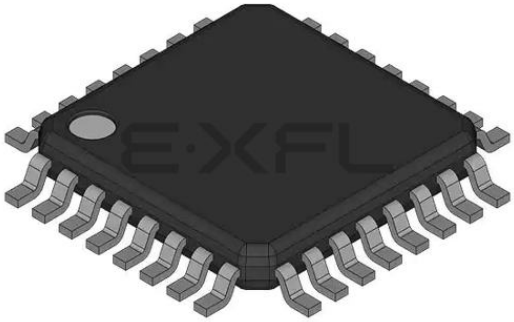


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	CANbus, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, WDT
Number of I/O	-
Program Memory Size	16KB (8K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 11x10b; D/A 1x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-TQFP
Supplier Device Package	32-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atmega16m1-15az

The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI ports, CAN, LIN/UART and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. The ADC noise reduction mode stops the CPU and all I/O modules except ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the boot flash section will continue to run while the application flash section is updated, providing true read-while-write operation. By combining an 8-bit RISC CPU with in-system self-programmable flash on a monolithic chip, the Atmel ATmega16/32/64/M1/C1 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega16/32/64/M1/C1 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Automotive Quality Grade

The Atmel® ATmega16/32/64/M1/C1 have been developed and manufactured according to the most stringent requirements of the international standard ISO-TS-16949. This data sheet contains limit values extracted from the results of extensive characterization (Temperature and Voltage). The quality and reliability of the ATmega16/32/64/M1/C1 have been verified during regular product qualification as per AEC-Q100 grade 1.

As indicated in the ordering information paragraph, the products are available in only one temperature grade.

Table 2-1. Temperature Grade Identification for Automotive Products

Temperature	Temperature Identifier	Comments
–40, +125	Z	Full automotive temperature range

2.3 Pin Descriptions

2.3.1 VCC

Digital supply voltage.

2.3.2 GND

Ground.

2.3.3 Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the Atmel ATmega16/32/64/M1/C1 as listed in Section 9.3.2 "Alternate Functions of Port B" on page 58.

2.3.4 Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves the functions of special features of the ATmega16/32/64/M1/C1 as listed in Section 9.3.3 "Alternate Functions of Port C" on page 61.

4.3 EEPROM Data Memory

The Atmel® ATmega16/32/64/M1/C1 contains 512/1024/2048 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

For a detailed description of SPI and Parallel data downloading to the EEPROM, see Section 25.9 “Serial Downloading” on page 270, and Section 25.6 “Parallel Programming Parameters, Pin Mapping, and Commands” on page 259 respectively.

4.3.1 EEPROM Read/Write Access

The EEPROM Access Registers are accessible in the I/O space.

The write access time for the EEPROM is given in Table 4-2. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write the EEPROM, some precautions must be taken. In heavily filtered power supplies, V_{CC} is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. Section 4.3.5 “Preventing EEPROM Corruption” on page 23 for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM Control Register for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

4.3.2 The EEPROM Address Registers – EEARH and EEARL

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	EEAR10	EEAR9	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	X	X	X	
	X	X	X	X	X	X	X	X	

- **Bits 15.11 – Reserved Bits**

These bits are reserved bits in the ATmega16/32/64/M1/C1 and will always read as zero.

- **Bits 9..0 – EEAR10..0: EEPROM Address**

The EEPROM address registers – EEARH and EEARL specify the EEPROM address in the 512/1024/2048 bytes EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 511/1023/2047. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

4.3.3 The EEPROM Data Register – EEDR

Bit	7	6	5	4	3	2	1	0	
	EEDR7	EEDR6	EEDR5	EEDR4	EEDR3	EEDR2	EEDR1	EEDR0	EEDR
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	

- **Bits 7..0 – EEDR7.0: EEPROM Data**

For the EEPROM write operation, the EEDR register contains the data to be written to the EEPROM in the address given by the EEAR register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

The next code examples show assembly and C functions for reading the EEPROM. The examples assume that interrupts are controlled so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_read:
        ; Wait for completion of previous write
        sbic            EECR,EEWE
        rjmp            EEPROM_read
        ; Set up address (r18:r17) in address register
        out             EEARH, r18
        out             EEARL, r17
        ; Start eeprom read by writing EERE
        sbi             EECR,EERE
        ; Read data from data register
        in              r16,EEDR
        ret
```

C Code Example

