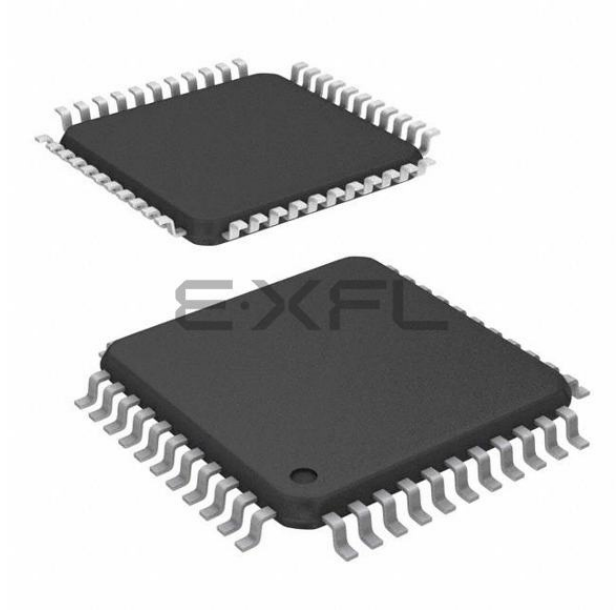


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	AVR
Core Size	8-Bit
Speed	8MHz
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)	4.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C
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/atmega323-8ac

Table 2. ATmega323 I/O Space (Continued)

I/O Address (SRAM Address)	Name	Function
$12 ($32)	PORTD	Data Register, Port D
$11 ($31)	DDRD	Data Direction Register, Port D
$10 ($30)	PIND	Input Pins, Port D
$0F ($2F)	SPDR	SPI I/O Data Register
$0E ($2E)	SPSR	SPI Status Register
$0D ($2D)	SPCR	SPI Control Register
$0C ($2C)	UDR	USART I/O Data Register
$0B ($2B)	UCSRA	USART Control and Status Register A
$0A ($2A)	UCSRB	USART Control and Status Register B
$09 ($29)	UBRRL	USART Baud Rate Register Low Byte
$08 ($28)	ACSR	Analog Comparator Control and Status Register
$07 ($27)	ADMUX	ADC Multiplexer Select Register
$06 ($26)	ADCSR	ADC Control and Status Register
$05 ($25)	ADCH	ADC Data Register High
$04 ($24)	ADCL	ADC Data Register Low
$03 ($23)	TWDR	Two-wire Serial Interface Data Register
$02 ($22)	TWAR	Two-wire Serial Interface (Slave) Address Register
$01 ($21)	TWSR	Two-wire Serial Interface Status Register
$00 ($20)	TWBR	Two-wire Serial Interface Bit Rate Register

Notes: 1. When the OCDEN Fuse is unprogrammed, the OSCCAL Register is always accessed on this address. Refer to the debugger specific documentation for details on how to use the OCDR Register.
2. Refer to the USART description for details on how to access UBRRH and UCSRC.

All ATmega323 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O Registers within the address range $00 - $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set chapter for more details. When using the I/O specific commands IN and OUT, the I/O addresses $00 - $3F must be used. When addressing I/O Registers as SRAM, $20 must be added to these addresses. All I/O Register addresses throughout this document are shown with the SRAM address in parentheses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O Memory addresses should never be written.

Some of the Status Flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O Register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

The I/O and Peripherals Control Registers are explained in the following sections.

Table 3. Reset and Interrupt Vectors (Continued)

Vector No.	Program Address(2)	Source	Interrupt Definition
10	$012	TIMER1 OVF	Timer/Counter1 Overflow
11	$014	TIMER0 COMP	Timer/Counter0 Compare Match
12	$016	TIMER0 OVF	Timer/Counter0 Overflow
13	$018	SPI, STC	Serial Transfer Complete
14	$01A	USART, RXC	USART, Rx Complete
15	$01C	USART, UDRE	USART Data Register Empty
16	$01E	USART, TXC	USART, Tx Complete
17	$020	ADC	ADC Conversion Complete
18	$022	EE_RDY	EEPROM Ready
19	$024	ANA_COMP	Analog Comparator
20	$026	TWSI	Two-wire Serial Interface

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see “Boot Loader Support” on page 177.
2. When the IVSEL bit in GICR is set, Interrupt Vectors will be moved to the start of the boot Flash section. The address of each Interrupt Vector will then be address in this table plus the start address of the boot Flash section.

Table 4 shows Reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings.

Table 4. Reset and Interrupt Vectors Placement

BOOTRST	IVSEL	Reset address	Interrupt Vectors Start Address
0	0	$0000	$0002
0	1	$0000	Boot Reset Address + $0002
1	0	Boot Reset Address	$0002
1	1	Boot Reset Address	Boot Reset Address + $0002

Note: The Boot Reset Address is shown in Table 59 on page 177.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega323 is:

```
Address   Labels  Code            Comments
$000              jmp   RESET     ; Reset Handler
$002              jmp   EXT_INT0  ; IRQ0 Handler
$004              jmp   EXT_INT1  ; IRQ1 Handler
$006              jmp   EXT_INT2  ; IRQ2 Handler
$008              jmp   TIM2_COMP ; Timer2 Compare Handler
$00a              jmp   TIM2_OVF  ; Timer2 Overflow Handler
$00c              jmp   TIM1_CAPT ; Timer1 Capture Handler
$00e              jmp   TIM1_COMPA ; Timer1 CompareA Handler
$010              jmp   TIM1_COMPB ; Timer1 CompareB Handler
$012              jmp   TIM1_OVF  ; Timer1 Overflow Handler
$014              jmp   TIM0_COMP ; Timer0 Compare Handler
$016              jmp   TIM0_OVF  ; Timer0 Overflow Handler
```

