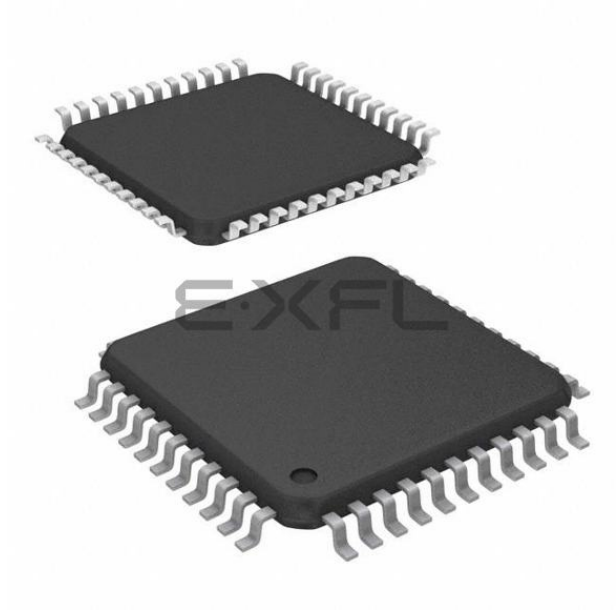


Welcome to E-XFL.COM

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	20MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	32
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K 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	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega324p-20aq

1. Refer to *Data Retention*.

Related Links

Data Retention on page 20

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.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

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

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F. In addition, this device has Extended I/O space from 0x60 - 0xFF in SRAM where only the ST/STS/STD and LD/LDS/LDD instructions can be used.

8.2. ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See *Instruction Set Summary* section for a detailed description.

Related Links

Instruction Set Summary on page 435

8.3. Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. The Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

10.1.1. CPU Clock – clk_{CPU}

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the data memory holding the Stack Pointer. Halting the CPU clock inhibits the core from performing general operations and calculations.

10.1.2. I/O Clock – $clk_{I/O}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counters, SPI, and USART. The I/O clock is also used by the External Interrupt module, but the start condition detection in the USI module is carried out asynchronously when $clk_{I/O}$ is halted, TWI address recognition in all sleep modes.

Note: If a level triggered interrupt is used for wake-up from Power-down, the required level must be held long enough for the MCU to complete the wake-up to trigger the level interrupt. If the level disappears before the end of the Start-up Time, the MCU will still wake up, but no interrupt will be generated. The start-up time is defined by the SUT and CKSEL Fuses.

10.1.3. Flash Clock – clk_{FLASH}

The Flash clock controls operation of the Flash interface. The Flash clock is usually active simultaneously with the CPU clock.

10.1.4. Asynchronous Timer Clock – clk_{ASY}

The Asynchronous Timer clock allows Asynchronous Timer/Counters to be clocked directly from an external clock or an external 32kHz clock crystal. The dedicated clock domain allows using this Timer/Counter as a real-time counter even when the device is in sleep mode.

10.1.5. ADC Clock – clk_{ADC}

The ADC is provided with a dedicated clock domain. This allows halting the CPU and I/O clocks in order to reduce noise generated by digital circuitry. This gives more accurate ADC conversion results.

10.2. Clock Sources

The device has the following clock source options, selectable by Flash Fuse bits as shown below. The clock from the selected source is input to the AVR clock generator, and routed to the appropriate modules.

Table 10-1. Device Clocking Options Select

Device Clocking Option	CKSEL[3:0]
Low Power Crystal Oscillator	1111 - 1000
Full Swing Crystal Oscillator	0111 - 0110
Low Frequency Crystal Oscillator	0101 - 0100
Internal 128kHz RC Oscillator	0011
Calibrated Internal RC Oscillator	0010
External Clock	0000
Reserved	0001

Note: For all fuses, '1' means unprogrammed while '0' means programmed.

13. Interrupts

13.1. Overview

This section describes the specifics of the interrupt handling of the device. For a general explanation of the AVR interrupt handling, refer to the description of *Reset and Interrupt Handling*.

In general:

- Each Interrupt Vector occupies two instruction words.
- The Reset Vector is affected by the BOOTRST fuse, and the Interrupt Vector start address is affected by the IVSEL bit in MCUCR (MCUCR.IVSEL)

