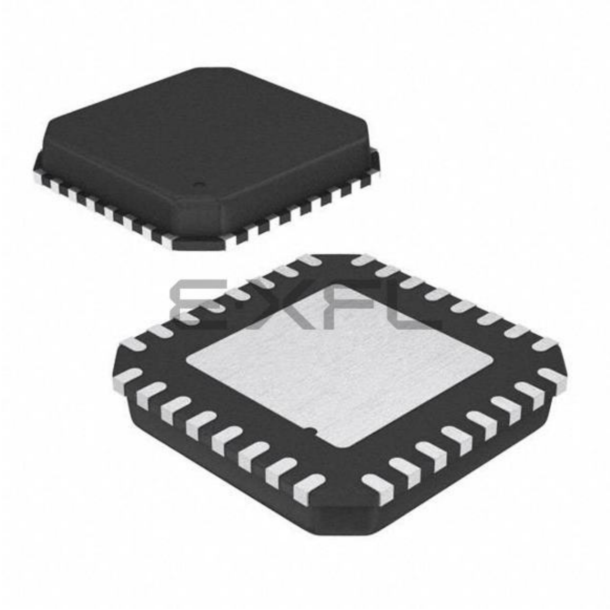
Microchip Technology - ATMEGA8A-MN Datasheet



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"

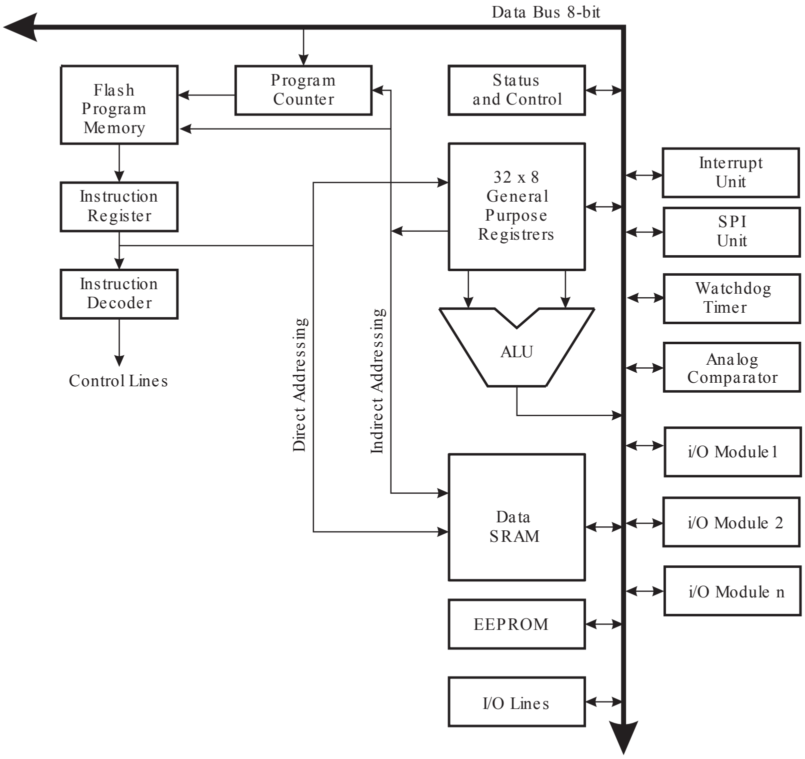
Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	23
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-VQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega8a-mn

11. AVR CPU Core

11.1. Overview

This section discusses the Atmel AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Figure 11-1 Block Diagram of the AVR MCU Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the Program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the Program memory. This concept enables instructions to be executed in every clock cycle. The Program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

11.5.1. SPH and SPL - Stack Pointer High and Stack Pointer Low Register

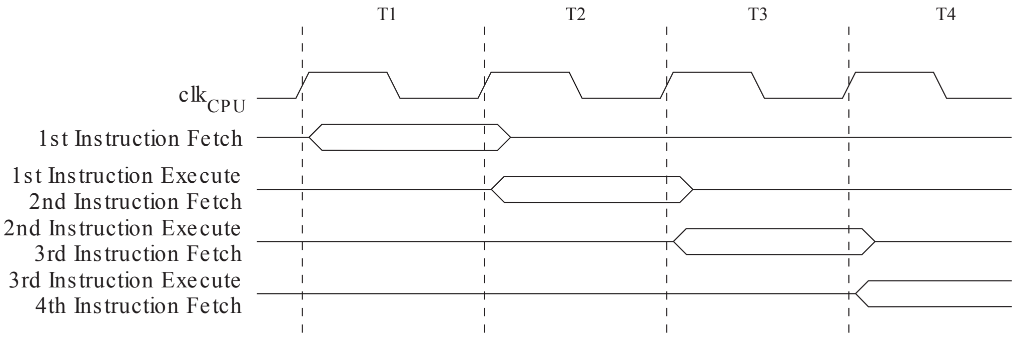
Bit	15	14	13	12	11	10	9	8	
0x3E	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
0x3D	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

11.6. Instruction Execution Timing

This section describes the general access timing concepts for instruction execution. The Atmel AVR CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

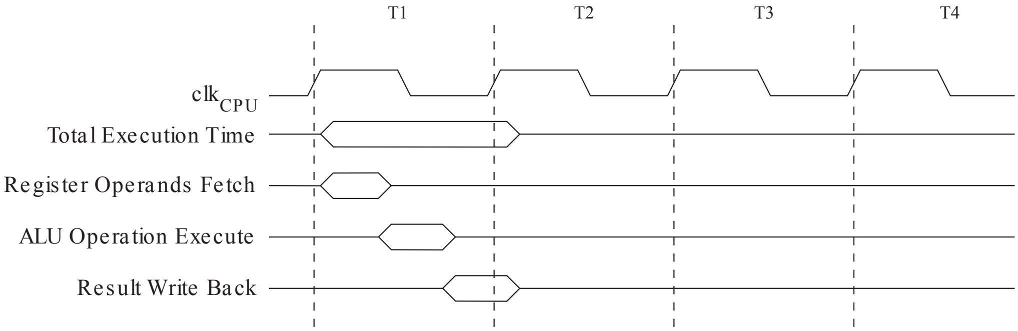
The following figure shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access Register File concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 11-4 The Parallel Instruction Fetches and Instruction Executions



