ISPC

TECNICATURA SUPERIOR EN TELECOMUNICACIONES

ELECTRÓNICA MICROCONTROLADA

Ing. JORGE E. MORALES
Téc. Sup. C. GONZALO VERA

Microcontrolador AVR: PERIFÉRICOS

GRUPO N°2

- DARIO ARRIOLA
- MARCOS JULIAN FINES
- DANIEL RODRIGUEZ
- NATALIA GALLIANI
- JEREMIAS CASTRO
- CARLA ARGENTINA WAYAR

· OSCILLATOR OVERVIEW ·

- **ATMega328 Clock Discussion**

AVR® Fuses are the locations in non-volatile memory that define the hardware configuration of an AVR device.

Microchip megaAVR® 8-bit microcontrollers have several clock source options, selectable via programming of the CKSEL Flash fuse bits.

The fuse bits can select one of:

- Low Power Crystal Oscillator
- Low Frequency Crystal Oscillator
- Internal 128 kHz RC Oscillator
- Calibrated Internal RC Oscillator, and
- External Clock.

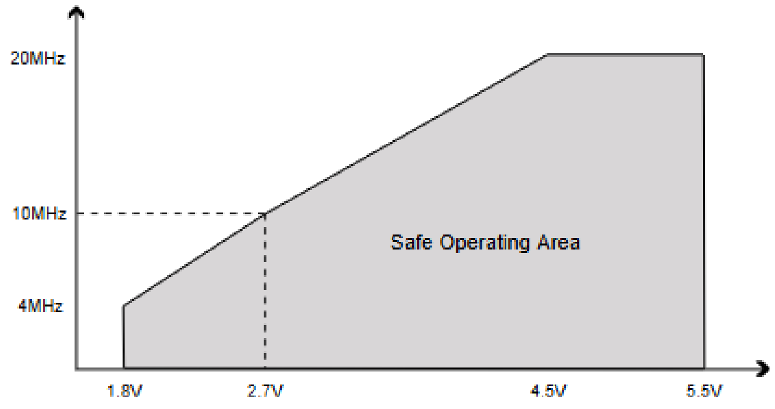
Note that:

- The ***system clock source cannot*** be changed during run-time, as it is set via fuse programming.
- The ***system clock frequency can*** be changed during run-time by writing to the **System Clock Prescaler** register (CLKPR).
- **megaAVR® maximum operating frequency is dependent on V_{CC}**. Application software must ensure that the selected clock source frequency falls within the safe operating area (see section 33.4 in the **device data sheet**).

Speed Grades

Maximum frequency is dependent on V_{CC}. As shown in Figure. Maximum Frequency vs. V_{CC}, the Maximum Frequency vs. V_{CC} curve is linear between $1.8V < V_{CC} < 2.7V$ and between $2.7V < V_{CC} < 4.5V$.

Figure 33-1. Maximum Frequency vs. V_{CC}



http://ww1.microchip.com/downloads/en/DeviceDoc/40001906A.pdf

- All the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes.

Clock Sources

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

For all fuses, '1' means unprogrammed while '0' means programmed

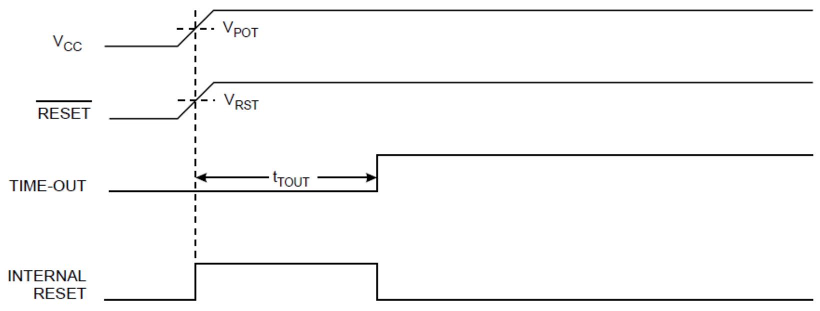
Default Clock Source

The device is shipped with the internal RC oscillator selected at 8.0 MHz and with the fuse CKDIV8 programmed, resulting in 1.0 MHz system clock. The start-up time is set to maximum, and the time-out period is enabled: CKSEL=0010, SUT=10, CKDIV8=0.

Clock Startup Sequence

Vcc Stability

To ensure sufficient V_{CC}, the device issues an internal reset with a time-out delay ($\mathbf{t}_{TOUT}$) after the device reset is released by all other reset sources:



The delay ($\mathbf{t}_{TOUT}$) is timed from the Watchdog Oscillator and the delay time is set by the SUTx and CKSELx fuse bits.

Typ. Time-out (V_{CC} = 5.0V)	Typ. Time-out (V_{CC} = 3.0V)
0ms	0ms
4ms	4.3ms
65ms	69ms

V_{CC} is not monitored during the delay, so it is required to select a delay longer than the V_{CC} rise time. If this is not possible, an internal or external Brown-Out Detection (BOD) circuit should be used. A BOD circuit will ensure sufficient V_{CC} before it releases the reset, and the time-out delay can be disabled. Disabling the time-out delay without utilizing a Brown-Out Detection circuit is not recommended.