- **Bit 6 – ISC2: Interrupt Sense Control 2**

The asynchronous external interrupt 2 is activated by the external pin INT2 if the SREG I-flag and the corresponding interrupt mask in the GICR are set. If ISC2 is cleared (zero), a falling edge on INT2 activates the interrupt. If ISC2 is set (one) a rising edge on INT2 activates the interrupt. Edges on INT2 are registered asynchronously. Pulses on INT2 wider than 50 ns will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. When changing the ISC2 bit, an interrupt can occur. Therefore, it is recommended to first disable INT2 by clearing its Interrupt Enable bit in the GICR Register. Then, the ISC2 bit can be changed. Finally, the INT2 Interrupt Flag should be cleared by writing a logical one to its Interrupt Flag bit in the GIFR Register before the interrupt is re-enabled.

- **Bit 5 – Res: Reserved Bit**

This bit is a reserved bit in the ATmega323 and always reads as zero.

- **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 MCUCSR 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.

Internal Voltage Reference

ATmega323 features an internal bandgap reference with a nominal voltage of 1.22V. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC. The 2.56V reference to the ADC is generated from the internal bandgap reference.

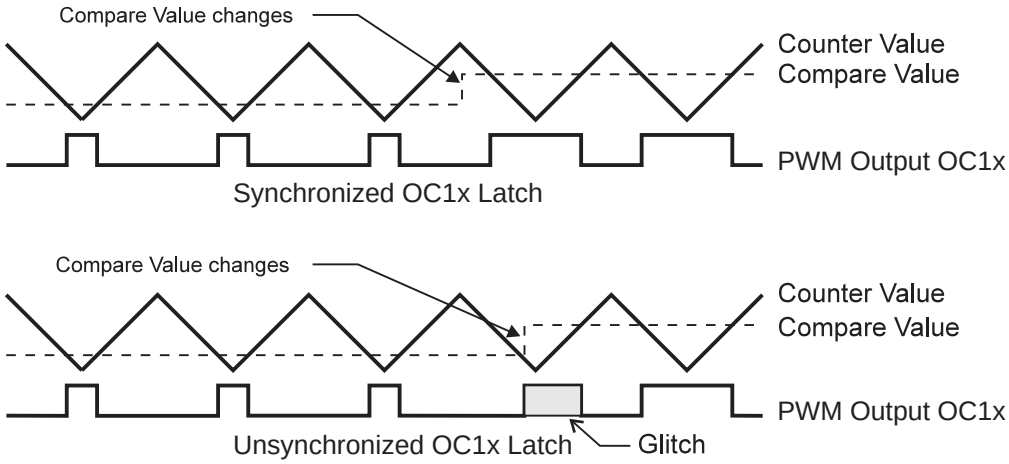
- **Bit 1 – PSR2: Prescaler Reset Timer/Counter2**

When this bit is set (one) the Timer/Counter2 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. This bit will always be read as zero if Timer/Counter2 is clocked by the internal CPU clock. If this bit is written when Timer/Counter2 is operating in asynchronous mode, the bit will remain one until the prescaler has been reset. See “Asynchronous Operation of Timer/Counter2” on page 53 for a detailed description of asynchronous operation.

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

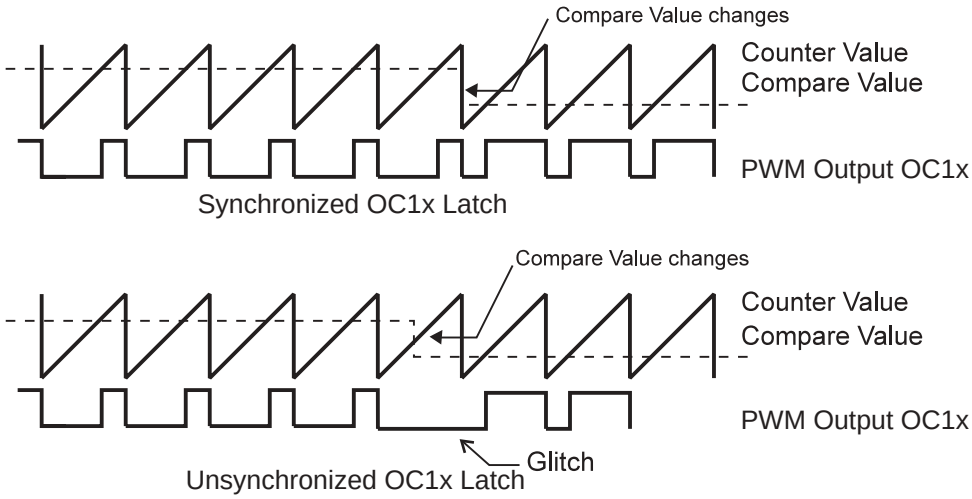
When this bit is set (one) the Timer/Counter1 and Timer/Counter0 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers. This bit will always be read as zero.