Related Links

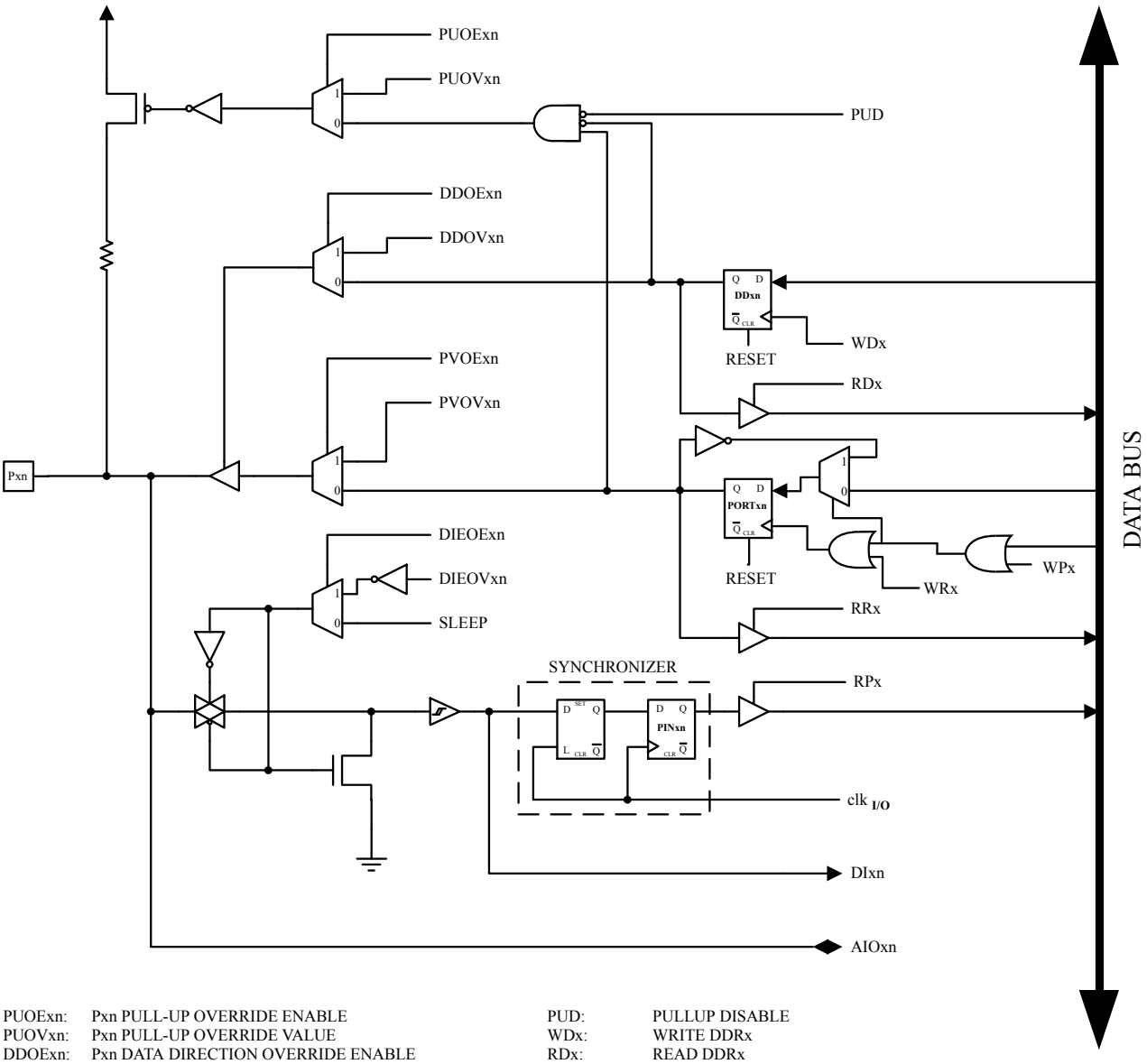
Reset and Interrupt Handling on page 27

13.2. Interrupt Vectors in ATmega324P

Table 13-1. Reset and Interrupt Vectors in ATmega324P

Vector No	Program Address(2)	Source	Interrupts definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	INT2	External Interrupt Request 2
5	0x0008	PCINT0	Pin Change Interrupt Request 0
6	0x000A	PCINT1	Pin Change Interrupt Request 1
7	0x000C	PCINT2	Pin Change Interrupt Request 2
8	0x000E	PCINT3	Pin Change Interrupt Request 3
9	0x0010	WDT	Watchdog Time-out Interrupt
10	0x0012	TIMER2_COMPA	Timer/Counter2 Compare Match A
11	0x0014	TIMER2_COMPB	Timer/Coutner2 Compare Match B
12	0x0016	TIMER2_OVF	Timer/Counter2 Overflow
13	0x0018	TIMER1_CAPT	Timer/Counter1 Capture Event
14	0x001A	TIMER1_COMPA	Timer/Counter1 Compare Match A
15	0x001C	TIMER1_COMPB	Timer/Coutner1 Compare Match B
16	0x001E	TIMER1_OVF	Timer/Counter1 Overflow
17	0x0020	TIMER0_COMPA	Timer/Counter0 Compare Match A
18	0x0022	TIMER0_COMPB	Timer/Coutner0 Compare Match B
19	0x0024	TIMER0_OVF	Timer/Counter0 Overflow
20	0x0026	SPI_STC	SPI Serial Transfer Complete

Figure 15-5. Alternate Port Functions[1]



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE
SLEEP:	SLEEP CONTROL

PUD:	PULLUP DISABLE
WDx:	WRITE DDRx
RDx:	READ DDRx
RRx:	READ PORTx REGISTER
WRx:	WRITE PORTx
RPx:	READ PORTx PIN
WPx:	WRITE PINx
$clk_{I/O}$:	I/O CLOCK
DIxn:	DIGITAL INPUT PIN n ON PORTx
AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

The following table summarizes the function of the overriding signals. The pin and port indexes from previous figure are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

- SCL/PCINT16 – Port C, Bit 0
  - SCL: two-wire Serial Bus Clock Line.
  - PCINT16: Pin Change Interrupt source 16. The PC0 pin can serve as an external interrupt source.

The tables below relate the alternate functions of Port C to the overriding signals shown in Figure 15-5.