The next figure shows the internal timing concept for the Register File. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 11-5 Single Cycle ALU Operation



CKOPT Fuse should always be unprogrammed when using this clock option. During reset, hardware loads the 1MHz calibration byte into the OSCCAL Register and thereby automatically calibrates the RC Oscillator. At 5V, 25°C and 1.0MHz Oscillator frequency selected, this calibration gives a frequency within ± 3% of the nominal frequency. Using run-time calibration methods as described in application notes available at www.atmel.com/avr it is possible to achieve ± 1% accuracy at any given V_{CC} and Temperature. When this Oscillator is used as the chip clock, the Watchdog Oscillator will still be used for the Watchdog Timer and for the Reset Time-out. For more information on the pre-programmed calibration value, see the section *Calibration Byte*.

Table 13-8 Internal Calibrated RC Oscillator Operating Modes

CKSEL3:0	Nominal Frequency (MHz)
0001(1)	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected.

When this Oscillator is selected, start-up times are determined by the SUT Fuses as shown in the table below. PB6 (XTAL1/TOSC1) and PB7(XTAL2/TOSC2) can be used as either general I/O pins or Timer Oscillator pins:

Table 13-9 Start-up Times for the Internal Calibrated RC Oscillator Clock Selection

SUT1:0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	6 CK	–	BOD enabled
01	6 CK	4.1ms	Fast rising power
10(1)	6 CK	65ms	Slowly rising power
11	Reserved		

Note: 1. The device is shipped with this option selected.

Related Links

Calibration Byte on page 286

13.7. External Clock

To drive the device from an external clock source, XTAL1 should be driven as shown in the figure below. To run the device on an external clock, the CKSEL Fuses must be programmed to “0000”. By programming the CKOPT Fuse, the user can enable an internal 36pF capacitor between XTAL1 and GND, and XTAL2 and GND.

14. Power Management and Sleep Modes

14.1. Sleep Modes

Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the application's requirements.

Figure *Clock Distribution* in section *Clock Systems and their Distribution* presents the different clock systems in the ATmega8A, and their distribution. The figure is helpful in selecting an appropriate sleep mode. The table below shows the different clock options and their wake-up sources.

Table 14-1 Active Clock Domains and Wake-up Sources in the Different Sleep Modes

	Active Clock Domains					Oscillators		Wake-up Sources					
Sleep Mode	**clk_{CPU}**	**clk_{FLASH}**	**clk_{IO}**	**clk_{ADC}**	**clk_{ASY}**	**Main Clock Source Enabled**	**Timer Osc. Enabled**	**INT1/ INT0**	**TWIAddress Match**	**Timer2**	**SPM/ EEPROM Ready**	**ADC**	**Other I/O**
Idle			X	X	X	X	X[2]	X	X	X	X	X	X
ADC Noise Reduction				X	X	X	X[2]	X[3]	X	X	X	X	
Power-down								X[3]	X				
Power-save					X[2]		X[2]	X[3]	X	X[2]			
Standby[1]						X		X[3]	X				

Note:

1. External Crystal or resonator selected as clock source.
2. If AS2 bit in ASSR is set.
3. Only level interrupt INT1 and INT0.

To enter any of the five sleep modes, the SE bit in MCUCR must be written to logic one and a SLEEP instruction must be executed. The SM2, SM1, and SM0 bits in the MCUCR Register select which sleep mode (Idle, ADC Noise Reduction, Power-down, Power-save, or Standby) will be activated by the SLEEP instruction. See the table above for a summary.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, it executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the Register File and SRAM are unaltered when the device wakes up from sleep. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset Vector.

Note that the Extended Standby mode present in many other AVR MCUs has been removed in the ATmega8A, as the TOSC and XTAL inputs share the same physical pins.

Related Links

Clock Systems and their Distribution on page 44

14.8.1. MCUCR – MCU Control Register

The MCU Control Register contains control bits for power management.

When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: MCUCR
Offset: 0x35
Reset: 0x00
Property: When addressing I/O Registers as data space the offset address is 0x55

Bit	7	6	5	4	3	2	1	0
			SE	SM2	SM1	SM0		
Access			R/W	R/W	R/W	R/W		
Reset			0	0	0	0		

Bit 5 – SE: Sleep Enable
The SE bit must be written to logic one to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode unless it is the programmer's purpose, it is recommended to set the Sleep Enable (SE) bit just before the execution of the SLEEP instruction.

Bits 4:2 – SMn: Sleep Mode n Select Bits [n=2:0]
These bits select between the five available sleep modes as shown in the table.

SM2	SM1	SM0	Sleep Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby[1]

Note: 1. Standby mode is only available with external crystals or resonators.