Figure 38. Effects of Unsynchronized OCR1 Latching.



Note: x = A or B

Figure 39. Effects of Unsynchronized OCR1 Latching in Overflow Mode



Note: X = A or B

During the time between the write and the latch operation, a read from OCR1A or OCR1B will read the contents of the temporary location. This means that the most recently written value always will read out of OCR1A/B.

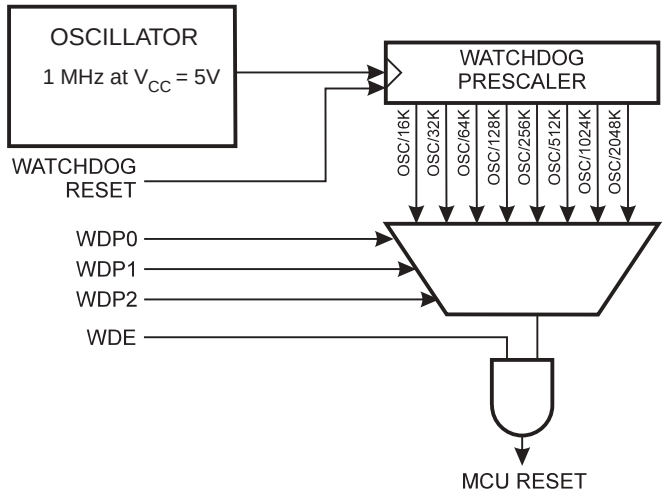
When the OCR1X contains $0000 or TOP, and the up/down PWM mode is selected, the output OC1A/OC1B is updated to low or high on the next compare match according to the settings of COM1A1/COM1A0 or COM1B1/COM1B0. This is shown in Table 23. In overflow PWM mode, the output OC1A/OC1B is held low or high only when the Output Compare Register contains TOP.

Watchdog Timer

The Watchdog Timer is clocked from a separate On-chip Oscillator which runs at 1 Mhz. This is the typical value at V_{CC} = 5V. See characterization data for typical values at other V_{CC} levels. By controlling the Watchdog Timer prescaler, the Watchdog Reset interval can be adjusted as shown in Table 24 on page 65. The WDR – Watchdog Reset – instruction resets the Watchdog Timer. Eight different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog Reset, the ATmega323 resets and executes from the Reset Vector. For timing details on the Watchdog Reset, refer to page 30.

To prevent unintentional disabling of the Watchdog, a special turn-off sequence must be followed when the Watchdog is disabled. Refer to the description of the Watchdog Timer Control Register for details.

Figure 40. Watchdog Timer



The Watchdog Timer Control Register – WDTCR

Bit	7	6	5	4	3	2	1	0	
$21 ($41)	–	–	–	WDTOE	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 bits in the ATmega323 and will always read as zero.

- **Bit 4 – WDTOE: Watchdog Turn-off Enable**

This bit must be set (one) when the WDE bit is cleared. Otherwise, the Watchdog will not be disabled. Once set, hardware will clear this bit to zero after four clock cycles. Refer to the description of the WDE bit for a Watchdog disable procedure.

- **Bit 3 – WDE: Watchdog Enable**

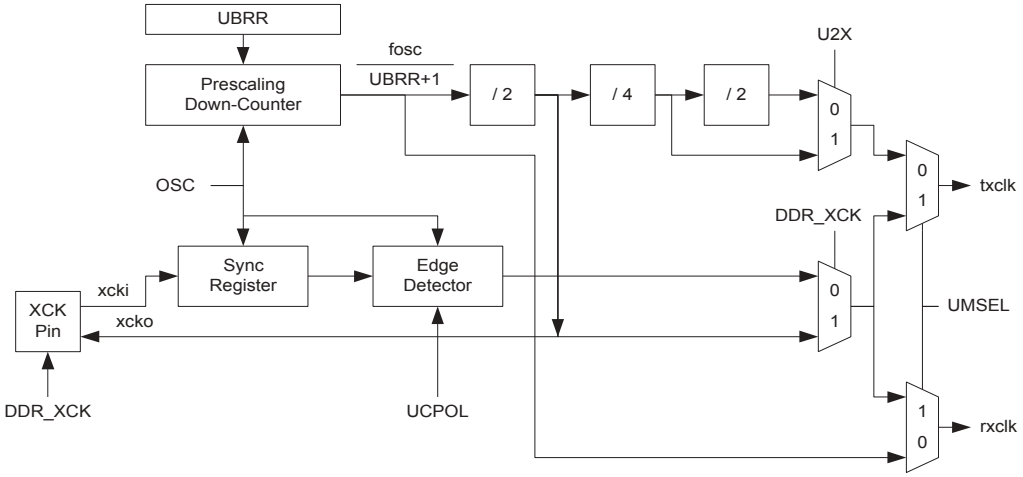
When the WDE is set (one) the Watchdog Timer is enabled, and if the WDE is cleared (zero) the Watchdog Timer function is disabled. WDE can only be cleared if the WDTOE bit is set(one). To disable an enabled Watchdog timer, the following procedure must be followed:

Clock Generation

The Clock Generation Logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation: Normal asynchronous, Double Speed asynchronous, Master synchronous and Slave synchronous mode. The UMSEL bit in USART Control and Status Register C (UCSRC) selects between asynchronous and synchronous operation. Double Speed (asynchronous mode only) is controlled by the U2X found in the UCSRA Register. When using synchronous mode (UMSEL = 1), the Data Direction Register for the XCK pin (DDR_XCK) controls whether the clock source is internal (Master mode) or external (Slave mode). The XCK pin is only active when using synchronous mode.