Oscillator Stability

The oscillator is required to oscillate for a minimum number of cycles before the clock is considered stable. **The recommended oscillator start-up time is dependent on the clock type,** and varies from 6 cycles for an externally applied clock to 32K cycles for a low frequency crystal. [Section 11 of Data Sheet]

External Clock

To drive the device from an external clock source, EXTCLK 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'.**

Timer/Counter Oscillator

The device uses the same crystal oscillator for Low-frequency Oscillator and Timer/Counter Oscillator. The Timer/Counter Oscillator Pins (TOSC1 and TOSC2) are shared with EXTCLK. When using the Timer/Counter Oscillator, the system clock needs to be four times the oscillator frequency. Due to this and the pin sharing, the Timer/Counter Oscillator can only be used when the Calibrated Internal RC Oscillator is selected as system clock source. Applying an external clock source to TOSC1 can be done if the Enable External Clock Input bit in the Asynchronous Status Register (ASSR.EXCLK) is written to '1'.

System Clock Prescaler

The device has a system clock prescaler, and the system clock can be divided by configuring the Clock Prescale Register (CLKPR). This feature can be used to decrease the system clock frequency and the power consumption when the requirement for processing power is low. This can be used with all clock source options, and it will affect the clock frequency of the CPU and all synchronous peripherals. clk$_{I/O}$,clk$_{ADC}$, clk$_{CPU}$, and clk$_{FLASH}$ are divided by a factor as shown in the CLKPR description:

Name: CLKPR
Offset: 0x61
Reset: Refer to the bit description
Property: -

Bit	7	6	5	4	3	2	1	0
	CLKPCE				CLKPSn	CLKPSn	CLKPSn	CLKPSn
Access	R/W				R/W	R/W	R/W	R/W
Reset	0				x	x	x	x

Bit 7 – CLKPCE: Clock Prescaler Change Enable
The CLKPCE bit must be written to logic one to enable change of the CLKPS bits. The CLKPCE bit is only updated when the other bits in CLKPR are simultaneously written to zero. CLKPCE is cleared by hardware four cycles after it is written or when CLKPS bits are written. Rewriting the CLKPCE bit within this time-out period does neither extend the time-out period, nor clear the CLKPCE bit.

Bits 3:0 – CLKPSn: Clock Prescaler Select n [n = 3:0]
These bits define the division factor between the selected clock source and the internal system clock. These bits can be written run-time to vary the clock frequency to suit the application requirements. As the divider divides the master clock input to the MCU, the speed of all synchronous peripherals is reduced when a division factor is used. The division factors are given in the table below.

CLKPS[3:0]	Clock Division Factor
0000	1
0001	2
0010	4
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	Reserved
1010	Reserved
1011	Reserved
1100	Reserved
1101	Reserved
1110	Reserved
1111	Reserved

Writing to CLKPR

To avoid unintentional changes of clock frequency, a special write procedure must be followed to change the CLKPS bits: **1- Write the Clock Prescaler Change Enable (CLKPCE) bit to '1' and all other bits in CLKPR to zero: CLKPR=0x80. 2-Within four cycles, write the desired value to CLKPS[3:0] while writing a zero to CLKPCE: CLKPR=0x0N**

· AVR® USART OVERVIEW ·

- Microchip AVR® 8-bit microcontrollers contain a highly flexible communication peripheral known as a **USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter).**
- This peripheral can be used to communicate with a wide variety of other components, including other **microcontrollers, wireless modules, LCD displays, GPS modules, etc.** The USART peripheral can operate in one of two main modes: Synchronous or Asynchronous.

Key registers (highlighted in grey) include:
Control-and-Status Registers (**UCSRnA**, **UCSRnB**, **UCSRnC**) shared by all three sections.
Data register **UDRn** shared by both Transmitter and Receiver sections.
Baud rate control registers **UBRRn[H:L]** used by the Clock Generator.
"**n**" in the register/bit name identifies the specific USART hardware instance (0, 1, 2) the register/bit is associated to. For example **UCSR0A** refers to **USART0** Control & Status Register **A**

Using the USART (Summary)

For basic polled operation, the following minimum steps need to be performed:
1—> **Choose a baud rate and program the UBRRn[H:L] registers accordingly.**
2—> **Enable the USART serial transmit and receive sections.**
3—> **If transmitting, wait until the transmit shift register is empty (poll on UCSRnA.UDREn), then load your data byte into UDRn.**
4—> **If receiving, wait until the receiver data-received bit is set (poll on UCSRnA.RXCn), then read the data out of UDRn. Reading UDRn auto-clears the bit, and prepares the hardware to receive the next byte.**

Initialization

The USART has to be initialized before any communication can take place. The initialization process normally consists of:

