public JTAG instruction for setting the AVR device in the reset mode or taking the device out from the reset mode. The TAP controller is not reset by this instruction. The one bit reset register is selected as data register. Note that the reset will be active as long as there is a logic “one” in the reset chain. The output from this chain is not latched.

The active states are:

- Shift-DR: The reset register is shifted by the TCK input.

Figure 26-17. Flash Data Byte Register



The state machine controlling the flash data byte register is clocked by TCK. During normal operation in which eight bits are shifted for each flash byte, the clock cycles needed to navigate through the TAP controller automatically feeds the state machine for the flash data byte register with sufficient number of clock pulses to complete its operation transparently for the user. However, if too few bits are shifted between each update-DR state during page load, the TAP controller should stay in the run-test/idle state for some TCK cycles to ensure that there are at least 11 TCK cycles between each update-DR state.

26.9.12 Programming Algorithm

All references below of type “1a”, “1b”, and so on, refer to Table 26-17 on page 272.

26.9.13 Entering Programming Mode

1. Enter JTAG instruction AVR_RESET and shift 1 in the reset register.
2. Enter instruction PROG_ENABLE and shift 0b1010_0011_0111_0000 in the programming enable register.

26.9.14 Leaving Programming Mode

1. Enter JTAG instruction PROG_COMMANDS.
2. Disable all programming instructions by using no operation instruction 11a.
3. Enter instruction PROG_ENABLE and shift 0b0000_0000_0000_0000 in the programming enable register.
4. Enter JTAG instruction AVR_RESET and shift 0 in the reset register.

26.9.15 Performing Chip Erase

1. Enter JTAG instruction PROG_COMMANDS.
2. Start chip erase using programming instruction 1a.
3. Poll for chip erase complete using programming instruction 1b, or wait for t_{WLRH_CE} (refer to Table 26-13 on page 263).

Figure 27-4. SPI Interface Timing Requirements (Slave Mode)

27.7 ADC Characteristics

Parameter	Condition	Symbol	Min	Typ	Max	Unit
Resolution	Single ended conversion			10		Bits
Absolute accuracy	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200kHz	TUE		2.0	4.0	LSB
Integral non linearity	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200KHz	INL		0.5	1.5	LSB
Differential non linearity	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200kHz	DNL		0.25	0.7	LSB
Gain error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200kHz		–4.0	–2.0	+4.0	LSB
Offset error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200kHz		–4.0	+2.0	+4.0	LSB
Conversion time	Free running conversion		65		260	µs
Clock frequency	Single ended conversion		50		200	kHz
Analog supply voltage		AVCC	V_{CC} – 0.3		V_{CC} + 0.3	V
Reference voltage	Single ended conversion	V_{REF}	1.0		AVCC	V
Pin input voltage	Single ended channels	V_{IN}	GND		V_{REF}	V
Internal voltage reference		V_{INT}	1.0	1.1	1.2	V
Reference input resistance		R_{REF}		32		kΩ
Analog input resistance		R_{AIN}		100		MΩ

27.8 LCD Controller Characteristics

Parameter	Condition	Symbol	Min	Typ	Max	Unit
SEG driver output impedance	V_{LCD} = 5.0V Load = 100µA	R_{SEG}		7	12	kΩ
COM driver output impedance	V_{LCD} = 5.0V Load = 100µA	R_{COM}		1.2	2	kΩ

28. Typical Characteristics

The following charts show typical behavior. These figures are not tested during manufacturing. All current consumption measurements are performed with all I/O pins configured as inputs and with internal pull-ups enabled. A sine wave generator with rail-to-rail output is used as clock source.

All active and Idle current consumption measurements are done with all bits in the PRR register set and thus, the corresponding I/O modules are turned off. Also the Analog Comparator is disabled during these measurements. Table 28-1 shows the additional current consumption compared to I_{CC} Active and I_{CC} Idle for every I/O module controlled by the Power Reduction Register. See Section 8.7 “Power Reduction Register” on page 35 for details.

The power consumption in Power-down mode is independent of clock selection.

The current consumption is a function of several factors such as: operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature. The dominating factors are operating voltage and frequency.

The current drawn from capacitive loaded pins may be estimated (for one pin) as $C_L \times V_{CC} \times f$ where C_L = load capacitance, V_{CC} = operating voltage and f = average switching frequency of I/O pin.

The parts are characterized at frequencies higher than test limits. Parts are not guaranteed to function properly at frequencies higher than the ordering code indicates.

The difference between current consumption in Power-down mode with watchdog timer enabled and power-down mode with watchdog timer disabled represents the differential current drawn by the watchdog timer.

28.1 Active Supply Current

Figure 28-1. Active Supply Current versus Frequency (0.1 - 1.0MHz)