Signal Name	PB3/MOSI/ OC2	PB2/$\overline{\text{SS}}$/ OC1B	PB1/OC1A	PB0/ICP1
DDOV	0	0	0	0
PVOE	SPE • MSTR + OC2 ENABLE	OC1B ENABLE	OC1A ENABLE	0
PVOV	SPI MSTR OUTPUT + OC2	OC1B	OC1A	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	SPI SLAVE INPUT	SPI $\overline{\text{SS}}$	–	ICP1 INPUT
AIO	–	–	–	–

18.3.2. Alternate Functions of Port C

The Port C pins with alternate functions are shown in the table below:

Table 18-6 Port C Pins Alternate Functions

Port Pin	Alternate Function
PC6	$\overline{\text{RESET}}$ (Reset pin)
PC5	ADC5 (ADC Input Channel 5) SCL (Two-wire Serial Bus Clock Line)
PC4	ADC4 (ADC Input Channel 4) SDA (Two-wire Serial Bus Data Input/Output Line)
PC3	ADC3 (ADC Input Channel 3)
PC2	ADC2 (ADC Input Channel 2)
PC1	ADC1 (ADC Input Channel 1)
PC0	ADC0 (ADC Input Channel 0)

The alternate pin configuration is as follows:

- **$\overline{\text{RESET}}$ – Port C, Bit 6**

$\overline{\text{RESET}}$, Reset pin: When the RSTDISBL Fuse is programmed, this pin functions as a normal I/O pin, and the part will have to rely on Power-on Reset and Brown-out Reset as its reset sources. When the RSTDISBL Fuse is unprogrammed, the reset circuitry is connected to the pin, and the pin can not be used as an I/O pin.

If PC6 is used as a reset pin, DDC6, PORTC6 and PINC6 will all read 0.

- **SCL/ADC5 – Port C, Bit 5**

SCL, Two-wire Serial Interface Clock: When the TWEN bit in TWCR is set (one) to enable the Two-wire Serial Interface, pin PC5 is disconnected from the port and becomes the Serial Clock I/O pin for the Two-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC5 can also be used as ADC input Channel 5. Note that ADC input channel 5 uses digital power.

- **SDA/ADC4 – Port C, Bit 4**

18.4.5. PORTC – The Port C Data Register

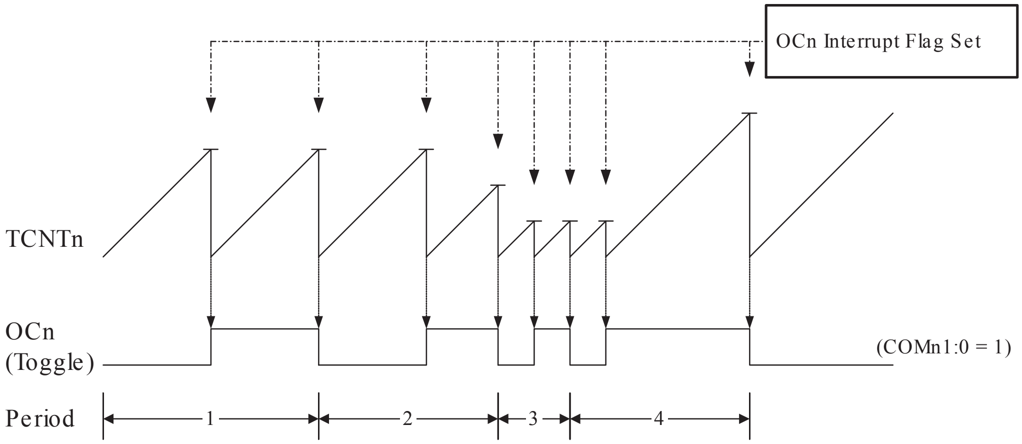
When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: PORTC
Offset: 0x15
Reset: 0x00
Property: When addressing I/O Registers as data space the offset address is 0x35

Bit	7	6	5	4	3	2	1	0
		PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 6:0 – PORTCn: Port C Data [n = 6:0]

Figure 22-5 CTC Mode, Timing Diagram



An interrupt can be generated each time the counter value reaches the TOP value by using the OCF2 Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing the TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR2 is lower than the current value of TCNT2, the counter will miss the Compare Match. The counter will then have to count to its maximum value (0xFF) and wrap around starting at 0x00 before the Compare Match can occur.

For generating a waveform output in CTC mode, the OC2 output can be set to toggle its logical level on each Compare Match by setting the Compare Output mode bits to toggle mode (COM21:0 = 1). The OC2 value will not be visible on the port pin unless the data direction for the pin is set to output. The waveform generated will have a maximum frequency of $f_{OC2} = f_{clk\_I/O}/2$ when OCR2 is set to zero (0x00). The waveform frequency is defined by the following equation:

$$f_{\mathrm{OCn}} = \frac{f_{\mathrm{clk\_I/O}}}{2 \cdot N \cdot (1 + \mathrm{OCRn})}$$

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

As for the Normal mode of operation, the TOV2 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x00.

22.7.3. Fast PWM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM21:0 = 3) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to MAX then restarts from BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC2) is cleared on the Compare Match between TCNT2 and OCR2, and set at BOTTOM. In inverting Compare Output mode, the output is set on Compare Match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct PWM mode that uses dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the MAX value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in the following figure. 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

Bit 3 – WGM21: Waveform Generation Mode [n=0:1]
Refer to WGM20.

Bits 2:0 – CS2n: Clock Select [n = 2:0]
The three Clock Select bits select the clock source to be used by the Timer/Counter.

Table 22-6 Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/32 (From prescaler)
1	0	0	clkI/O/64 (From prescaler)
1	0	1	$clk_{I/O}$/128 (From prescaler)
1	1	0	$clk_{I/O}$/256 (From prescaler)
1	1	1	$clk_{I/O}$/1024 (From prescaler)

22.11.4. ASSR – Asynchronous Status Register

When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: ASSR
Offset: 0x22
Reset: 0x00
Property: When addressing I/O Registers as data space the offset address is 0x42

Bit	7	6	5	4	3	2	1	0
					AS2	TCN2UB	OCR2UB	TCR2UB
Access					R/W	R	R	R
Reset					0	0	0	0

Bit 3 – AS2: Asynchronous Timer/Counter2

When AS2 is written to zero, Timer/Counter 2 is clocked from the I/O clock, $clk_{I/O}$. When AS2 is written to one, Timer/Counter 2 is clocked from a crystal Oscillator connected to the Timer Oscillator 1 (TOSC1) pin. When the value of AS2 is changed, the contents of TCNT2, OCR2, and TCCR2 might be corrupted.