```
unsigned char EEPROM_read(unsigned int uiAddress)
{
        /* Wait for completion of previous write */
        while(EECR & (1<<EEWE))
                ;
        /* Set up address register */
        EEAR = uiAddress;
        /* Start eeprom read by writing EERE */
        EECR |= (1<<EERE);
        /* Return data from data register */
        return EEDR;
}
```

4.3.5 Preventing EEPROM Corruption

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

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

EEPROM data corruption can easily be avoided by following this design recommendation:

Keep the AVR® RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD). If the detection level of the internal BOD does not match the needed detection level, an external low V_{CC} reset Protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.

4.4 I/O Memory

The I/O space definition of the ATmega16/32/64/M1/C1 is shown in Section 29. “Register Summary” on page 299.

All Atmel® ATmega16/32/64/M1/C1 I/Os and peripherals are placed in the I/O space. All I/O locations may be accessed by the LD/LDS/LDD and ST/STS/STD instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range 0x00 - 0x1F 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 section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O registers as data space using LD and ST instructions, 0x20 must be added to these addresses. The Atmel ATmega16/32/64/M1/C1 is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

6.7 Minimizing Power Consumption

There are several issues to consider when trying to minimize the power consumption in an AVR® controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the device's functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

6.7.1 Analog to Digital Converter

If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. Refer to Section 18. "Analog to Digital Converter - ADC" on page 197 for details on ADC operation.

6.7.2 Analog Comparator

When entering Idle mode, the analog comparator should be disabled if not used. When entering ADC noise reduction mode, the analog comparator should be disabled. In other sleep modes, the analog comparator is automatically disabled. However, if the analog comparator is set up to use the internal voltage reference as input, the analog comparator should be disabled in all sleep modes. Otherwise, the internal voltage reference will be enabled, independent of sleep mode. Refer to Section 20. "Analog Comparator" on page 225 for details on how to configure the analog comparator.

6.7.3 Brown-out Detector