Figure 46 shows a block diagram of the clock generation logic.

Figure 46. Clock Generation Logic, Block Diagram



Signal description:

- **txclk** Transmitter clock. (Internal Signal)
- **rxclk** Receiver base clock. (Internal Signal)
- **xcki** Input from XCK pin (internal Signal). Used for Synchronous Slave operation.
- **xcko** Clock output to XCK pin (Internal Signal). Used for synchronous Master operation.
- **fosc** XTAL pin frequency (System Clock).

Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the asynchronous and the synchronous Master modes of operation. The description in this section refers to Figure 46.

The USART Baud Rate Register (UBRR) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock (fosc), is loaded with the UBRR value each time the counter has counted down to zero or when the UBRRL Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= fosc/(UBRR+1)). The Transmitter divides the baud rate generator clock output by 2, 8, or 16 depending on mode. The baud rate generator output is used directly by the Receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8, or 16 states depending on mode set by the state of the UMSEL, U2X and DDR_XCK bits.

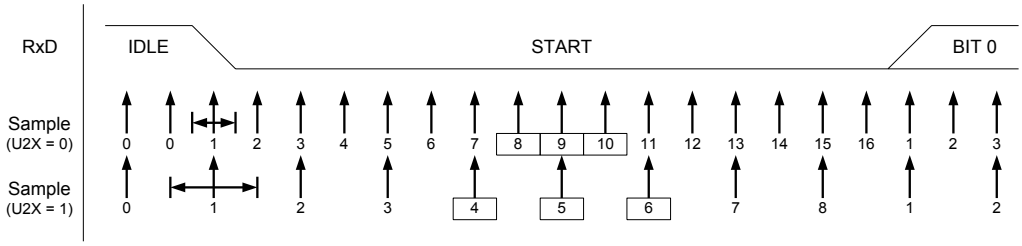
Asynchronous Data Reception

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

Asynchronous Clock Recovery

The clock recovery logic synchronizes internal clock to the incoming serial frames. Figure 49 illustrates the sampling process of the start bit of an incoming frame. The sample rate is 16 times the baud rate for normal mode, and 8 times the baud rate for Double Speed mode. The horizontal arrows illustrate the synchronization variation due to the sampling process. Note the larger time variation when using the double speed mode (U2X = 1) of operation. Samples denoted zero are samples done when the RxD line is idle (i.e. no communication activity).

Figure 49. Start Bit Sampling

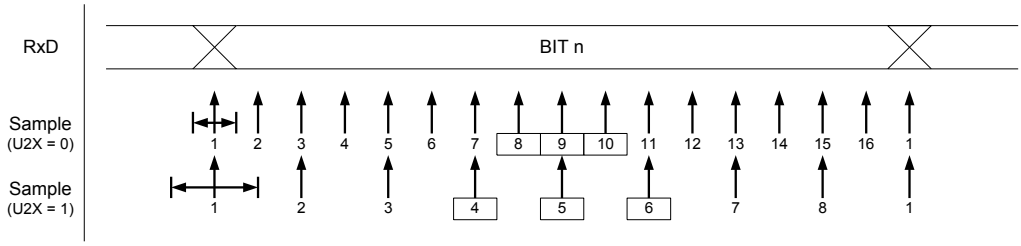


When the clock recovery logic detects a high (idle) to low (start) transition on the RxD line, the start bit detection sequence is initiated. Let sample 1 denote the first zero-sample as shown in the figure. The clock recovery logic then uses samples 8, 9 and 10 for Normal mode, and samples 4, 5 and 6 for Double Speed mode (indicated with sample numbers inside boxes on the figure), to decide if a valid start bit is received. If two or more of these three samples have logical high levels (the majority wins), the start bit is rejected as a noise spike and the Receiver starts looking for the next high to low-transition. If however, a valid start bit is detected, the clock recovery logic is synchronized and the data recovery can begin. The synchronization process is repeated for each start bit.

Asynchronous Data Recovery

When the Receiver clock is synchronized to the start bit, the data recovery can begin. The data recovery unit uses a state machine that has 16 states for each bit in Normal mode and 8 states for each bit in Double Speed mode. Figure 50 shows the sampling of the data bits and the parity bit. Each of the samples is given a number that is equal to the state of the recovery unit.

Figure 50. Sampling of Data and Parity Bit



USART Control and Status Register B – UCSRB

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	UCSRB
Read/Write	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 – RXCIE: RX Complete Interrupt Enable**

Setting this bit to one enables interrupt on the RXC Flag. A USART Receive Complete interrupt will be generated only if the RXCIE bit is set, the Global Interrupt Flag in SREG is set and the RXC bit in UCSRA is set.

- **Bit 6 – TXCIE: TX Complete Interrupt Enable**

Setting this bit to one enables interrupt on the TXC Flag. A USART Transmit Complete interrupt will be generated only if the TXCIE bit is set, the Global Interrupt Flag in SREG is set and the TXC bit in UCSRA is set.