Bit 2 – TCN2UB: Timer/Counter2 Update Busy

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

Bit 1 – OCR2UB: Output Compare Register2 Update Busy

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

Bit 0 – TCR2UB: Timer/Counter Control Register2 Update Busy

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

If a write is performed to any of the three Timer/Counter2 Registers while its update busy flag is set, 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.

Assembly Code Example[1]

```
USART_Transmit:
   ; Wait for empty transmit buffer
   sbis    UCSRA,UDRE
   rjmp    USART_Transmit
   ; Put data (r16) into buffer, sends the data
   out    UDR,r16
   ret
```

C Code Example[1]

```
void USART_Transmit( unsigned char data )
{
   /* Wait for empty transmit buffer */
   while ( !( UCSRA & (1<<UDRE)) )
      ;
   /* Put data into buffer, sends the data */
   UDR = data;
}
```

Note: 1. See *About Code Examples*.

The function simply waits for the transmit buffer to be empty by checking the UDRE Flag, before loading it with new data to be transmitted. If the Data Register Empty Interrupt is utilized, the interrupt routine writes the data into the buffer.

Related Links

About Code Examples on page 23

24.6.2. Sending Frames with 9 Data Bits

If 9-bit characters are used (UCSZ = 7), the ninth bit must be written to the TXB8 bit in UCSRB before the Low byte of the character is written to UDR. The following code examples show a transmit function that handles 9-bit characters. For the assembly code, the data to be sent is assumed to be stored in registers R17:R16.

Assembly Code Example[1]

```
USART_Transmit:
   ; Wait for empty transmit buffer
   sbis    UCSRA,UDRE
   rjmp    USART_Transmit
   ; Copy 9th bit from r17 to TXB8
   cbi     UCSRB,TXB8
   sbrc    r17,0
   sbi     UCSRB,TXB8
   ; Put LSB data (r16) into buffer, sends the data
   out    UDR,r16
   ret
```

C Code Example[1]

```
void USART_Transmit( unsigned int data )
{
   /* Wait for empty transmit buffer */
   while ( !( UCSRA & (1<<UDRE))) )
      ;
   /* Copy 9th bit to TXB8 */
```

receive buffer FIFO and consequently the TXB8, FE, DOR, and PE bits, which all are stored in the FIFO, will change.

The following code example shows a simple USART receive function that handles both 9-bit characters and the status bits.

Assembly Code Example[(1)]

```
USART_Receive:
   ; Wait for data to be received
   sbis   r16, RXC
   rjmp   USART_Receive
   ; Get status and 9th bit, then data from buffer
   in     r18, UCSRA
   in     r17, UCSRB
   in     r16, UDR
   ; If error, return -1
   andi   r18,(1<<FE)|(1<<DOR)|(1<<PE)
   breq   USART_ReceiveNoError
   ldi    r17, HIGH(-1)
   ldi    r16, LOW(-1)
USART_ReceiveNoError:
   ; Filter the 9th bit, then return
   lsr    r17
   andi   r17, 0x01
   ret
```

C Code Example[(1)]

```
unsigned int USART_Receive( void )
{
   unsigned char status, resh, resl;
   /* Wait for data to be received */
   while ( !(UCSRA & (1<<RXC)) )
      ;
   /* Get status and 9th bit, then data */
   /* from buffer */
   status = UCSRA;
   resh = UCSRB;
   resl = UDR;
   /* If error, return -1 */
   if ( status & (1<<FE)|(1<<DOR)|(1<<PE) )
      return -1;
   /* Filter the 9th bit, then return */
   resh = (resh >> 1) & 0x01;
   return ((resh << 8) | resl);
}
```

Note: 1. See *About Code Examples.*

The receive function example reads all the I/O Registers into the Register File before any computation is done. This gives an optimal receive buffer utilization since the buffer location read will be free to accept new data as early as possible.

Related Links

About Code Examples on page 23

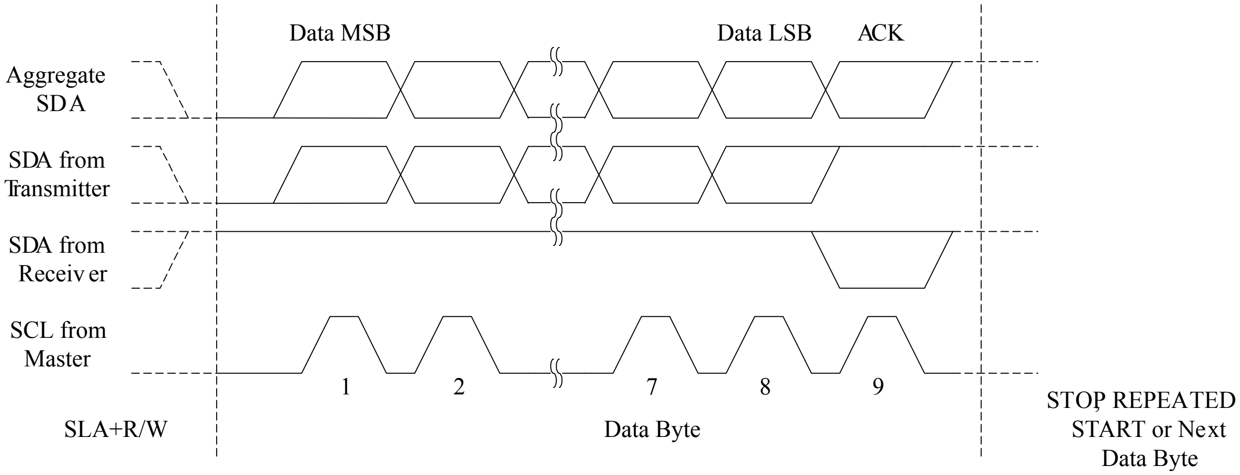
24.7.3. Receive Compete Flag and Interrupt

The USART Receiver has one flag that indicates the Receiver state.