Table 15-10. Overriding Signals for Alternate Functions in PC[7:4]

Signal Name	PC7/TOSC2/PCINT23	PC6/TOSC1/PCINT22	PC5/TDI/PCINT21	PC4/TDO/PCINT20
PUOE	AS2 • $\overline{\text{EXCLK}}$	AS2	JTAGEN	JTAGEN
PUOV	0	0	1	1
DDOE	AS2 • $\overline{\text{EXCLK}}$	AS2	JTAGEN	JTAGEN
DDOV	0	0	0	SHIFT_IR + SHIFT_DR
PVOE	0	0	0	JTAGEN
PVOV	0	0	0	TDO
DIEOE	AS2 • $\overline{\text{EXCLK}}$ + PCINT23 • PCIE2	AS2 + PCINT22 • PCIE2	JTAGEN + PCINT21 • PCIE2	JTAGEN + PCINT20 • PCIE2
DIEOV	$\overline{\text{AS2}}$	EXCLK + $\overline{\text{AS2}}$	$\overline{\text{JTAGEN}}$	$\overline{\text{JTAGEN}}$
DI	PCINT23 INPUT	PCINT22 INPUT	PCINT21 INPUT	PCINT20 INPUT
AIO	TC2 OSC OUTPUT	TC1 OSC INPUT	TDI INPUT	-

Table 15-11. Overriding Signals for Alternate Functions in PC[3:0]

Signal Name	PC3/TMS/PCINT19	PC2/TCK/PCINT18	PC1/SDA/PCINT17	PC0/SCL/PCINT16
PUOE	JTAGEN	JTAGEN	TWEN	TWEN
PUOV	1	1	PORTC1 • $\overline{\text{PUD}}$	PORTC0 • $\overline{\text{PUD}}$
DDOE	JTAGEN	JTAGEN	TWEN	TWEN
DDOV	0	0	0	0
PVOE	0	0	TWEN	TWEN
PVOV	0	0	SDA OUT	SCL OUT
DIEOE	JTAGEN + PCINT19 • PCIE2	JTAGEN + PCINT18 • PCIE2	PCINT17 • PCIE2	PCINT16 • PCIE2
DIEOV	$\overline{\text{JTAGEN}}$	$\overline{\text{JTAGEN}}$	1	1
DI	PCINT19 INPUT	PCINT18 INPUT	PCINT17 INPUT	PCINT16 INPUT
AIO	TMS INPUT	TCK INPUT	SDA INPUT	SCL INPUT

15.3.4. Alternate Functions of Port D

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

17.14.1. TC1 Control Register A

Name: TCCR1A
Offset: 0x80
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	COM1	COM1	COM1	COM1			WGM11	WGM10
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bits 4, 5, 6, 7 – COM1, COM1, COM1, COM1: Compare Output Mode for Channel
The COM1A[1:0] and COM1B[1:0] control the Output Compare pins (OC1A and OC1B respectively) behavior. If one or both of the COM1A[1:0] bits are written to one, the OC1A output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B[1:0] bit are written to one, the OC1B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC1A or OC1B pin must be set in order to enable the output driver.

When the OC1A or OC1B is connected to the pin, the function of the COM1x[1:0] bits is dependent of the WGM1[3:0] bits setting. The table below shows the COM1x[1:0] bit functionality when the WGM1[3:0] bits are set to a Normal or a CTC mode (non-PWM).

Table 17-3. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on Compare Match.
1	0	Clear OC1A/OC1B on Compare Match (Set output to low level).
1	1	Set OC1A/OC1B on Compare Match (Set output to high level).

The table below shows the COM1x[1:0] bit functionality when the WGM1[3:0] bits are set to the fast PWM mode.

Table 17-4. Compare Output Mode, Fast PWM

COM1A1/ COM1B1	COM1A0/ COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM1[3:0] = 14 or 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.

19.11.4. TC2 Output Compare Register A

Name: OCR2A
Offset: 0xB3
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	OCR2A[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – OCR2A[7:0]: Output Compare 2 A
The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT2). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC2A pin.

Table 21-2. Recommended Maximum Receiver Baud Rate Error for Normal Speed Mode (U2X = 0)

D # (Data+Parity Bit)	R_{slow} [%]	R_{fast} [%]	Max. Total Error [%]	Recommended Max. Receiver Error [%]
5	93.20	106.67	+6.67/-6.8	±3.0
6	94.12	105.79	+5.79/-5.88	±2.5
7	94.81	105.11	+5.11/-5.19	±2.0
8	95.36	104.58	+4.58/-4.54	±2.0
9	95.81	104.14	+4.14/-4.19	±1.5
10	96.17	103.78	+3.78/-3.83	±1.5

Table 21-3. Recommended Maximum Receiver Baud Rate Error for Double Speed Mode (U2X = 1)

D # (Data+Parity Bit)	R_{slow} [%]	R_{fast} [%]	Max Total Error [%]	Recommended Max Receiver Error [%]
5	94.12	105.66	+5.66/-5.88	±2.5
6	94.92	104.92	+4.92/-5.08	±2.0
7	95.52	104,35	+4.35/-4.48	±1.5
8	96.00	103.90	+3.90/-4.00	±1.5
9	96.39	103.53	+3.53/-3.61	±1.5
10	96.70	103.23	+3.23/-3.30	±1.0

The recommendations of the maximum receiver baud rate error was made under the assumption that the Receiver and Transmitter equally divides the maximum total error.