If the brown-out detector is not needed by the application, this module should be turned off. If the brown-out detector is enabled by the BODLEVEL fuses, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to Section 7.2.3 "Brown-out Detection" on page 40 for details on how to configure the brown-out detector.

6.7.4 Internal Voltage Reference

The internal voltage reference will be enabled when needed by the brown-out detection, the analog comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to Section 7.3 "Internal Voltage Reference" on page 42 for details on the start-up time.

6.7.5 Watchdog Timer

If the watchdog timer is not needed in the application, the module should be turned off. If the watchdog timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to Section 7.4 "Watchdog Timer" on page 43 for details on how to configure the watchdog timer.

6.7.6 Port Pins

When entering a sleep mode, all port pins should be configured to use minimum power. The most important is then to ensure that no pins drive resistive loads. In sleep modes where both the I/O clock ($clk_{I/O}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section Section 9. "I/O-Ports" on page 51 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or have an analog signal level close to $V_{CC}/2$, the input buffer will use excessive power.

For analog input pins, the digital input buffer should be disabled at all times. An analog signal level close to $V_{CC}/2$ on an input pin can cause significant current even in active mode. Digital input buffers can be disabled by writing to the digital input disable registers (DIDR1 and DIDR0). Refer to "Digital Input Disable Register 1– DIDR1" and "Digital Input Disable Register 0 – DIDR0" on 232 and 214 for details.

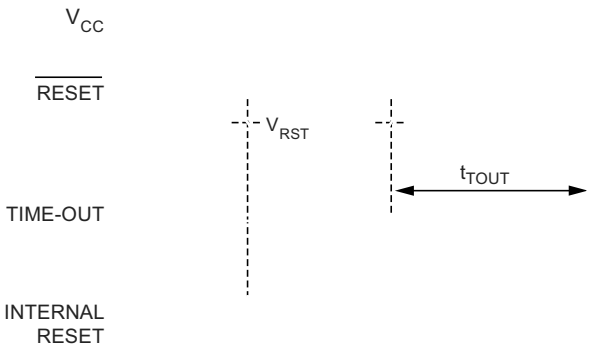
6.7.7 On-chip Debug System

If the on-chip debug system is enabled by OCDEN Fuse and the chip enter sleep mode, the main clock source is enabled, and hence, always consumes power. In the deeper sleep modes, this will contribute significantly to the total current consumption.

7.2.2 External Reset

An external reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than the minimum pulse width (see Table 7-1) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} – on its positive edge, the delay counter starts the MCU after the Time-out period – t_{TOUT} – has expired.

Figure 7-4. External Reset during Operation



7.2.3 Brown-out Detection

ATmega16/32/64/M1/C1 has an on-chip brown-out detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the BODLEVEL Fuses. The trigger level has a hysteresis to ensure spike free brown-out detection. The hysteresis on the detection level should be interpreted as $V_{BOT+} = V_{BOT} + V_{HYST}/2$ and $V_{BOT-} = V_{BOT} – V_{HYST}/2$.