25.4.4. Data Packet Format

All data packets transmitted on the TWI bus are nine bits long, consisting of one data byte and an acknowledge bit. During a data transfer, the Master generates the clock and the START and STOP conditions, while the Receiver is responsible for acknowledging the reception. An Acknowledge (ACK) is signalled by the Receiver pulling the SDA line low during the ninth SCL cycle. If the Receiver leaves the SDA line high, a NACK is signalled. When the Receiver has received the last byte, or for some reason cannot receive any more bytes, it should inform the Transmitter by sending a NACK after the final byte. The MSB of the data byte is transmitted first.

Figure 25-6 Data Packet Format

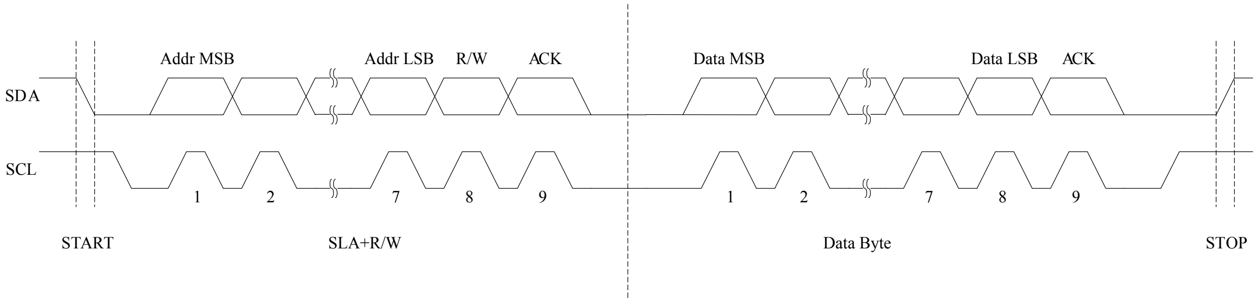


25.4.5. Combining Address and Data Packets Into a Transmission

A transmission basically consists of a START condition, a SLA+R/W, one or more data packets and a STOP condition. An empty message, consisting of a START followed by a STOP condition, is illegal. Note that the Wired-ANDing of the SCL line can be used to implement handshaking between the Master and the Slave. The Slave can extend the SCL low period by pulling the SCL line low. This is useful if the clock speed set up by the Master is too fast for the Slave, or the Slave needs extra time for processing between the data transmissions. The Slave extending the SCL low period will not affect the SCL high period, which is determined by the Master. As a consequence, the Slave can reduce the TWI data transfer speed by prolonging the SCL duty cycle.

The following figure depicts a typical data transmission. Note that several data bytes can be transmitted between the SLA+R/W and the STOP condition, depending on the software protocol implemented by the application software.

Figure 25-7 Typical Data Transmission



26.3.2. ACSR – Analog Comparator Control and Status Register

When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: ACSR
Offset: 0x08
Reset: N/A
Property: When addressing I/O Registers as data space the offset address is 0x28

Bit	7	6	5	4	3	2	1	0
	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
Access	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
Reset	0	0	a	0	0	0	0	0

Bit 7 – ACD: Analog Comparator Disable

When this bit is written logic one, the power to the Analog Comparator is switched off. This bit can be set at any time to turn off the Analog Comparator. This will reduce power consumption in Active and Idle mode. When changing the ACD bit, the Analog Comparator Interrupt must be disabled by clearing the ACIE bit in ACSR. Otherwise an interrupt can occur when the bit is changed.

Bit 6 – ACBG: Analog Comparator Bandgap Select

When this bit is set, a fixed bandgap reference voltage replaces the positive input to the Analog Comparator. When this bit is cleared, AIN0 is applied to the positive input of the Analog Comparator. Refer to Internal Voltage Reference in System Control and Reset.

Bit 5 – ACO: Analog Comparator Output

The output of the Analog Comparator is synchronized and then directly connected to ACO. The synchronization introduces a delay of 1 - 2 clock cycles.

Bit 4 – ACI: Analog Comparator Interrupt Flag

This bit is set by hardware when a comparator output event triggers the interrupt mode defined by ACIS1 and ACIS0. The Analog Comparator interrupt routine is executed if the ACIE bit is set and the I-bit in SREG is set. ACI is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ACI is cleared by writing a logic one to the flag.

Bit 3 – ACIE: Analog Comparator Interrupt Enable

When the ACIE bit is written logic one and the I-bit in the Status Register is set, the Analog Comparator interrupt is activated. When written logic zero, the interrupt is disabled.

Bit 2 – ACIC: Analog Comparator Input Capture Enable

When written logic one, this bit enables the input capture function in Timer/Counter1 to be triggered by the Analog Comparator. The comparator output is in this case directly connected to the input capture front-end logic, making the comparator utilize the noise canceler and edge select features of the Timer/Counter1 Input Capture interrupt. When written logic zero, no connection between the Analog Comparator and the input capture function exists. To make the comparator trigger the Timer/Counter1 Input Capture interrupt, the ICIE1 bit in the Timer Interrupt Mask Register (TIMSK1) must be set.

27.8.6. ADCH – ADC Data Register High (ADLAR=1)

When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these offset addresses. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

Name: ADCH
Offset: 0x05
Reset: 0x00
Property: When addressing I/O Registers as data space the offset address is 0x25

Bit	7	6	5	4	3	2	1	0
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – ADCn: ADC Conversion Result [n = 7:0]
Refer to ADCL.