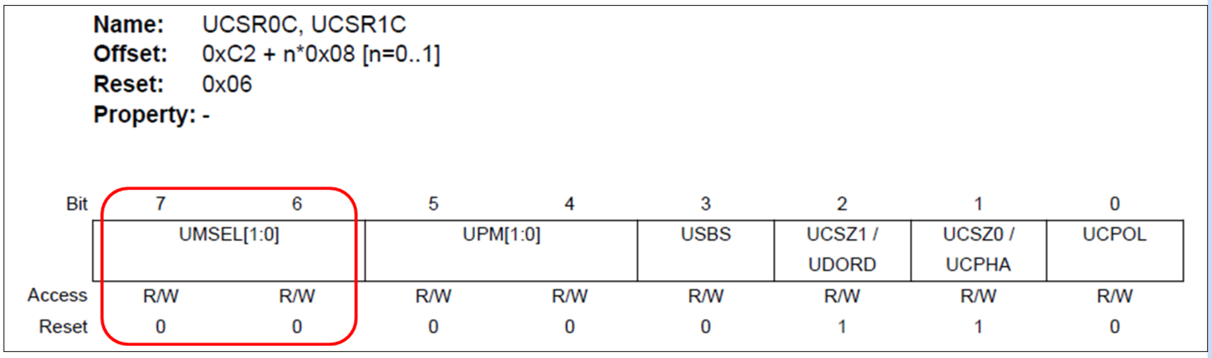
- Setting the baud rate,
- Setting frame format and
- Enabling the Transmitter or the Receiver depending on the usage.

Setting the Baud Rate

Internal clock generation is used for the asynchronous mode of operation. The Clock Generation logic generates the base clock for the Transmitter and Receiver (key registers and control bits are highlighted)

USART Mode Select (UMSELn)

The baud rate equation used by the module is set based on the operating mode. For asynchronous mode operation, the USART Mode Select bits in the USART Control and Status Register C (**UCSRnC.UMSELn[1:0]**) are used to select **asynchronous operation** (**UMSEL[1:0] = 00**) as shown:



Name: UCSR0C, UCSR1C
Offset: 0xC2 + n*0x08 [n=0..1]
Reset: 0x06
Property: -

Bit	7	6	5	4	3	2	1	0
	UMSEL[1:0]		UPM[1:0]		USBS	UCSZ1 / UDORD	UCSZ0 / UCPHA	UCPOL
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

Double-Speed Mode (U2Xn)

For asynchronous mode, the USART TX rate can be doubled by setting the U2Xn bit in the UCSRnA register (**UCSRnA.U2Xn = 1**).

With double-speed mode set, the Receiver will only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used.

The down-counter, running at system clock (f_{osc}), is loaded with the UBRRn value each time the counter has counted down to zero or when the UBRRnL Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= f_{osc}/(UBRRn+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, U2Xn and DDR_XCK bits.

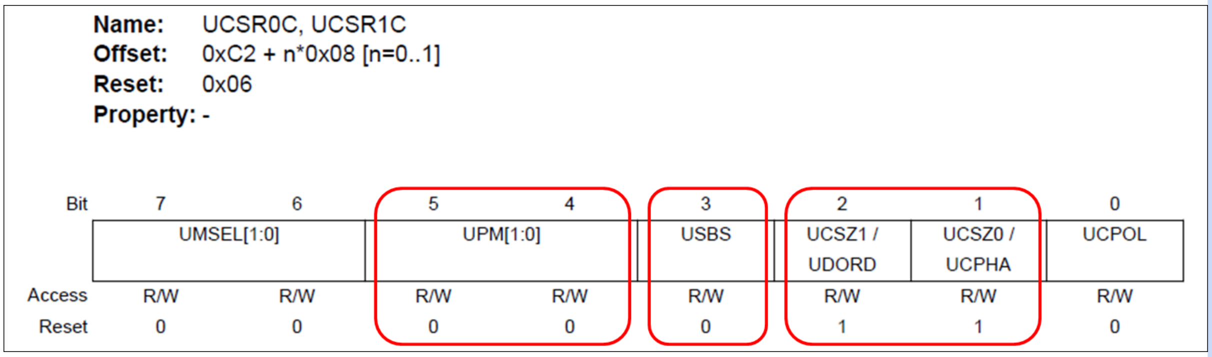
Table 24-1. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate(1)	Equation for Calculating UBRRn Value
Asynchronous Normal mode (U2Xn = 0)	$\mathrm{BAUD} = \dfrac{f_{\mathrm{OSC}}}{16(\mathbf{UBRR}n + 1)}$	$\mathbf{UBRR}n = \dfrac{f_{\mathrm{OSC}}}{16\mathrm{BAUD}} - 1$
Asynchronous Double Speed mode (U2Xn = 1)	$\mathrm{BAUD} = \dfrac{f_{\mathrm{OSC}}}{8(\mathbf{UBRR}n + 1)}$	$\mathbf{UBRR}n = \dfrac{f_{\mathrm{OSC}}}{8\mathrm{BAUD}} - 1$

The **AVR-LIBC Setbaud** library contains useful macros for calculating the correct values to write to UBRRnH and UBRRnL registers. See initialization code example below.