Table 7-2. BODLEVEL Fuse Coding(1)(2)

BODLEVEL 2..0 Fuses	Typ V_{BOT}	Unit
111	Disabled	
110	4.5	V
011	4.4	V
100	4.3	V
010	4.2	V
001	2.8	V
101	2.7	V
000	2.6	V

Notes: 1. V_{BOT} may be below nominal minimum operating voltage for some devices. For devices where this is the case, the device is tested down to $V_{CC} = V_{BOT}$ during the production test. This guarantees that a brown-out reset will occur before V_{CC} drops to a voltage where correct operation of the microcontroller is no longer guaranteed. The test is performed using BODLEVEL = 010 for low operating voltage and BODLEVEL = 101 for high operating voltage.

2. Values are guidelines only.

Table 7-3. Brown-out Characteristics(1)

Parameter	Symbol	Min.	Typ.	Max.	Unit
Brown-out Detector Hysteresis	V_{HYST}		80		mV
Min Pulse Width on Brown-out Reset	t_{BOD}		2		µs

Note: 1. Values are guidelines only.

Table 9-4 and Table 9-5 relates the alternate functions of Port B to the overriding signals shown in Figure 9-5 on page 56.

Table 9-4. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/ADC4/ PSCOUT0B/SCK/ PCINT7	PB6/ADC7/ PSCOUT1B/ PCINT6	PB5/ADC6/ INT2/ACMPN1/ AMP2-/PCINT5	PB4/AMP0+/ PCINT4
PUOE	SPE × $\overline{\text{MSTR}}$ × $\overline{\text{SPIPS}}$	0	0	0
PUOV	PB7 × $\overline{\text{PUD}}$ × $\overline{\text{SPIPS}}$	0	0	0
DDOE	SPE × $\overline{\text{MSTR}}$ × $\overline{\text{SPIPS}}$ + PSCen01	PSCen11	0	0
DDOV	PSCen01	1	0	0
PVOE	SPE × MSTR × $\overline{\text{SPIPS}}$	PSCen11	0	0
PVOV	PSCout01 × SPIPS + $\overline{\text{PSCout01}}$ × PSCen01 × $\overline{\text{SPIPS}}$ + PSCout01 × PSCen01 × $\overline{\text{SPIPS}}$	PSCOUT11	0	0
DIEOE	ADC4D	ADC7D	ADC6D + In2en	AMP0ND
DIEOV	0	0	In2en	0
DI	SCKin × $\overline{\text{SPIPS}}$ × $\overline{\text{ireset}}$	ICP1B	INT2	
AIO	ADC4	ADC7	ADC6	AMP0+

Table 9-5. Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/AMP0-/ PCINT3	PB2/ADC5/INT1/ ACMPN0/PCINT2	PB1/MOSI/ PSCOUT2B/ PCINT1	PB0/MISO/ PSCOUT2A/ PCINT0
PUOE	0	0	–	–
PUOV	0	0	–	–
DDOE	0	0	–	–
DDOV	0	0	–	–
PVOE	0	0	–	–
PVOV	0	0	–	–
DIEOE	AMP0ND	ADC5D + In1en	0	0
DIEOV	0	In1en	0	0
DI		INT1	MOSI_IN × $\overline{\text{SPIPS}}$ × $\overline{\text{ireset}}$	MISO_IN × $\overline{\text{SPIPS}}$ × $\overline{\text{ireset}}$
AIO	AMP0-	ADC5	–	–

Table 9-7 and Table 9-8 relate the alternate functions of port C to the overriding signals shown in Figure 9-5 on page 56.

Table 9-7. Overriding Signals for Alternate Functions in PC7..PC4

Signal Name	PC7/D2A/AMP2+/ PCINT15	PC6/ADC10/ ACMP1/ PCINT14	PC5/ADC9/ AMP1+/ACMP3/ PCINT13	PC4/ADC8/ AMP1-/ACMPN3/ PCINT12