High Fuse Byte	Bit No.	Description	Default Value
BOOTSZ1	2	Select Boot Size (see *ATmega8A Boot Loader Parameters*)	0 (programmed)(3)
BOOTSZ0	1	Select Boot Size (see *ATmega8A Boot Loader Parameters*)	0 (programmed)(3)
BOOTRST	0	Select Reset Vector	1 (unprogrammed)

Note:

1. The SPIEN Fuse is not accessible in Serial Programming mode.
2. The CKOPT Fuse functionality depends on the setting of the CKSEL bits, see *Clock Sources* for details.
3. The default value of BOOTSZ1:0 results in maximum Boot Size. See *Boot Loader Parameters* for details.
4. When programming the RSTDISBL Fuse Parallel Programming has to be used to change fuses or perform further programming.

Table 29-4 Fuse Low Byte

Low Fuse Byte	Bit No.	Description	Default Value
BODLEVEL	7	Brown out detector trigger level	1 (unprogrammed)
BODEN	6	Brown out detector enable	1 (unprogrammed, BOD disabled)
SUT1	5	Select start-up time	1 (unprogrammed)(1)
SUT0	4	Select start-up time	0 (programmed)(1)
CKSEL3	3	Select Clock source	0 (programmed)(2)
CKSEL2	2	Select Clock source	0 (programmed)(2)
CKSEL1	1	Select Clock source	0 (programmed)(2)
CKSEL0	0	Select Clock source	1 (unprogrammed)(2)

Note:

1. The default value of SUT1:0 results in maximum start-up time. See table *Start-up Times for the Internal Calibrated RC Oscillator Clock Selection* in *Calibrated Internal RC Oscillator* of the System Clock and Clock Options chapter for details.
2. The default setting of CKSEL3:0 results in internal RC Oscillator @ 1MHz. See table *Internal Calibrated RC Oscillator Operating Modes* in *Calibrated Internal RC Oscillator* of the System Clock and Clock Options chapter for details.

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

Related Links

Clock Sources on page 45

ATmega8A Boot Loader Parameters on page 278

Calibrated Internal RC Oscillator on page 48

31. Electrical Characteristics – TA = -40°C to 105°C

Absolute Maximum Ratings*

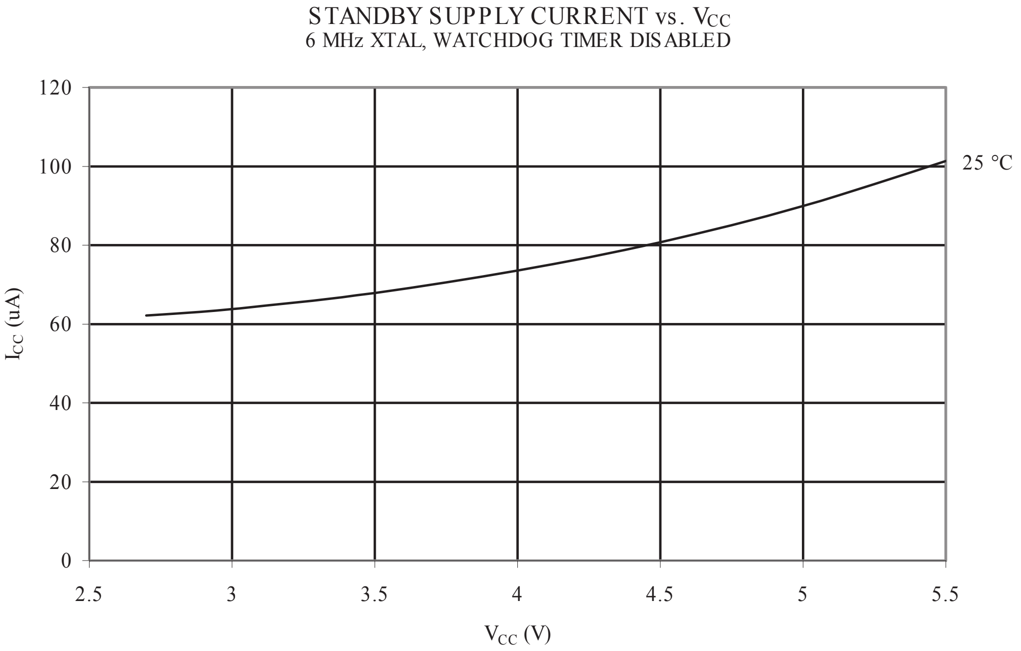
Operating Temperature	-55°C to +125°C	*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.
Storage Temperature	-65°C to +150°C	
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC} +0.5V	
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V	
Maximum Operating Voltage	6.0V	
DC Current per I/O Pin	40.0mA	
DC Current V_{CC} and GND Pins	200.0mA	

31.1. DC Characteristics

Table 31-1 T_A = -40°C to 105°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{IL}	Input Low Voltage, Except XTAL1 and $\overline{\text{RESET}}$ pin	V_{CC} = 2.7V - 5.5V	-0.5		$0.2V_{CC}$(1)	V
V_{IL1}	Input Low Voltage, XTAL1 pin	V_{CC} = 2.7V - 5.5V	-0.5		$0.1V_{CC}$(1)	V
V_{IL2}	Input Low Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 2.7V - 5.5V	-0.5		$0.1V_{CC}$(1)	V
V_{IH}	Input High Voltage, Except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 2.7V - 5.5V	$0.6V_{CC}$(2)		V_{CC} + 0.5	V
V_{IH1}	Input High Voltage, XTAL1 pin	V_{CC} = 2.7V - 5.5V	$0.8V_{CC}$(2)		V_{CC} + 0.5	V
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 2.7V - 5.5V	$0.9V_{CC}$(2)		V_{CC} + 0.5	V
V_{OL}	Output Low Voltage(3), Port B (except $\overline{\text{RESET}}$)	I_{OL} =20mA, V_{CC} = 5V I_{OL} =10mA, V_{CC} = 3V			0.8 0.6	V
V_{OH}	Output High Voltage(4), Port B (except $\overline{\text{RESET}}$)	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.0 2.2			V
I_{IL}	Input Leakage Current I/O Pin				3	µA