Setting the Frame Format

USART Control and Status Register C (**UCSRnC**) is used to configure the UART communication frame format - parity, number of stop bits, and number of data bits. Settings for the typical "8N1" frame format are as follows:

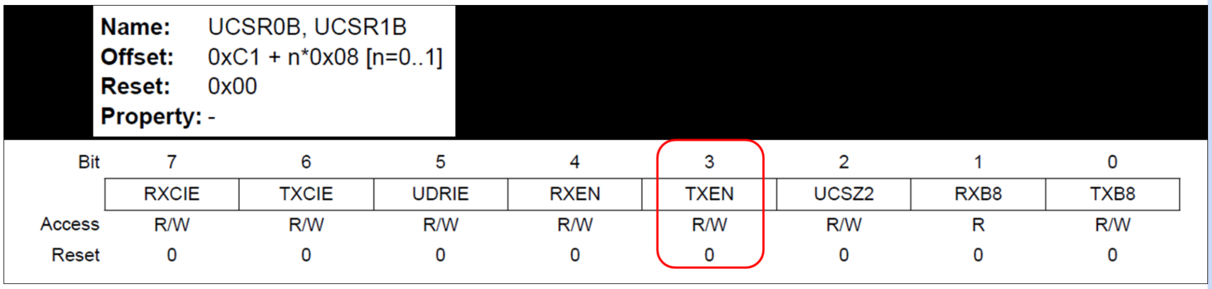


Name: UCSR0C, UCSR1C
Offset: 0xC2 + n*0x08 [n=0..1]
Reset: 0x06
Property: -

Bit	7	6	5	4	3	2	1	0
	UMSEL[1:0]		UPM[1:0]		USBS	UCSZ1 / UDORD	UCSZ0 / UCPHA	UCPOL
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

Enabling the Transmitter

The USART Transmitter is enabled by setting the **Transmit Enable (TXEN)** bit in the **UCSRnB** Register:



Name: UCSR0B, UCSR1B
Offset: 0xC1 + n*0x08 [n=0..1]
Reset: 0x00
Property: -

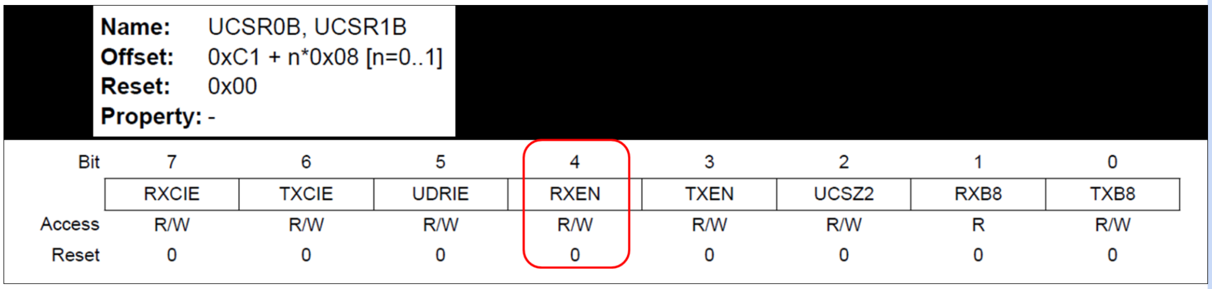
Bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

When the Transmitter is enabled, the normal port operation of the TxDn pin is overridden by the USART and given the function as the Transmitter's serial output.

The baud rate, mode of operation and frame format must be set up once before doing any transmissions.

Enabling the Receiver

The USART Receiver is enabled by writing the **Receive Enable (RXEN)** bit in the **UCSRnB** Register to '1':



Name: UCSR0B, UCSR1B
Offset: 0xC1 + n*0x08 [n=0..1]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Reset	0	0	0	0	0	0	0	0

When the Receiver is enabled, the normal port operation of the RxDn pin is overridden by the USART and given the function as the Receiver's serial input.

The baud rate, mode of operation and frame format must be set up once before doing any transmissions.