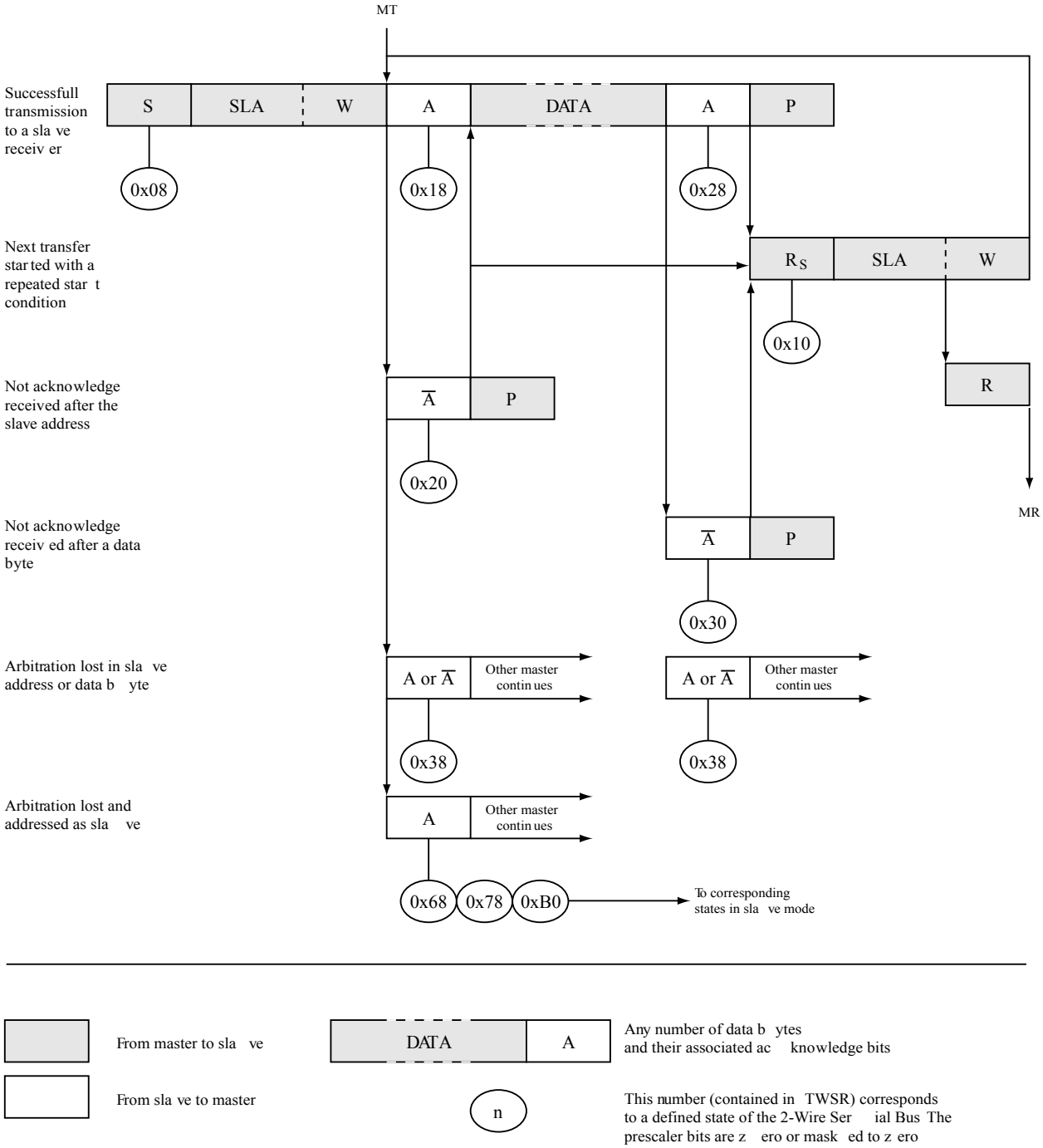
There are two possible sources for the receivers baud rate error. The Receiver's system clock (EXTCLK) will always have some minor instability over the supply voltage range and the temperature range. When using a crystal to generate the system clock, this is rarely a problem, but for a resonator, the system clock may differ more than 2% depending of the resonator's tolerance. The second source for the error is more controllable. The baud rate generator can not always do an exact division of the system frequency to get the baud rate wanted. In this case an UBRRn value that gives an acceptable low error can be used if possible.

21.10. 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 9 data bits, then the ninth bit (RXB8) is used for identifying address and data frames. When the frame type bit (the first

Figure 23-12. Formats and States in the Master Transmitter Mode



23.7.2. Master Receiver Mode

In the Master Receiver (MR) mode, a number of data bytes are received from a Slave Transmitter (see next figure). In order to enter a Master mode, a START condition must be transmitted. The format of the following address packet determines whether Master Transmitter (MT) or MR mode is to be entered. If SLA+W is transmitted, MT mode is entered, if SLA+R is transmitted, MR mode is entered. All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Table 23-7. Miscellaneous States

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDRn	To TWCRn				
			STA	STO	TWINT	TWEA	
0xF8	No relevant state information available; TWINT = “0”	No TWDRn action	No TWCRn action				Wait or proceed current transfer
0x00	Bus error due to an illegal START or STOP condition	No TWDRn action	0	1	1	X	Only the internal hardware is affected, no STOP condition is sent on the bus. In all cases, the bus is released and TWSTO is cleared.