- **Bit 5 – UDRIE: USART Data Register Empty Interrupt Enable**

Setting this bit to one enables interrupt on the UDRE Flag. A Data Register Empty Interrupt will be generated only if the UDRIE bit is set, the Global Interrupt Flag in SREG is set and the UDRE bit in UCSRA is set.

- **Bit 4 – RXEN: Receiver Enable**

Setting this bit to one enables the USART Receiver. The Receiver will override normal port operation for the RxD pin when enabled. Disabling the Receiver will flush the Receive Buffer invalidating the FE, DOR, and PE Flags.

- **Bit 3 – TXEN: Transmitter Enable**

Setting this bit to one enables the USART Transmitter. The Transmitter will override normal port operation for the TxD pin when enabled. The disabling of the Transmitter (setting the TXEN to zero) will not become effective until ongoing and pending transmissions are completed, i.e. when the Transmit Shift Register and Transmit Buffer Register does not contain data to be transmitted. When disabled the Transmitter will no longer override the TxD port.

- **Bit 2 – UCSZ2: Character Size**

The UCSZ2 bits combined with the UCSZ1:0 bit in UCSRC sets the number of data bits (character size) in a frame the Receiver and Transmitter use.

- **Bit 1 – RXB8: Receive Data Bit 8**

RXB8 is the ninth data bit of the received character when operating with serial frames with nine data bits. Must be read before reading the low bits from UDR.

- **Bit 0 – TXB8: Transmit Data Bit 8**

TXB8 is the ninth data bit in the character to be transmitted when operating with serial frames with nine data bits. Must be written before writing the low bits to UDR.

When the Two-wire Serial Interface Interrupt Flag is set, the status code in TWSR is used to determine the appropriate software action. For each status code, the required software action and details of the following serial transfer are given in Table 37 to Table 44.

Master Transmitter Mode

In the Master Transmitter mode, a number of data bytes are transmitted to a Slave Receiver (see Figure 55). Before Master Transmitter mode can be entered, the TWCR must be initialized as follows:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
Value	0	X	0	0	0	1	0	X

TWEN must be set to enable the Two-wire Serial Interface, TWSTA and TWSTO must be cleared.

The Master Transmitter mode may now be entered by setting the TWSTA bit. The Two-wire Serial Interface logic will then test the Two-wire Serial Bus and generate a START condition as soon as the bus becomes free. When a START condition is transmitted, the Two-wire Serial Interface Interrupt Flag (TWINT) is set by hardware, and the status code in TWSR will be $08. TWDR must then be loaded with the slave address and the Data Direction bit (SLA+W). Clearing the TWINT bit in software will continue the transfer. The TWINT Flag is cleared by writing a logic one to the flag.

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, TWINT is set again and a number of status codes in TWSR are possible. Possible status codes in Master mode are $18, $20, or $38. The appropriate action to be taken for each of these status codes is detailed in Table 37. The data must be loaded when TWINT is high only. If not, the access will be discarded, and the Write Collision bit – TWWC will be set in the TWCR Register. This scheme is repeated until the last byte is sent and the transfer is ended by generating a STOP condition or a repeated START condition. A STOP condition is generated by setting TWSTO, a repeated START condition is generated by setting TWSTA and TWSTO.

After a repeated START condition (state $10) the Two-wire Serial Interface can access the same Slave again, or a new Slave without transmitting a STOP condition. Repeated START enables the Master to switch between Slaves, Master Transmitter mode and Master Receiver mode without loosing control over the bus.

Assembly code illustrating operation of the Master Transmitter mode is given at the end of the TWI section.

Master Receiver Mode

In the Master Receiver mode, a number of data bytes are received from a Slave Transmitter (see Figure 56). The transfer is initialized as in the Master Transmitter mode. When the START condition has been transmitted, the TWINT Flag is set by hardware. The software must then load TWDR with the 7-bit slave address and the Data Direction bit (SLA+R). The transfer will then continue when the TWINT Flag is cleared by software.

When the slave address and the direction bit have been transmitted and an acknowledgement bit has been received, TWINT is set again and a number of status codes in TWSR are possible. Possible status codes in Master mode are $40, $48, or $38. The appropriate action to be taken for each of these status codes is detailed in Table . Received data can be read from the TWDR Register when the TWINT Flag is set high by hardware. This scheme is repeated until the last byte has been received and a STOP condition is transmitted by writing a logic one to the TWSTO bit in the TWCR Register.

I/O Ports

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies for changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input).

Port A

Port A is an 8-bit bi-directional I/O port with optional internal pull-ups.

Three I/O Memory address locations are allocated for Port A, one each for the Data Register – PORTA, $1B($3B), Data Direction Register – DDRA, $1A($3A) and the Port A Input Pins – PINA, $19($39). The Port A Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The Port A output buffers can sink 20 mA and thus drive LED displays directly. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

Port A has an alternate function as analog inputs for the ADC. If some Port A pins are configured as outputs, it is essential that these do not switch when a conversion is in progress. This might corrupt the result of the conversion.

During Power-down mode, the Schmitt Trigger of the digital input is disconnected. This allows analog signals that are close to $V_{CC}/2$ to be present during Power-down without causing excessive power consumption.