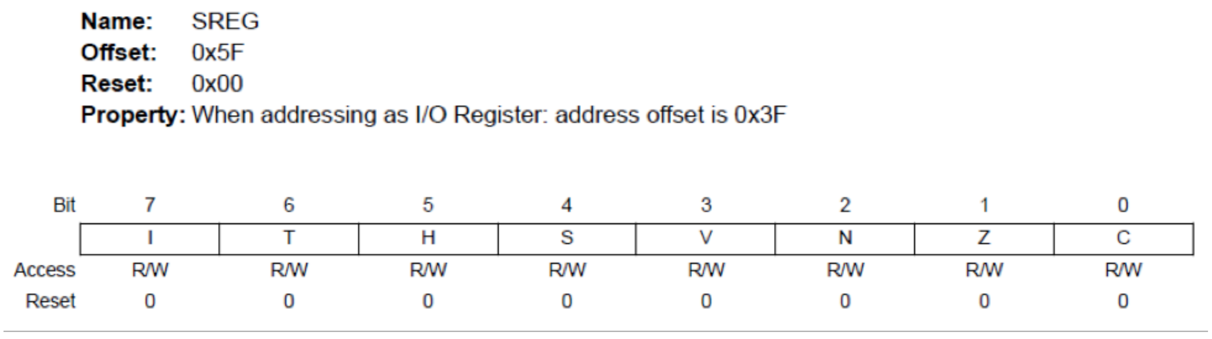
· megaAVR® Interrupts Overview ·

The megaAVR® family provides several different interrupt sources, all of which are maskable and are divided into three categories:

- **Internal Peripheral Interrupts**
  - Associated with Timers, USART, SPI, ADC peripherals
- **External Pin Interrupts**
  - Associated with the INT0-INT7 external interrupt pins

- **Pin Change Interrupts**
    - Associated with PCINT0-PCINT2 external interrupts occurring on a port pin change

Peripherals are assigned individual **interrupt enable bits** in their respective **interrupt mask register** which must be written as a logic one together with the **Global Interrupt Enable I-bit** in the **Status Register** in order to enable the interrupt.



Name: SREG
Offset: 0x5F
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x3F

Bit	7	6	5	4	3	2	1	0
	I	T	H	S	V	N	Z	C
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

Reset & Interrupt Vector Locations

The Reset & Interrupt sources each have a separate program vector in the **program memory space**. The lowest addresses in the program memory space are by default defined as the Reset and Interrupt Vectors as shown:

Vector Relocation

The user can relocate the RESET vector as well as the start location of the Interrupt vectors to the **Boot Flash Section** of program memory space by programming the **BOOTRST** fuse bit to "0" and setting the **IVSEL** bit of the Microcontroller Configuration Register (**MCUCR**) to "1". The possible RESET and interrupt vector placement are shown here:

BOOTRST	IVSEL	Reset Addr.	Interrupt Vector Start Addr.
1	0	0x0000	0x0002
1	1	0x0000	Boot Reset Addr. + 0x0002
0	0	Boot Reset Addr.	0x0002
0	1	Boot Reset Addr.	Boot Reset Addr. + 0x0002

The **Boot Reset Address** is set by BOOTSZ0/BOOTSZ1 fuse bits as shown here for ATmega328PB

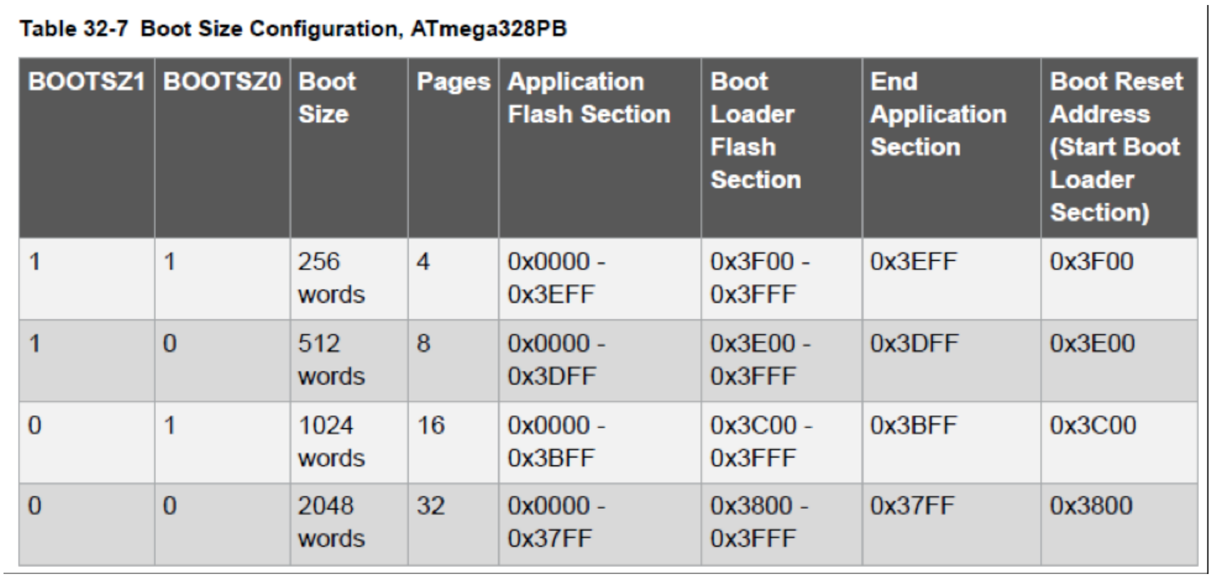


Table 32-7 Boot Size Configuration, ATmega328PB

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	256 words	4	0x0000 - 0x3EFF	0x3F00 - 0x3FFF	0x3EFF	0x3F00
1	0	512 words	8	0x0000 - 0x3DFF	0x3E00 - 0x3FFF	0x3DFF	0x3E00
0	1	1024 words	16	0x0000 - 0x3BFF	0x3C00 - 0x3FFF	0x3BFF	0x3C00
0	0	2048 words	32	0x0000 - 0x37FF	0x3800 - 0x3FFF	0x37FF	0x3800

- Fuses are programmed using a **special programming procedure** within Atmel Studio 7 or other programmer.
- To avoid unintentional changes of Interrupt Vector tables, a special write procedure must be followed to change the IVSEL bit:
    - Write the Interrupt Vector Change Enable (IVCE) bit to one.
    - Within four cycles, write the desired value to IVSEL while writing a zero to IVCE.