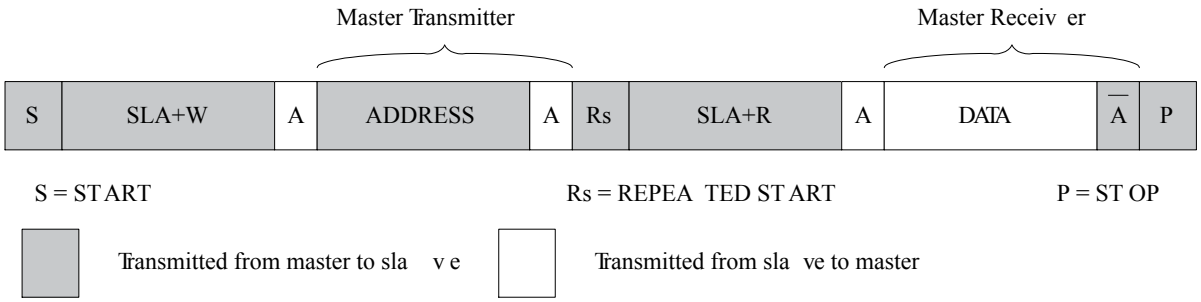
23.7.6. Combining Several TWI Modes

In some cases, several TWI modes must be combined in order to complete the desired action. Consider for example reading data from a serial EEPROM. Typically, such a transfer involves the following steps:

1. The transfer must be initiated.
2. The EEPROM must be instructed what location should be read.
3. The reading must be performed.
4. The transfer must be finished.

Note that data is transmitted both from Master to Slave and vice versa. The Master must instruct the Slave what location it wants to read, requiring the use of the MT mode. Subsequently, data must be read from the Slave, implying the use of the MR mode. Thus, the transfer direction must be changed. The Master must keep control of the bus during all these steps, and the steps should be carried out as an atomical operation. If this principle is violated in a multi master system, another Master can alter the data pointer in the EEPROM between steps 2 and 3, and the Master will read the wrong data location. Such a change in transfer direction is accomplished by transmitting a REPEATED START between the transmission of the address byte and reception of the data. After a REPEATED START, the Master keeps ownership of the bus. The flow in this transfer is depicted in the following figure:

Figure 23-19. Combining Several TWI Modes to Access a Serial EEPROM



23.8. Multi-master Systems and Arbitration

If multiple masters are connected to the same bus, transmissions may be initiated simultaneously by one or more of them. The TWI standard ensures that such situations are handled in such a way that one of the masters will be allowed to proceed with the transfer, and that no data will be lost in the process. An example of an arbitration situation is depicted below, where two masters are trying to transmit data to a Slave Receiver.

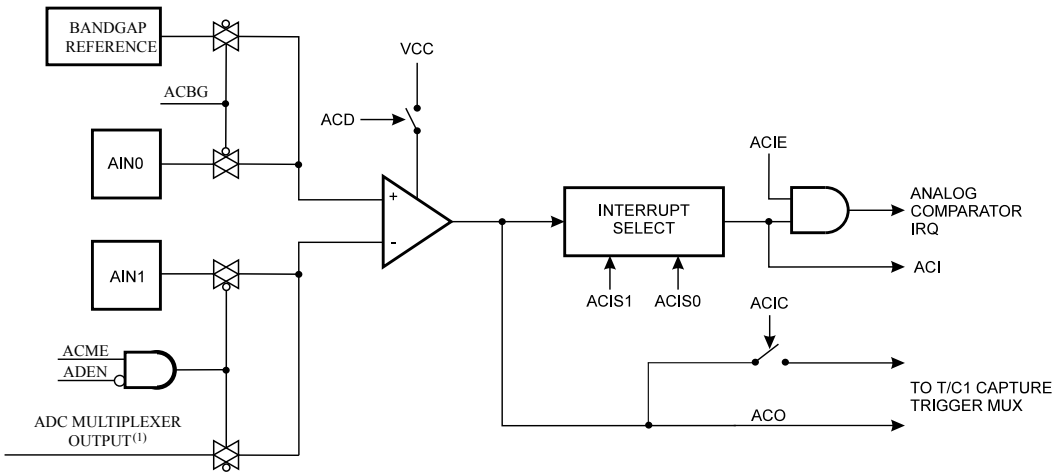
24. AC - Analog Comparator

24.1. 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 below.

The Power Reduction ADC bit in the Power Reduction Register (0.PRADC) must be written to '0' in order to be able to use the ADC input MUX.

Figure 24-1. Analog Comparator Block Diagram



Note: Refer to the *Pin Configuration* and the I/O Ports description for Analog Comparator pin placement

Related Links

I/O-Ports on page 95

Power Management and Sleep Modes on page 56

Minimizing Power Consumption on page 59

Pinout on page 15

24.2. Analog Comparator Multiplexed Input

It is possible to select any of the ADC[7:0] pins to replace the negative input to the Analog Comparator. The ADC multiplexer is used to select this input, and consequently, the ADC must be switched off to utilize this feature. If the Analog Comparator Multiplexer Enable bit in the ADC Control and Status Register B (ADCSRB.ACME) is '1' and the ADC is switched off (ADCSRA.ADEN=0), the three least significant Analog Channel Selection bits in the ADC Multiplexer Selection register (ADMUX.MUX[2:0]) select the input pin to replace the negative input to the Analog Comparator, as shown in the table below. When ADCSRB.ACME=0 or ADCSRA.ADEN=1, AIN1 is applied to the negative input of the Analog Comparator.