PUOE	0	0	0	
PUOV	0	0	0	
DDOE	DAEN	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	–
PVOV	0	0	0	–
DIEOE	DAEN	ADC10D	ADC9D	ADC8D
DIEOV	0	0	0	0
DI				
AIO	–	ADC10 Amp1	ADC9 Amp1+	ADC8 Amp1- ACMPN3

Table 9-8. Overriding Signals for Alternate Functions in PC3..PC0

Signal Name	PC3/T1/RXCAN/ ICP1B/PCINT11	PC2/T0/TXCAN/ PCINT10	PC1/PSCIN1/ OC1B/SS_A/ PCINT9	PC0/INT3/ PSCOUT1A/ PCINT8
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE			0	PSCen10
DDOV	1	1	0	1
PVOE			OC1Ben	PSCen10
PVOV			OC1B	PSCout10
DIEOE				In3en
DIEOV				In3en
DI	T1	T0	PSCin1 SS_A	INT3
AIO				

11. Timer/Counter0 and Timer/Counter1 Prescalers

Timer/Counter1 and Timer/Counter0 share the same prescaler module, but the Timer/Counters can have different prescaler settings. The description below applies to both Timer/Counter1 and Timer/Counter0.

11.1 Internal Clock Source

The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK\_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source. The prescaled clock has a frequency of either $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, or $f_{CLK\_I/O}/1024$.

11.2 Prescaler Reset

The prescaler is free running, i.e., operates independently of the clock select logic of the Timer/Counter, and it is shared by Timer/Counter1 and Timer/Counter0. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (6 > CSn2:0 > 1). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

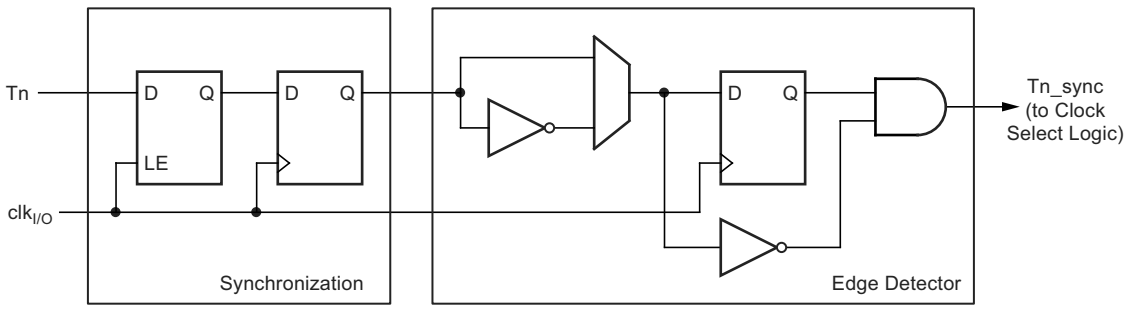
It is possible to use the prescaler reset for synchronizing the Timer/Counter to program execution. However, care must be taken if the other Timer/Counter that shares the same prescaler also uses prescaling. A prescaler reset will affect the prescaler period for all Timer/Counters it is connected to.

11.3 External Clock Source

An external clock source applied to the Tn pin can be used as Timer/Counter clock (clk_{T1}/clk_{T0}). The Tn pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 11-1 shows a functional equivalent block diagram of the Tn/T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

The edge detector generates one clk_{T1}/clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects.

Figure 11-1. Tn Pin Sampling



The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the Tn/T0 pin to the counter is updated.

Enabling and disabling of the clock input must be done when Tn/T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