Priority Level

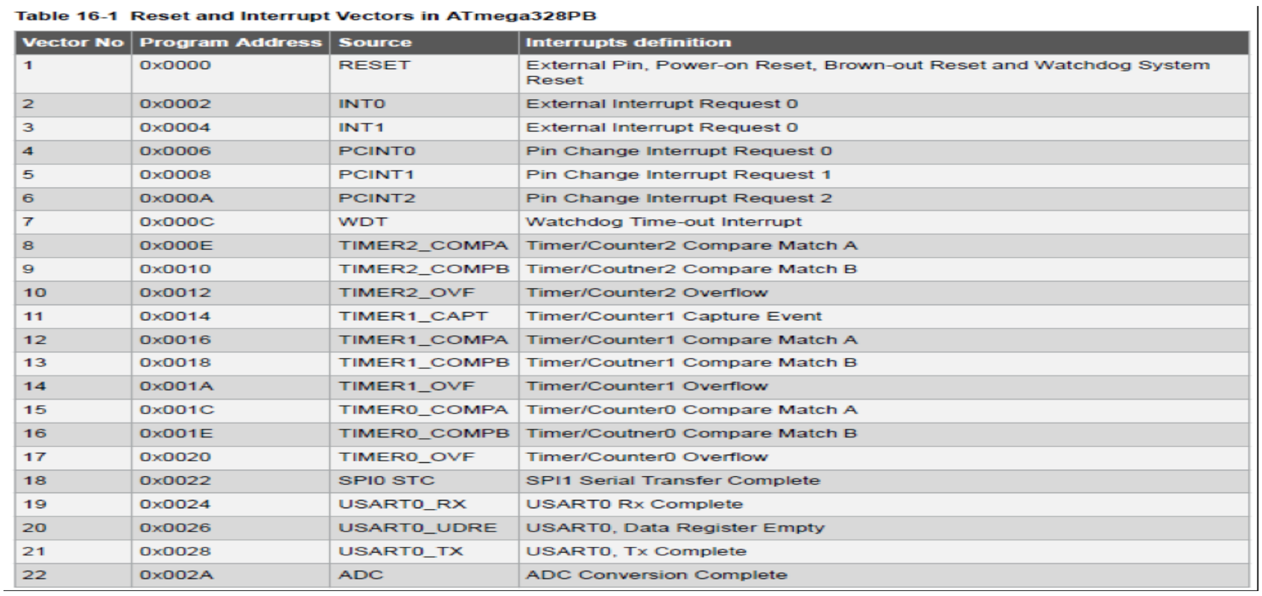


Table 16-1 Reset and Interrupt Vectors in ATmega328PB

Vector No	Program Address	Source	Interrupts definition
1	0x0000	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 0
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2_COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2_COMPB	Timer/Coutner2 Compare Match B
10	0x0012	TIMER2_OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1_CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1_COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1_COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1_OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0_COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0_COMPB	Timer/Coutner0 Compare Match B
17	0x0020	TIMER0_OVF	Timer/Counter0 Overflow
18	0x0022	SPI0 STC	SPI1 Serial Transfer Complete
19	0x0024	USART0_RX	USART0 Rx Complete
20	0x0026	USART0_UDRE	USART0, Data Register Empty
21	0x0028	USART0_TX	USART0, Tx Complete
22	0x002A	ADC	ADC Conversion Complete

Interrupt Processing

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared and all interrupts are disabled. The I-bit is automatically set when a Return from Interrupt instruction – RETI – is executed.

User software can write logic one to the I-bit to enable **nested interrupts**. All enabled interrupts can then interrupt the current interrupt routine.

There are basically two types of interrupts:

Persistent Interrupts

This type of interrupt will trigger as long as the interrupt condition is present. These interrupts do not necessarily have Interrupt Flags.