25. ADC - Analog to Digital Converter

25.1. Features

- 10-bit Resolution
- 0.5 LSB Integral Non-Linearity
- ±2 LSB Absolute Accuracy
- 13 - 260µs Conversion Time
- Up to 15kSPS at Maximum Resolution
- 8 Multiplexed Single Ended Input Channels
- Differential mode with selectable gain at 1x, 10x or 200x(1)
- Optional Left Adjustment for ADC Result Readout
- 0 - V_{CC} ADC Input Voltage Range
- 2.7V - V_{CC} Differential ADC Voltage Range
- Selectable 2.56V or 1.1V ADC Reference Voltage
- Free Running or Single Conversion Mode
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

Note:

1. The differential input channels are not tested for devices in PDIP Package. This feature is only guaranteed to work for devices in TQFP and VQFN/QFN/MLF Packages.

25.2. Overview

The device features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows 8 single-ended voltage inputs constructed from the pins of Port A. The single-ended voltage inputs refer to 0V (GND).

The device also supports 16 differential voltage input combinations. Two of the differential inputs (ADC1, ADC0 and ADC3, ADC2) are equipped with a programmable gain stage. This provides amplification steps of 0 dB (1x), 20 dB (10x), or 46 dB (200x) on the differential input voltage before the A/D conversion. Seven differential analog input channels share a common negative terminal (ADC1), while any other ADC input can be selected as the positive input terminal. If 1x or 10x gain is used, 8-bit resolution can be expected. If 200x gain is used, 6- bit resolution can be expected. Note that internal references of 1.1V should not be used on 10x and 200x gain.

The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown below.

The ADC has a separate analog supply voltage pin, AV_{CC}. AV_{CC} must not differ more than ±0.3V from V_{CC}. See section ADC Noise Canceler on how to connect this pin.

The Power Reduction ADC bit in the Power Reduction Register (PRR.PRADC) must be written to '0' in order to be enable the ADC.

The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the voltage on the AREF pin minus 1 LSB. Optionally, AV_{CC} or an internal 2.56V reference voltage may be connected to the AREF pin by

25.8.3. ADC Data Register Low and High Byte (ADLAR=0)

The ADCL and ADCH register pair represents the 16-bit value, ADC Data Register. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + 0x01. For more details on reading and writing 16-bit registers, refer to Accessing 16-bit Registers.

When an ADC conversion is complete, the result is found in these two registers.

If differential channels are used, the result is presented in two's complement form.

When ADCL is read, the ADC Data Register is not updated until ADCH is read. Consequently, if the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH.

The ADLAR bit and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set (ADLAR=1), the result is left adjusted. If ADLAR is cleared (ADLAR=0 which is the default value), the result is right adjusted.

Name: ADCL and ADCH
Offset: 0x78
Reset: 0x00
Property: ADLAR = 0

Bit	15	14	13	12	11	10	9	8
							ADC9	ADC8
Access							R	R
Reset							0	0

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 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 – ADCn: ADC Conversion Result
These bits represent the result from the conversion. Refer to ADC Conversion Result for details.

Table 26-1. AVR JTAG Part Number

Part Number	JTAG Part Number
ATmega324P	9511

26.11.2.3. Manufacturer ID

The Manufacturer ID is a 11-bit code identifying the manufacturer. The JTAG manufacturer ID for ATMEL is listed in the table below.

Table 26-2. Manufacturer ID

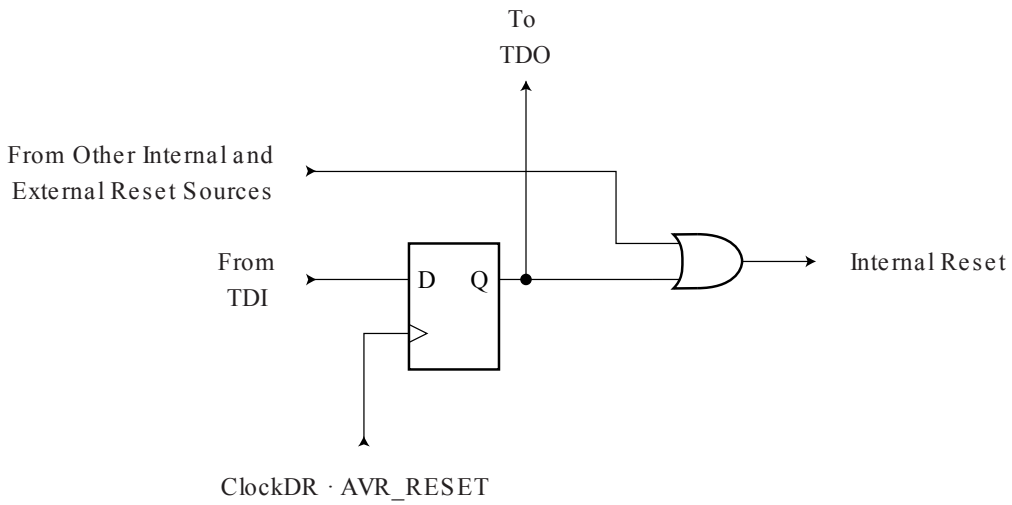
Manufacturer	JTAG Manufacturer ID (Hex)
ATMEL	0x01F

26.11.3. Reset Register

The Reset Register is a Test Data Register used to reset the part. Since the AVR tri-states Port Pins when reset, the Reset Register can also replace the function of the unimplemented optional JTAG instruction HIGHZ.