Each half period of the external clock applied must be longer than one system clock cycle to ensure correct sampling. The external clock must be guaranteed to have less than half the system clock frequency ($f_{ExtClk} < f_{clk\_I/O}/2$) given a 50/50% duty cycle. Since the edge detector uses sampling, the maximum frequency of an external clock it can detect is half the sampling frequency (Nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by Oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk\_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 12-10 shows the setting of OCF0B in all modes and OCF0A in all modes except CTC mode and PWM mode, where OCR0A is TOP.

Figure 12-10.Timer/Counter Timing Diagram, Setting of OCF0x, with Prescaler ($f_{clk\_I/O}$/8)

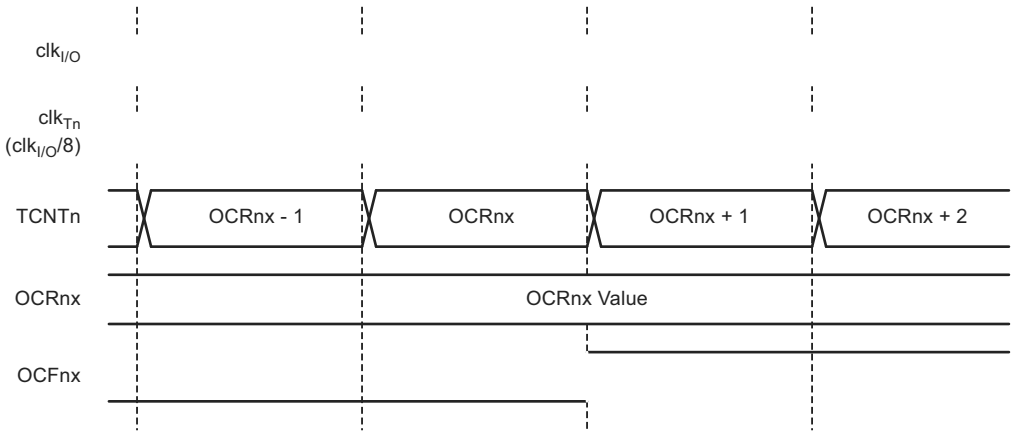
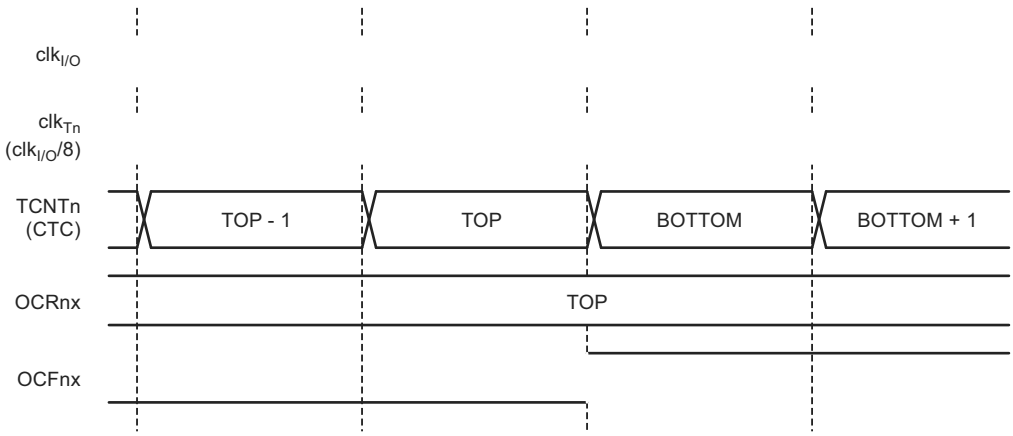


Figure 12-11 shows the setting of OCF0A and the clearing of TCNT0 in CTC mode and fast PWM mode where OCR0A is TOP.

Figure 12-11.Timer/Counter Timing Diagram, Clear Timer on Compare Match mode, with Prescaler ($f_{clk\_I/O}$/8)



12.8 8-bit Timer/Counter Register Description

12.8.1 Timer/Counter Control Register A – TCCR0A

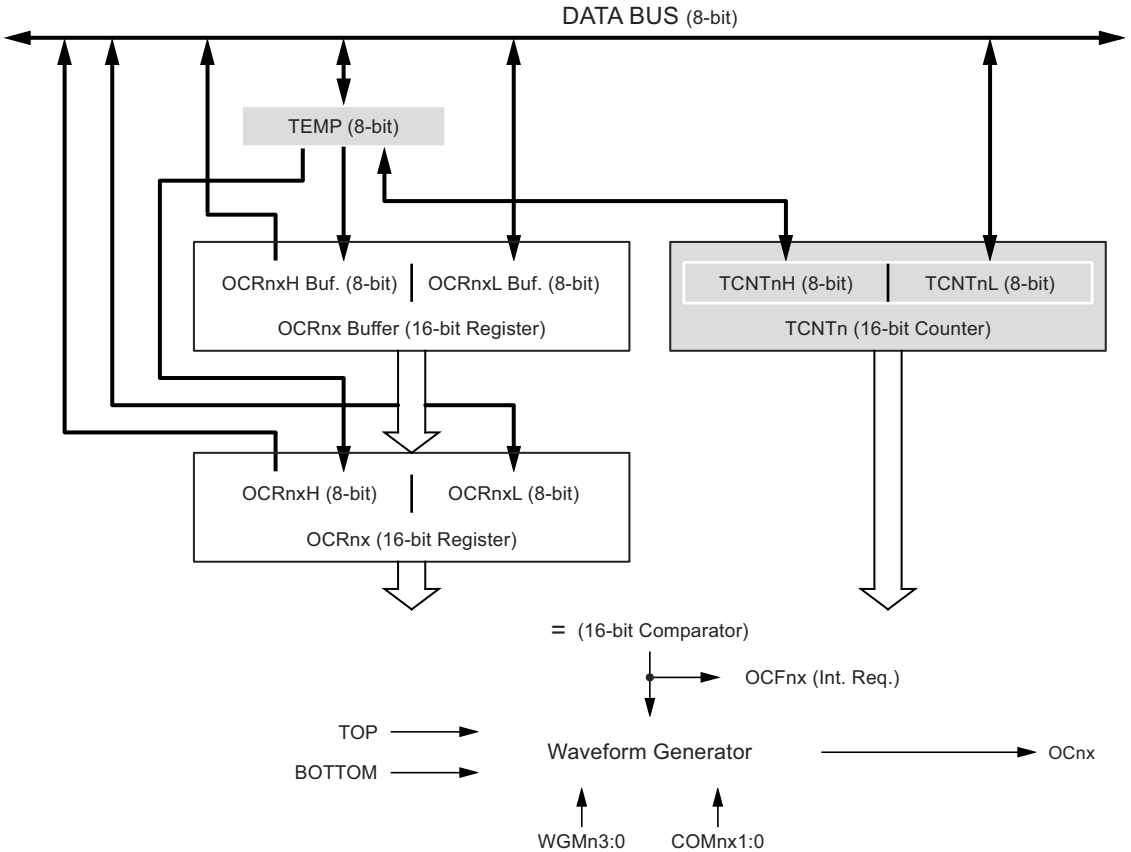
Bit	7	6	5	4	3	2	1	0	
	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – COM0A1:0: Compare Match Output A Mode**

These bits control the Output Compare pin (OC0A) behavior. If one or both of the COM0A1:0 bits are set, the OC0A 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 OC0A pin must be set in order to enable the output driver.

When OC0A is connected to the pin, the function of the COM0A1:0 bits depends on the WGM02:0 bit setting. Table 12-2 on page 87 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Figure 13-4. Output Compare Unit, Block Diagram



The OCRnx register is double buffered when using any of the twelve *Pulse Width Modulation* (PWM) modes. For the normal and *Clear Timer on Compare* (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCRnx compare register to either TOP or BOTTOM of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCRnx register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCRnx buffer register, and if double buffering is disabled the CPU will access the OCRnx directly. The content of the OCR1x (buffer or compare) register is only changed by a write operation (the Timer/Counter does not update this register automatically as the TCNT1 and ICR1 register). Therefore OCR1x is not read via the high byte temporary register (TEMP). However, it is a good practice to read the low byte first as when accessing other 16-bit registers. Writing the OCRnx registers must be done via the TEMP register since the compare of all 16 bits is done continuously. The high byte (OCRnxH) has to be written first. When the high byte I/O location is written by the CPU, the TEMP register will be updated by the value written. Then when the low byte (OCRnxL) is written to the lower eight bits, the high byte will be copied into the upper 8-bits of either the OCRnx buffer or OCRnx compare register in the same system clock cycle.