The Port A Data Register – PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	**PORTA7**	**PORTA6**	**PORTA5**	**PORTA4**	**PORTA3**	**PORTA2**	**PORTA1**	**PORTA0**	**PORTA**
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 Port A Data Direction Register – DDRA

Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	**DDA7**	**DDA6**	**DDA5**	**DDA4**	**DDA3**	**DDA2**	**DDA1**	**DDA0**	**DDRA**
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 Port A Input Pins Address – PINA

Bit	7	6	5	4	3	2	1	0	
$19 ($39)	**PINA7**	**PINA6**	**PINA5**	**PINA4**	**PINA3**	**PINA2**	**PINA1**	**PINA0**	**PINA**
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

The Port A Input Pins address – PINA – is not a register, and this address enables access to the physical value on each Port A pin. When reading PORTA the Port A Data Latch is read, and when reading PINA, the logical values present on the pins are read.

- **TCK – Port C, Bit 2**

TCK, JTAG Test Clock: JTAG operation is synchronous to TCK. When the JTAG interface is enabled, this pin can not be used as an I/O pin. Refer to the section “JTAG Interface and the On-chip Debug System” on page 157 for details on operation of the JTAG interface.

- **SDA – Port C, Bit 1**

SDA, Two-wire Serial Interface Data: When the TWEN bit in TWCR is set (one) to enable the Two-wire Serial Interface, pin PC1 is disconnected from the port and becomes the Serial Data I/O pin for the Two-wire Serial Interface. In this mode, there is a spike filter on the pin to capture spikes shorter than 50 ns on the input signal, and the pin is driven by an open collector driver with slew rate limitation.

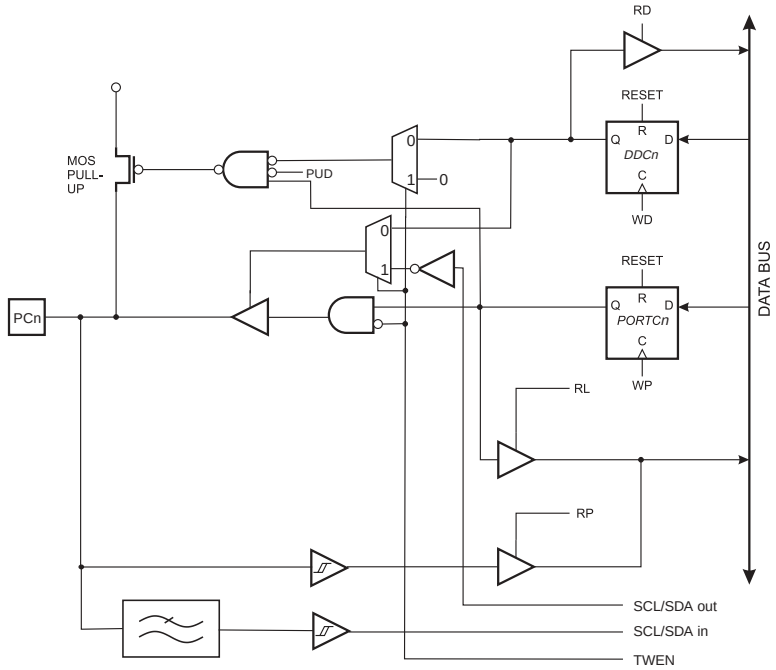
- **SCL – Port C, Bit 0**

SCL, Two-wire Serial Interface Clock: When the TWEN bit in TWCR is set (one) to enable the Two-wire Serial Interface, pin PC1 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 capture spikes shorter than 50 ns on the input signal.

Port C Schematics

Note that all port pins are synchronized. The synchronization latches are not shown in the figure.

Figure 75. Port C Schematic Diagram (Pins PC0 - PC1)



WP: WRITE PORTC
WD: WRITE DDRC
RL: READ PORTC LATCH
RP: READ PORTC PIN
RD: READ DDRC
PUD: PULL-UP DISABLE
n = 0, 1

JTAG Interface and the On-chip Debug System

Features

- **JTAG (IEEE std. 1149.1 Compliant) Interface**
- **Boundary-Scan Capabilities According to the JTAG Standard**
- **Debugger Access to:**
  - **All Internal Peripheral Units**
  - **Internal and External RAM**
  - **The Internal Register File**
  - **Program Counter**
  - **EEPROM and Flash Memories**
- **Extensive On-chip Debug Support for Break Conditions, Including**
  - **Break on Change of Program Memory Flow**
  - **Single Step Break**
  - **Program Memory Break Points on Single Address or Address Range**
  - **Data Memory Break Points on Single Address or Address Range**
- **Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface**
- **On-chip Debugging Supported by AVR Studio®**

Overview

The AVR IEEE std. 1149.1 compliant JTAG interface can be used for

- Testing PCBs by using the JTAG Boundary-Scan Capability.
- Programming the Non-volatile Memories, Fuses and Lock bits.
- On-chip Debugging.

A brief description is given in the following sections. Detailed descriptions for Programming via the JTAG interface, and using the Boundary-Scan Chain can be found in the sections “Programming via the JTAG Interface” on page 202 and “IEEE 1149.1 (JTAG) Boundary-Scan” on page 164**,** respectively. The On-chip Debug support is considered being private JTAG instructions, and distributed within ATMEL and to selected third party vendors only.