A high value in the Reset Register corresponds to pulling the External Reset low. The part is reset as long as there is a high value present in the Reset Register. Depending on the Fuse settings for the clock options, the part will remain reset for a Reset Time-Out Period (refer to *Clock Sources*) after releasing the Reset Register. The output from this Data Register is not latched, so the Reset will take place immediately, as shown in the figure below.

Figure 26-4. Reset Register



26.11.4. Boundary-scan Chain

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. Refer to Boundary-scan Chain for a complete description.

26.12. Boundry-scan Specific JTAG Instructions

The Instruction Register is 4-bit wide, supporting up to 16 instructions. Listed below are the JTAG instructions useful for Boundary-scan operation. Note that the optional HIGHZ instruction is not

Table 28-11. No. of Words in a Page and No. of Pages in the EEPROM

Device	EEPROM Size	Page Size	PCWORD	No. of Pages	PCPAGE	EEAMSB
ATmega324PA	1Kbytes	4bytes	EEA[1:0]	256	EEA[9:2]	9

28.7. Parallel Programming Parameters, Pin Mapping, and Commands

This section describes how to parallel program and verify Flash Program memory, EEPROM Data memory, Memory Lock bits, and Fuse bits in the device. Pulses are assumed to be at least 250ns unless otherwise noted.

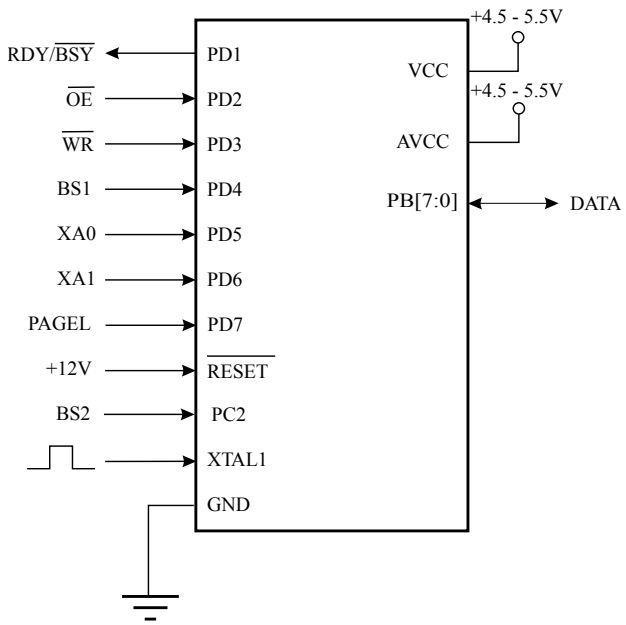
28.7.1. Signal Names

In this section, some pins of this device are referenced by signal names describing their functionality during parallel programming, please refer to Figure. Parallel Programming and Table. Pin Name Mapping in this section. Pins not described in the following table are referenced by pin names.

The XA1/XA0 pins determine the action executed when the XTAL1 pin is given a positive pulse. The bit coding is shown in the table, XA1 and XA0 Coding.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The different Commands are shown in the table, Command Byte Bit Coding Command Byte Command Executed.

Figure 28-1. Parallel Programming



Note: V_{CC} - 0.3V < AV_{CC} < V_{CC} + 0.3V, however, AV_{CC} should always be within 4.5 - 5.5V

Table 28-12. Pin Name Mapping

Signal Name in Programming Mode	Pin Name	I/O	Function
RDY/$\overline{BSY}$	PD1	O	0: Device is busy programming, 1: Device is ready for new command
$\overline{OE}$	PD2	I	Output Enable (Active low)

Signal Name in Programming Mode	Pin Name	I/O	Function
$\overline{WR}$	PD3	I	Write Pulse (Active low)
BS1	PD4	I	Byte Select 1 (“0” selects Low byte, “1” selects High byte)
XA0	PD5	I	XTAL Action Bit 0
XA1	PD6	I	XTAL Action Bit 1
PAGEL	PD7	I	Program memory and EEPROM Data Page Load
BS2	PC2	I	Byte Select 2 (“0” selects Low byte, “1” selects 2’nd High byte)
DATA	PB[7:0]	I/O	Bi-directional Data bus (Output when OE is low)

Table 28-13. BS2 and BS1 encoding.

BS2	BS1	Flash / EEPROM address	Flash data loading / reading	Fuse programming	Reading fuse and lock bits
0	0	Low Byte	Low Byte	Low Byte	Fuse Low Byte
0	1	High Byte	High Byte	High Byte	Lockbits
1	0	Extended High Byte	Reserved	Extended Byte	Extended Fuse Byte
1	1	Reserved	Reserved	Reserved	Fuse High Byte

Table 28-14. Pin Values Used to Enter Programming Mode

Pin	Symbol	Value
PAGEL	Prog_enable[3]	0
XA1	Prog_enable[2]	0
XA0	Prog_enable[1]	0
BS1	Prog_enable[0]	0

Table 28-15. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash or EEPROM Address (High or low address byte determined by BS1)
0	1	Load Data (High or Low data byte for Flash determined by BS1)
1	0	Load Command
1	1	No Action, Idle