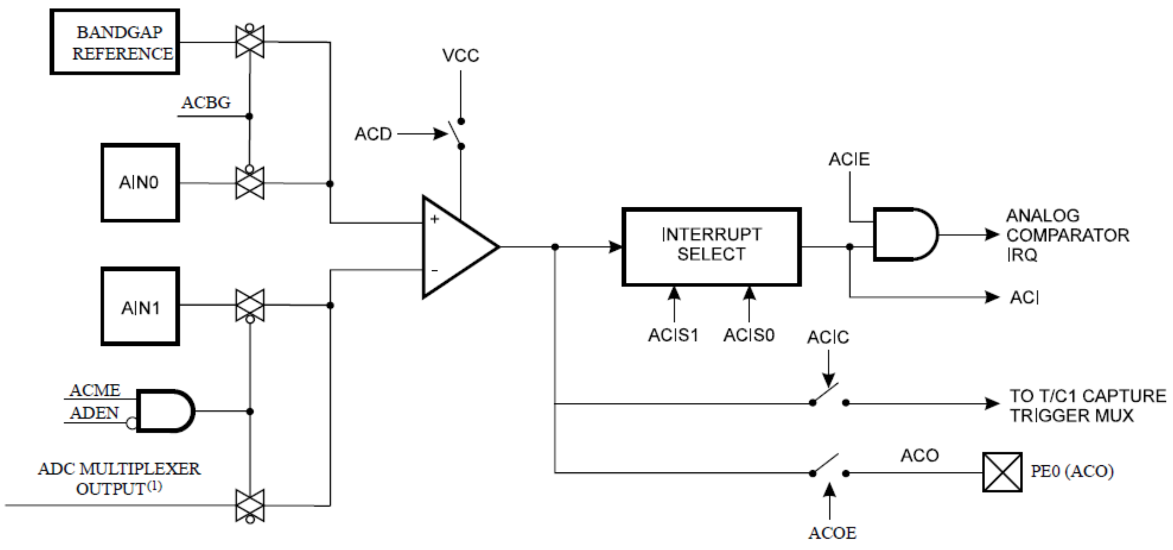
Non-Persistent Interrupts

This type of interrupt is triggered by an event that sets an **Interrupt Flag**. For these interrupts, the Program Counter is vectored to the actual Interrupt Vector in order to execute the interrupt handling routine, and **hardware clears the corresponding Interrupt Flag**. Interrupt Flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared.

· ANALOG COMPARATOR VOLTAGE REFERENCE 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 (on Port E[0]), 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.



· AVR® MCU TIMER OVERVIEW ·

Timers are a very useful feature of a microcontroller for counting pulses on an input pin. When driven by the instruction clock, they can become an accurate time base. AVR® devices have 8-bit- and 16-bit-wide timers and they offer different features based on the device. A very typical set of timers can be found in the AVR **ATmega328PB** microcontroller. This device has five Timer/Counters as described here:

Timer 0	TC0	8-bit Timer/counter with Pulse Width Modulation (PWM)
Timer 1	TC1	16-bit Timer/counter with PWM and Asynchronous Operation
Timer 2	TC2	8-bit Timer/counter with PWM and Asynchronous Operation
Timer 3	TC3	16-bit Timer/counter with PWM and Asynchronous Operation
Timer 4	TC4	16-bit Timer/counter with PWM and Asynchronous Operation

Definitions:

Register Nomenclature

BOTTOM	The counter reaches the BOTTOM when it becomes zero (0x00 for 8-bit counters and 0x0000 for 16-bit counters)
MAX	The counter reaches its maximum value when it becomes 0x0F (15 decimal) for 8-bit counters and 0x00FF (255 decimal) for 16-bit counters

TOP	The counter reaches the TOP when its value becomes equal to the highest value possible. The TOP value can be assigned a fixed value MAX or the value stored in the OCRxA register. This assignment is dependent upon the mode of operation

TC0 - 8-bit Timer/Counter with PWM

Timer/Counter0 (TC0) is a general purpose 8-bit Timer/Counter module, with two independent Output Compare Units and PWM support.

TC0 Registers

The Timer/Counter 0 register (TCNT0) and Output Compare TC0x registers (OCR0x) are 8-bit registers. Interrupt request signals are all visible in the Timer Interrupt Flag Register 0 (TIFR0). All interrupts are individually masked with the Timer Interrupt Mask Register 0 (TIMSK0).

Name: TCCR0B
Offset: 0x45
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x25

Bit	7	6	5	4	3	2	1	0
	FOC0A	FOC0B			WGM02	CS02	CS01	CS00
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

CS02	**CS01**	**CS00**	**Description**
0	0	0	No Clock source (Timer stopped)
0	0	1	clkio/1 (No prescaling)
0	1	0	clkio/8 (From prescaler)
0	1	1	clkio/64 (From prescaler)
1	0	0	clkio/256 (From prescaler)
1	0	1	clkio/1012 (From prescaler)
1	1	0	External clock source on T0 pin (clock on falling edge)
1	1	1	External clock source on T0 pin (clock on rising edge)

TC0 Counter Unit

Depending on the mode of operation used, T0 is cleared, incremented, or decremented at each timer clock (clkT0). clkT0 can be generated from an external or internal clock source, selected by the Clock Select bits (CS0[2:0]).

The counting sequence is determined by the setting of the WGM01 and WGM00 bits located in the T0 Control Register A (TCCR0A) and the WGM02 bit located in the Timer/Counter Control Register B (TCCR0B).