For more information of how to access the 16-bit registers refer to Section 13.2 “Accessing 16-bit Registers” on page 94.

13.6.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a one to the *Force Output Compare* (FOCnx) bit. Forcing compare match will not set the OCFnx Flag or reload/clear the timer, but the OCnx pin will be updated as if a real compare match had occurred (the COMn1:0 bits settings define whether the OCnx pin is set, cleared or toggled).

13.6.2 Compare Match Blocking by TCNTn Write

All CPU writes to the TCNTn register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCRnx to be initialized to the same value as TCNTn without triggering an interrupt when the Timer/Counter clock is enabled.

On-time 0 = 2 × POCRnSAH/L × 1/Fclkpsc

On-time 1 = 2 × (POCRnRBH/L – POCRnSBH/L + 1) × 1/Fclkpsc

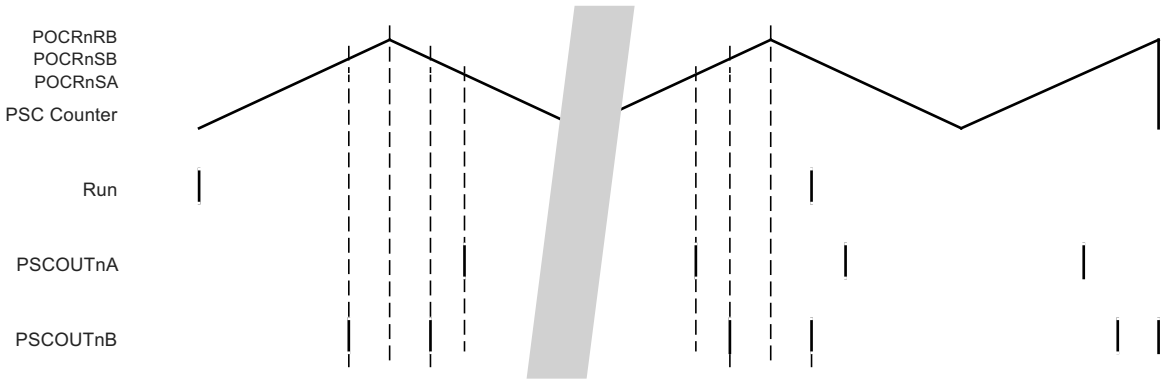
Dead-time = (POCRnSBH/L – POCRnSAH/L) × 1/Fclkpsc

PSC cycle = 2 × (POCRnRBH/L + 1) × 1/Fclkpsc

Minimal value for PSC cycle = 2 × 1/Fclkpsc

Note that in center aligned mode, POCRnRAH/L is not required (as it is in one-ramp mode) to control PSC Output waveform timing. This allows POCRnRAH/L to be freely used to adjust ADC synchronization (See Section 14.12 “Analog Synchronization” on page 126).

Figure 14-7. Controlled Start and Stop Mechanism in Centered Mode

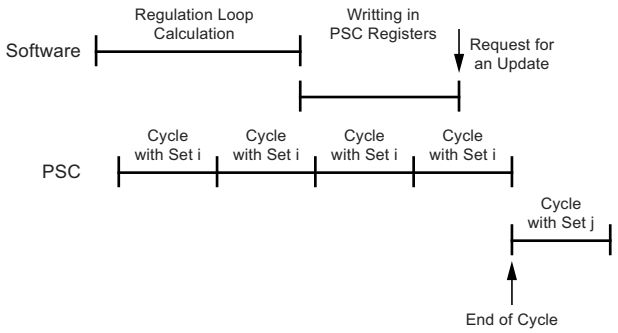


Note: See Section 14.16.8 “PSC Control Register – PCTL” on page 130 (PCCYC = 1)

14.6 Update of Values

To avoid unasynchronous and incoherent values in a cycle, if an update of one of several values is necessary, all values are updated at the same time at the end of the cycle by the PSC. The new set of values is calculated by software and the update is initiated by software.

Figure 14-8. Update at the End of Complete PSC Cycle



The software can stop the cycle before the end to update the values and restart a new PSC cycle.

The following code examples show how to initialize the SPI as a master and how to perform a simple transmission.

DDR_SPI in the examples must be replaced by the actual data direction register controlling the SPI pins. DD_MOSI, DD_MISO and DD_SCK must be replaced by the actual data direction bits for these pins. E.g. if MOSI is placed on pin PB2, replace DD_MOSI with DDB2 and DDR_SPI with DDRB.

Assembly Code Example[1]