Figure 32-23 Standby Supply Current vs. V_{CC} (6MHz Xtal, Watchdog Timer Disabled)



32.6. Pin Pull-up

Figure 32-24 I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5V)

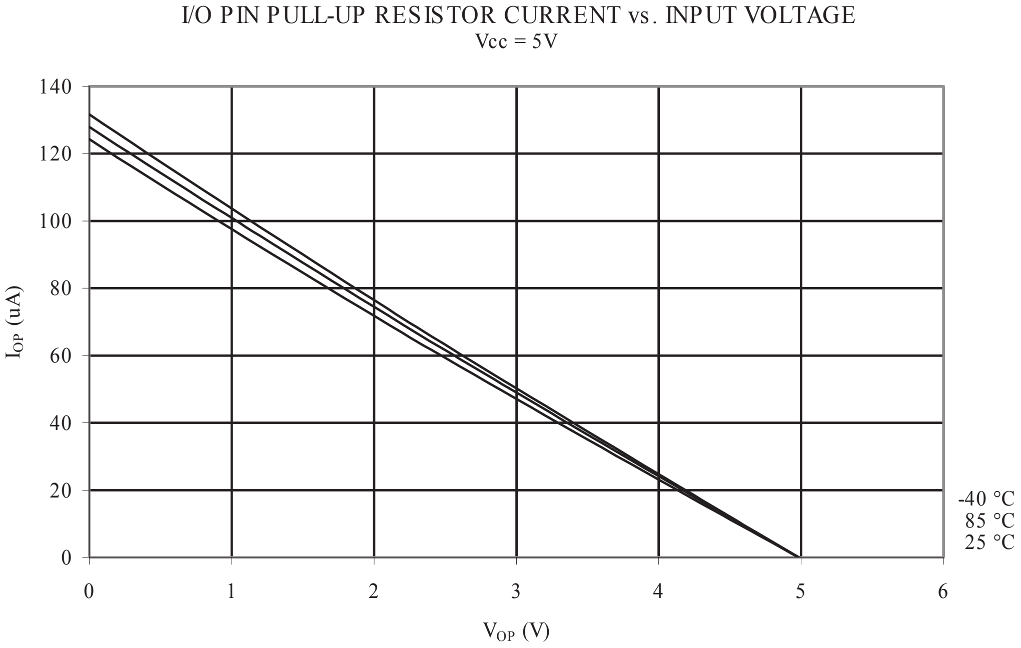


Figure 32-71 Reset Supply Current vs. V_{CC} (1 - 16MHz, Excluding Current Through The Reset Pull-up)

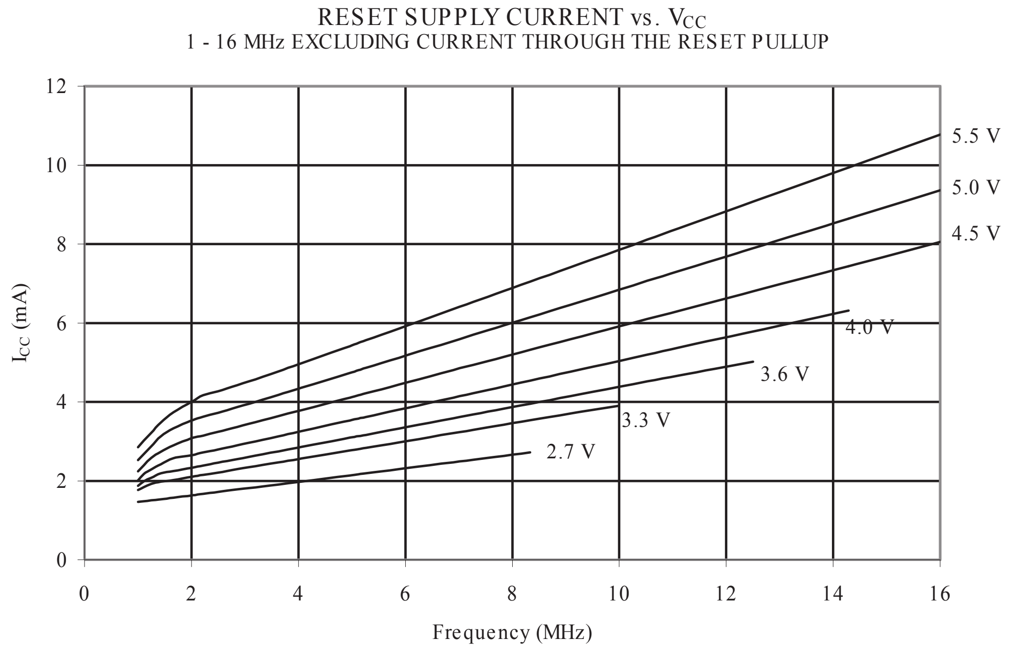
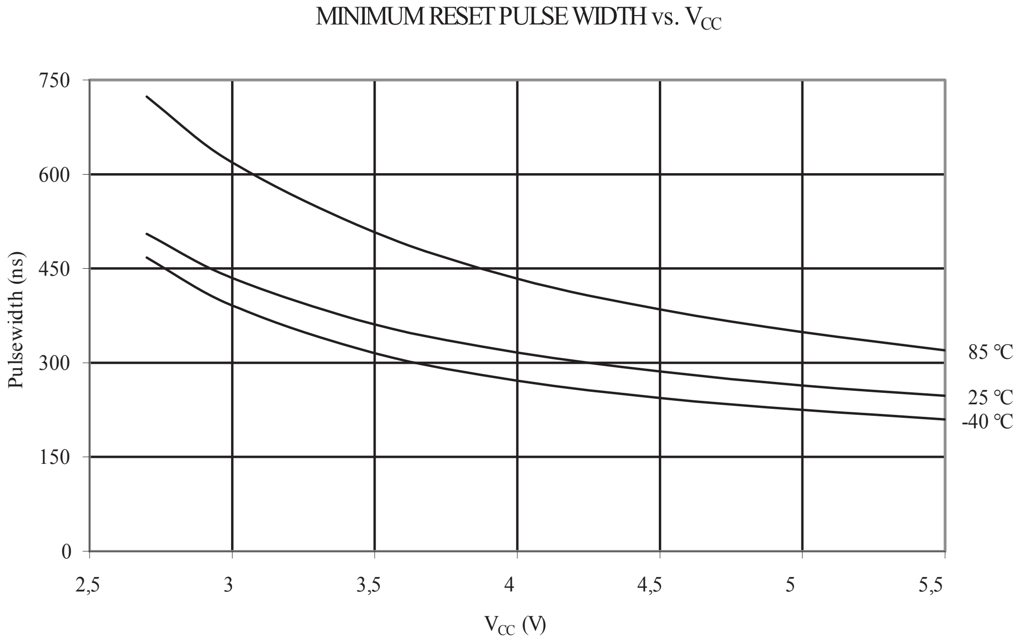