Table 1-2 Waveform Generation Mode Bit Description

Mode	WGM2:0	Mode of Operation	TOP	OCR0x Update	TOV flag set on
0	0 0 0	Normal	0xFF	Immediate	MAX
1	0 0 1	PWM Phase Correct	0xFF	TOP	BOTTOM
2	0 1 0	CTC	OCRA	Immediate	MAX
3	0 1 1	Fast PWM	0xFF	BOTTOM	MAX
4	1 0 0	Reserved	~	~	~
5	1 0 1	PWM Phase Correct	OCRA	TOP	BOTTOM
6	1 1 0	Reserved	~	~	~
7	1 1 1	Fast PWM	OCRA	BOTTOM	MAX

Note:1. MAX = 0xFF ; 2. BOTTOM = 0x00

Modes of Operation for TC0

The mode of operation determines the behavior of TC0 and the Output Compare pins. It is defined by the combination of the Waveform Generation mode bits and Compare Output mode bits in the Timer/Counter control Registers A and B (TCCR0B.WGMn2, TCCR0A.WGM01, TCCR0A.WGM00 and TCCR0A.COM0x[1:0]).

Available modes of operation for TC0 are:

- Normal Mode
- Clear Timer on Compare Match (CTC) Mode
- Fast PWM Mode
- Phase Correct PWM Mode

Clear Timer on Compare Match Mode

In Clear Timer on Compare or CTC mode (WGM0[2:0]=0x2), the **OCR0A Register** is used to manipulate the counter resolution: the counter is cleared to ZERO when the counter value (TCNT0) matches the OCR0A. The OCR0A defines the top value for the counter, hence also its resolution.

The counter value (TCNT0) increases until a compare match occurs between TCNT0 and OCR0A and then counter (TCNT0) is cleared. An interrupt can be generated each time the counter value reaches the TOP value by setting the OCF0A Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value.

The waveform frequency is defined by the following equation:

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

N represents the prescaler factor (1, 8, 64, 256, or 1024).

TC1, TC3, & TC4 - 16-bit Timer/Counters with PWM

The 16-bit Timer/Counter units allow accurate program execution timing (event management), wave generation and signal timing measurement.

Registers (TC1, TC3, TC4)

- The Timer/Counter (TCNTn), Output Compare Registers (OCRA/B) and Input Capture Register (ICRn) are all 16-bit registers.
- The Timer/Counter Control Registers (TCCRnA/B) are 8-bit registers and have no CPU access restrictions.
- Interrupt requests (abbreviated to Int.Req. in the block diagram) signals are all visible in the Timer Interrupt Flag Register (TIFRn). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSKn).

Name: TCCR1B, TCCR3B, TCCR4B
Offset: 0x81 + n*0x10 [n=0..2]
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	ICNC	ICES		WGM3	WGM2		CS[2:0]	
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Bits 2:0 – CS[2:0]: Clock Select [n = 0..2]

The three Clock Select bits select the clock source to be used by the Timer/Counter

CS02	CS01	CS00	Description	CS02
0	0	0	No Clock source (Timer stopped)	0
0	0	1	clkio/1 (No prescaling)	0
0	1	0	clkio/8 (From prescaler)	0
0	1	1	clkio/64 (From prescaler)	0
1	0	0	clkio/256 (From prescaler)	1
1	0	1	clkio/1012 (From prescaler)	1
1	1	0	External clock source on T0 pin (clock on falling edge)	1
1	1	1	External clock source on T0 pin (clock on rising edge)	1

TC2 - 8-bit Timer/Counter2 with PWM and Asynchronous Operation

Timer/Counter2 (TC2) is a general purpose, dual channel, 8-bit Timer/Counter module.

TC2 Registers

The Timer/Counter (TCNT2) and Output Compare Register (OCR2A and OCR2B) are 8-bit registers. Interrupt request signals are all visible in the Timer Interrupt Flag Register (TIFR2). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK2).

Name: TCCR2B
Offset: 0xB1
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	FOC2A	FOC2B			WGM22	CS22	CS21	CS20
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

TC2 Counter Unit

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clkT2). clkT2 can be generated from an external or internal clock source, selected by the Clock Select bits (CS2[2:0]).

The counting sequence is determined by the setting of the WGM21 and WGM20 bits located in the Timer/Counter Control Register (TCCR2A) and the WGM22 bit located in the Timer/Counter Control Register B (TCCR2B).

TC2 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the Waveform Generation mode (WGM2[2:0]) and Compare Output mode (COM2x[1:0]) bits.

Available modes of operation are:

- Normal Mode
- Clear Timer on Compare Match (CTC) Mode
- Fast PWM Mode
- Phase Correct PWM Mode

Normal Mode

In the Normal mode (WGM22:0 = 0) the counting direction is always up (incrementing) without having the counter cleared. The counter will roll over to 0c00 when it passes its maximum 8-bit value (TOP = 0xFF).