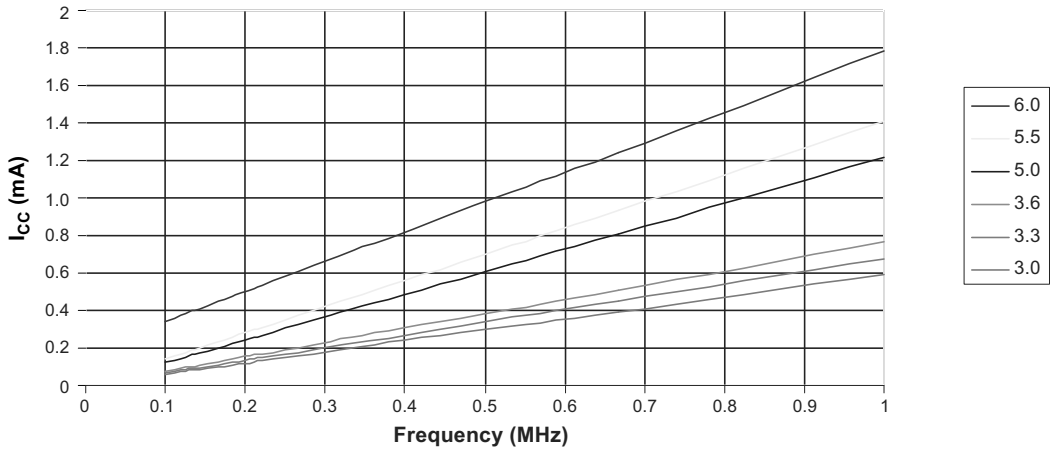


Figure 28-2. Active Supply Current versus Frequency (1 - 16MHz)

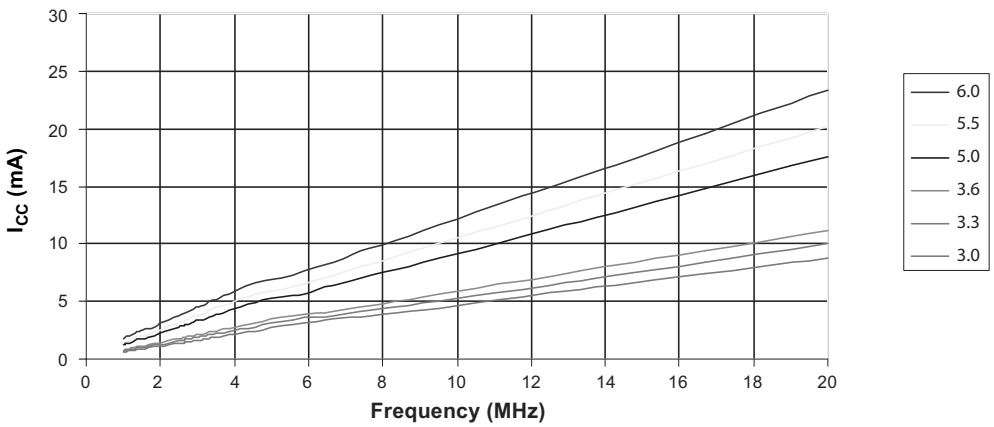


Figure 28-20. Output Low Voltage Port B - V_{CC} = 5V

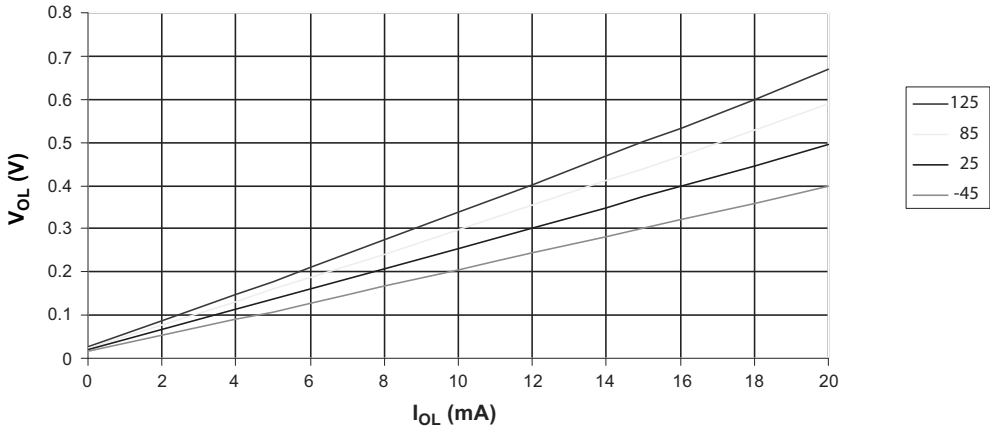


Figure 28-21. Output Low Voltage Port B - V_{CC} = 3V

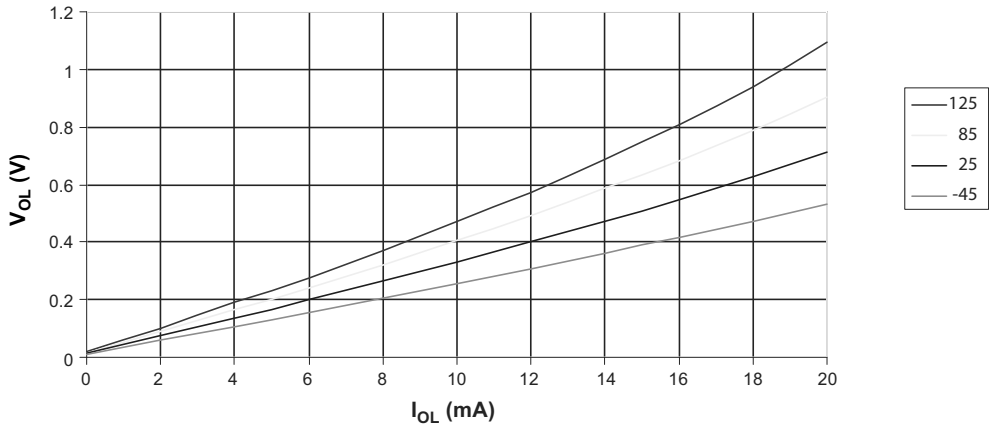


Figure 28-22. Output High Voltage Ports A, C, D, E, F, G - V_{CC} = 5V