Table 28-16. Command Byte Bit Coding

Command Byte	Command Executed
1000 0000	Chip Erase
0100 0000	Write Fuse bits

Figure 28-11. Programming Command Register

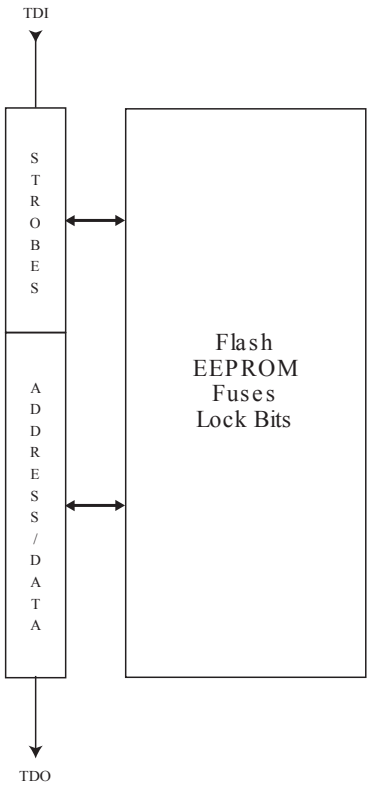


Table 28-20. JTAG Programming Instruction Set

a = address high bits, b = address low bits, H = 0 - Low byte, 1 - High Byte, o = data out, i = data in, x = don't care

Instruction	TDI sequence	TDO sequence	Notes
1a. Chip erase	0100011_10000000 0110001_10000000 0110011_10000000 0110011_10000000	xxxxxxx_xxxxxxxx xxxxxxx_xxxxxxxx xxxxxxx_xxxxxxxx xxxxxxx_xxxxxxxx	
1b. Poll for chip erase complete	0110011_10000000	xxxxxox_xxxxxxxx	(2)
2a. Enter Flash Write	0100011_00010000	xxxxxxx_xxxxxxxx	
2b. Load Address High Byte	0000111_aaaaaaaa	xxxxxxx_xxxxxxxx	(9)
2c. Load Address Low Byte	0000011_bbbbbbbb	xxxxxxx_xxxxxxxx	
2d. Load Data Low Byte	0010011_iiiiiiii	xxxxxxx_xxxxxxxx	
2e. Load Data High Byte	0010111_iiiiiiii	xxxxxxx_xxxxxxxx	
2f. Latch Data	0110111_00000000 1110111_00000000 0110111_00000000	xxxxxxx_xxxxxxxx xxxxxxx_xxxxxxxx xxxxxxx_xxxxxxxx	(1)

29.7. SPI Timing Characteristics

Table 29-10. SPI Timing Parameters

Description	Mode	Min.	Typ	Max	Units
SCK period	Master	-	See Table. Relationship Between SCK and the Oscillator Frequency in "SPCR – SPI Control Register"	-	ns
SCK high/low	Master	-	50% duty cycle	-	
Rise/Fall time	Master	-	3.6	-	
Setup	Master	-	10	-	
Hold	Master	-	10	-	
Out to SCK	Master	-	$0.5 \cdot t_{sck}$	-	
SCK to out	Master	-	10	-	
SCK to out high	Master	-	10	-	
$\overline{SS}$ low to out	Slave	-	15	-	
SCK period	Slave	$4 \cdot t_{ck}$	-	-	
SCK high/low(1)	Slave	$2 \cdot t_{ck}$	-	-	
Rise/Fall time	Slave	-	-	1600	
Setup	Slave	10	-	-	
Hold	Slave	t_{ck}	-	-	
SCK to out	Slave	-	15	-	
SCK to $\overline{SS}$ high	Slave	20	-	-	
$\overline{SS}$ high to tri-state	Slave	-	10	-	
$\overline{SS}$ low to SCK	Slave	$2 \cdot t_{ck}$	-	-	

Note:

1. In SPI Programming mode the minimum SCK high/low period is:

- $2\ t_{CLCL}$ for $f_{CK} < 12MHz$
- $3\ t_{CLCL}$ for $f_{CK} > 12MHz$

30.11. Internal Oscillator Speed

Figure 30-34. Watchdog Oscillator Frequency vs. Temperature

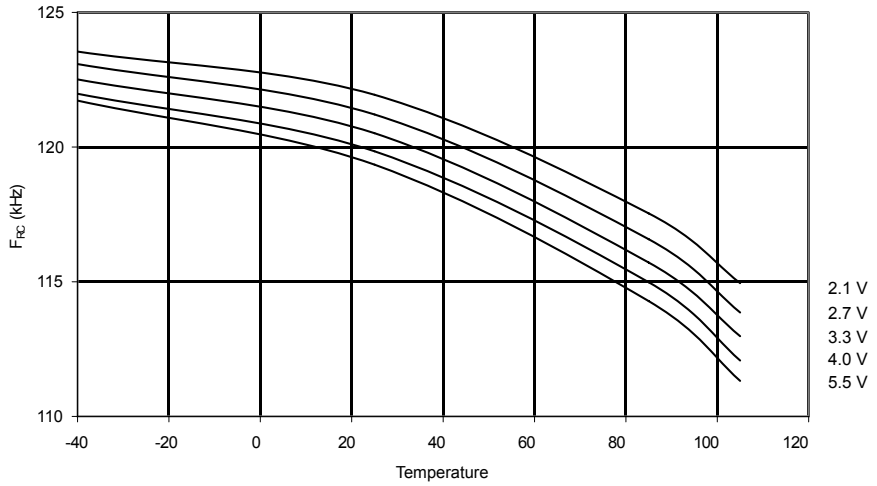


Figure 30-35. Watchdog Oscillator Frequency vs. V_{CC}