Figure 32-72 Reset Pulse Width vs. V_{CC}



33.1.2. Idle Supply Current

Figure 33-6 Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 8MHz)

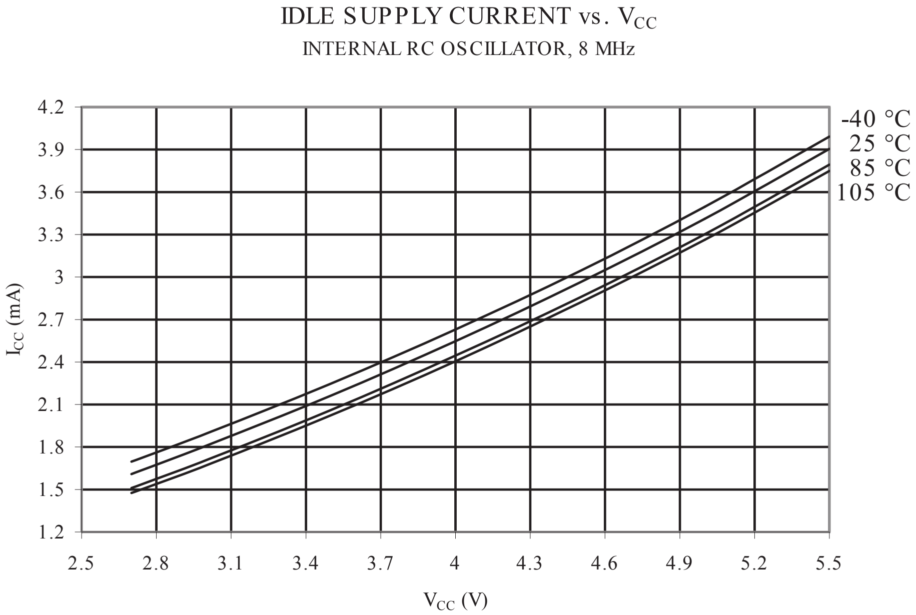
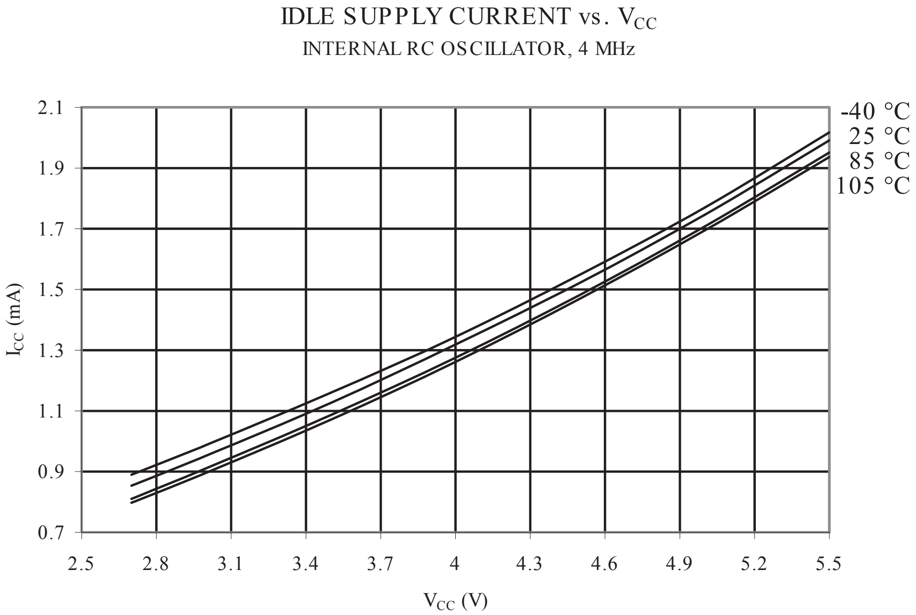


Figure 33-7 Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 4MHz)


IDLE SUPPLY CURRENT vs. VCC
INTERNAL RC OSCILLATOR, 4 MHz
ICC (mA)
VCC (V)
-40 °C
25 °C
85 °C
105 °C