Figure 85 shows a block diagram of the JTAG interface and the On-chip Debug system. The TAP Controller is a state machine controlled by the TCK and TMS signals. The TAP Controller selects either the JTAG Instruction Register or one of several Data Registers as the scan chain (Shift Register) between the TDI – input and TDO – output. The Instruction Register holds JTAG instructions controlling the behavior of a Data Register.

Of the Data Registers, the ID-Register, Bypass Register, and the Boundary-Scan Chain are used for board-level testing. The JTAG Programming Interface (actually consisting of several physical and virtual Data Registers) is used for Serial Programming via the JTAG interface. The Internal Scan Chain and Break Point Scan Chain are used for On-chip debugging only.

Perform a Page Write

To execute page write, set up the address in the Z-pointer, write “00101” to the five LSB in SPMCR and execute SPM within four clock cycles after writing SPMCR. The data in R1 and R0 is ignored. The page address must be written to Z14:Z7. During this operation, Z6:Z0 must be zero to ensure that the page is written correctly. It is recommended that the interrupts are disabled during the page write operation.

Consideration while Updating the Boot Loader Section

Special care must be taken if the user allows the Boot Loader section to be updated by leaving Boot Lock bit 11 unprogrammed. An accidental write to the Boot Loader itself can corrupt the entire Boot Loader, and further software updates might be impossible. If it is not necessary to change the Boot Loader software itself, it is recommended to program the Boot Lock bit 11 to protect the Boot Loader software from any internal software changes.

Wait for SPM Instruction to Complete

Though the CPU is halted during Page Write, Page Erase or Lock bit write, for future compatibility, the user software must poll for SPM complete by reading the SPMCR Register and loop until the SPMEN bit is cleared after a programming operation. See “Assembly code example for a Boot Loader” on page 185 for a code example.

Instruction Word Read after Page Erase, Page Write, and Lock Bit Write

To ensure proper instruction pipelining after programming action (Page Erase, Page Write, or Lock bit write), the SPM instruction must be followed with the sequence (.dw $FFFF - NOP) as shown below:

```
spm
.dw $FFFF
nop
```

If not, the instruction following SPM might fail. It is not necessary to add this sequence when the SPM instruction only loads the temporary buffer.

Avoid Reading the Application Section During Self-programming

During Self-programming (either Page Erase or Page Write), the user software should not read the application section. The user software itself must prevent addressing this section during the Self-programming operations. This implies that interrupts must be disabled or moved to the Boot Loader section. Before addressing the application section after the programming is completed, for future compatibility, the user software must write”10001” to the five LSB in SPMCR and execute SPM within four clock cycles. Then the user software should verify that the ASB bit is cleared. See “Assembly code example for a Boot Loader” on page 185 for an example. Though the ASB and ASRE bits have no special function in this device, it is important for future code compatibility that they are treated as described above.

Boot Loader Lock Bits

ATmega323 has two separate sets of Boot Lock bits which can be set independently. This gives the user a unique flexibility to select different levels of protection.

The user can select:

- To protect the entire Flash from a software update by the MCU.
- To only protect the Boot Loader Flash section from a software update by the MCU.
- To only protect application Flash section from a software update by the MCU.
- Allowing software update in the entire Flash.

See Table 61 for further details. The Boot Lock bits can be set in software and in Serial or Parallel Programming mode, but they can only be cleared by a chip erase command.

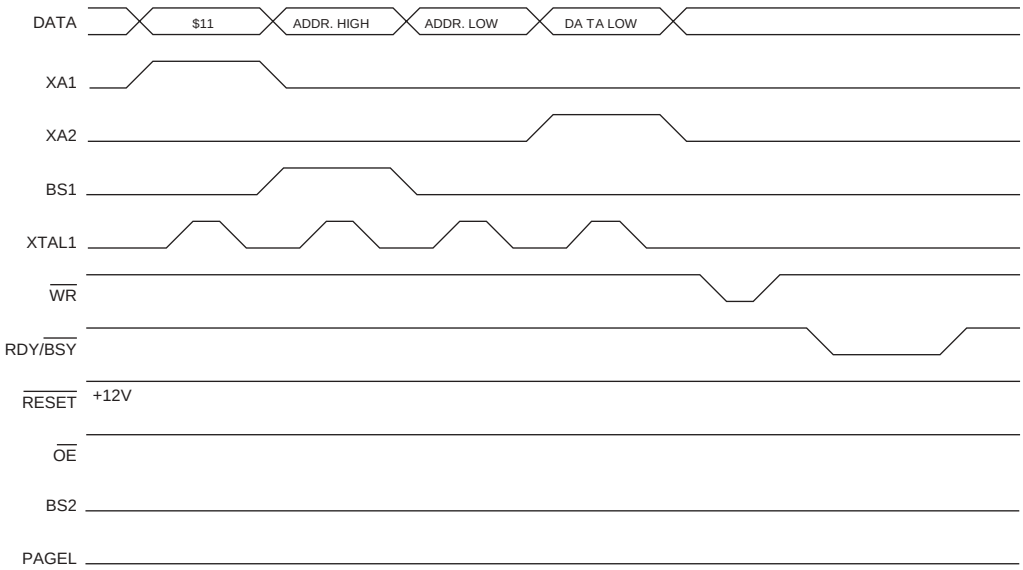
3. Wait until to RDY/$\overline{BSY}$ goes high before programming the next byte.
   (See Figure 96 for signal waveforms)

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

- The command needs only be loaded once when writing or reading multiple memory locations.
- Address High Byte needs only be loaded before programming a new 256 byte window in the EEPROM.
- Skip writing the data value $FF, that is the contents of the entire EEPROM after a Chip Erase.

These considerations also applies to Flash, EEPROM and Signature bytes reading.

Figure 96. Programming the EEPROM Waveforms



Reading the Flash

The algorithm for reading the Flash memory is as follows (refer to “Programming the Flash” on page 190 for details on Command and Address loading):

1. A: Load Command “0000 0010”.
2. H: Load Address High Byte ($00 - $3F)
3. B: Load Address Low Byte ($00 - $FF)
4. Set $\overline{OE}$ to “0”, and BS1 to “0”. The Flash word Low Byte can now be read at DATA.
5. Set BS to “1”. The Flash word High Byte can now be read at DATA.
6. Set $\overline{OE}$ to “1”.

Reading the EEPROM

The algorithm for reading the EEPROM Memory is as follows (refer to “Programming the Flash” on page 190 for details on Command and Address loading):

1. A: Load Command “0000 0011”.
2. H: Load Address High Byte ($00 - $03)
3. B: Load Address ($00 - $FF)
4. Set $\overline{OE}$ to “0”, and BS1 to “0”. The EEPROM Data byte can now be read at DATA.

Electrical Characteristics

Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
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	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*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.

DC Characteristics

T_A = -40°C to 85°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)	-0.5		0.3 V_{CC}(1)	V
V_{IL1}	Input Low Voltage	(XTAL1), CKSEL3 Fuse programmed	-0.5		0.3 V_{CC}(1)	V
		(XTAL1), CKSEL3 Fuse unprogrammed	-0.5		0.2 V_{CC}(1)	V
V_{IH}	Input High Voltage	(Except XTAL1, $\overline{\text{RESET}}$)	0.6 V_{CC}(2)		V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	(XTAL1), CKSEL3 Fuse programmed	0.6 V_{CC}(2)		V_{CC} + 0.5	V
		(XTAL1), CKSEL3 Fuse unprogrammed	0.8 V_{CC}(2)		V_{CC} + 0.5	V
V_{IH2}	Input High Voltage	($\overline{\text{RESET}}$)	0.9 V_{CC}(2)		V_{CC} + 0.5	V
V_{OL}	Output Low Voltage(3) (Ports A,B,C,D)	I_{OL} = 20 mA, V_{CC} = 5V I_{OL} = 10 mA, V_{CC} = 3V			0.6 0.5	V V
V_{OH}	Output High Voltage(4) (Ports A,B,C,D)	I_{OH} = -3 mA, V_{CC} = 5V I_{OH} = -1.5 mA, V_{CC} = 3V	4.2 2.3			V V
I_{IL}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin low (absolute value)			8.0	µA
I_{IH}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin high (absolute value)			980	nA
R_{RST}	Reset Pull-up Resistor		100		500	kΩ
$R_{I/O}$	I/O Pin Pull-up Resistor		35		120	kΩ

Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← $FF – Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← $00 – Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • ($FF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← $FF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
JMP	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3
ICALL		Indirect Call to (Z)	PC ← Z	None	3
CALL	k	Direct Subroutine Call	PC ← k	None	4
RET		Subroutine Return	PC ← Stack	None	4
RETI		Interrupt Return	PC ← Stack	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd,Rr	Compare	Rd – Rr	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	Rd – K	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	Stack ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← Stack	None	2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
4	2.7 - 5.5V	ATmega323L-4AC ATmega323L-4PC	44A 40P6	Commercial (0°C to 70°C)
		ATmega323L-4AI ATmega323L-4PI	44A 40P6	Industrial (-40°C to 85°C)
8	4.0 - 5.5V	ATmega323-8AC ATmega323-8PC	44A 40P6	Commercial (0°C to 70°C)
		ATmega323-8AI ATmega323-8PI	44A 40P6	Industrial (-40°C to 85°C)

Package Type	
44A	44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
40P6	40-pin, 0.600” Wide, Plastic Dual Inline Package (PDIP)

Problem Fix / Workaround

Select the Device ID Register of the ATmega323 (Either by issuing the IDCODE instruction or by entering the Test-Logic-Reset state of the TAP controller) to read out the contents of its Device ID Register and possibly data from succeeding devices of the scan chain. Note that data to succeeding devices cannot be entered during this scan, but data to preceding devices can. Issue the BYPASS instruction to the ATmega323 to select its Bypass Register while reading the Device ID Registers of preceding devices of the boundary scan chain. Never read data from succeeding devices in the boundary scan chain or upload data to the succeeding devices while the Device ID Register is selected for the ATmega323. Note that the IDCODE instruction is the default instruction selected by the Test-Logic-Reset state of the TAP-controller.

Alternative Problem Fix / Workaround

If the Device IDs of all devices in the boundary scan chain must be captured simultaneously (for instance if blind interrogation is used), the boundary scan chain can be connected in such way that the ATmega323 is the fist device in the chain. Update-DR will still not work for the succeeding devices in the boundary scan chain as long as IDCODE is present in the JTAG Instruction Register, but the Device ID registered cannot be uploaded in any case.