```
SPI_MasterInit:
       ; Set MOSI and SCK output, all others input
       ldi                 r17,(1<<DD_MOSI)|(1<<DD_SCK)
       out                 DDR_SPI,r17
       ; Enable SPI, Master, set clock rate fck/16
       ldi                 r17,(1<<SPE)|(1<<MSTR)|(1<<SPR0)
       out                 SPCR,r17
       ret

SPI_MasterTransmit:
       ; Start transmission of data (r16)
       out                 SPDR,r16
Wait_Transmit:
       ; Wait for transmission complete
       sbis                SPSR,SPIF
       rjmp                Wait_Transmit
       ret
```

C Code Example[1]

```
void SPI_MasterInit(void)
{
       /* Set MOSI and SCK output, all others input */
       DDR_SPI = (1<<DD_MOSI)|(1<<DD_SCK);
       /* Enable SPI, Master, set clock rate fck/16 */
       SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0);
}

void SPI_MasterTransmit(char cData)
{
       /* Start transmission */
       SPDR = cData;
       /* Wait for transmission complete */
       while(!(SPSR & (1<<SPIF)))
       ;
}
```

Note: 1. The example code assumes that the part specific header file is included.

16.2.3.8 Bit Lengthening

As a result of resynchronization, phase segment 1 may be lengthened or phase segment 2 may be shortened to compensate for oscillator tolerances. If, for example, the transmitter oscillator is slower than the receiver oscillator, the next falling edge used for resynchronization may be delayed. So phase segment 1 is lengthened in order to adjust the sample point and the end of the bit time.

16.2.3.9 Bit Shortening

If, on the other hand, the transmitter oscillator is faster than the receiver one, the next falling edge used for resynchronization may be too early. So phase segment 2 in bit N is shortened in order to adjust the sample point for bit N+1 and the end of the bit time

16.2.3.10 Synchronization Jump Width

The limit to the amount of lengthening or shortening of the phase segments is set by the Resynchronization jump width.

This segment may not be longer than phase segment 2.

16.2.3.11 Programming the Sample Point

Programming of the sample point allows “tuning” of the characteristics to suit the bus.

Early sampling allows more time quanta in the phase segment 2 so the synchronization jump width can be programmed to its maximum. This maximum capacity to shorten or lengthen the bit time decreases the sensitivity to node oscillator tolerances, so that lower cost oscillators such as ceramic resonators may be used.

Late sampling allows more time quanta in the propagation time segment which allows a poorer bus topology and maximum bus length.

16.2.3.12 Synchronization

Hard synchronization occurs on the recessive-to-dominant transition of the start bit. The bit time is restarted from that edge.

Re-synchronization occurs when a recessive-to-dominant edge doesn't occur within the synchronization segment in a message.

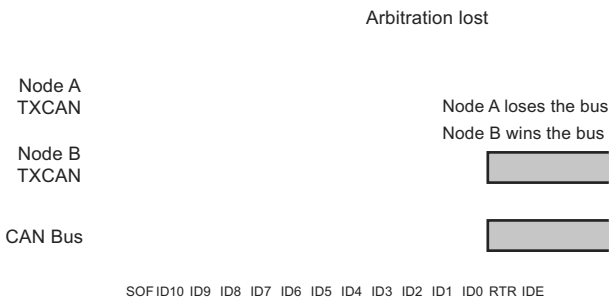
16.2.4 Arbitration

The CAN protocol handles bus accesses according to the concept called “Carrier Sense Multiple Access with Arbitration on Message Priority”.

during transmission, arbitration on the CAN bus can be lost to a competing device with a higher priority CAN Identifier. This arbitration concept avoids collisions of messages whose transmission was started by more than one node simultaneously and makes sure the most important message is sent first without time loss.

The bus access conflict is resolved during the arbitration field mostly over the identifier value. If a data frame and a remote frame with the same identifier are initiated at the same time, the data frame prevails over the remote frame (c.f. RTR bit).

Figure 16-4. Bus Arbitration



16.11.3 CAN Identifier Tag Registers - CANIDT1, CANIDT2, CANIDT3, and CANIDT4

V2.0 part A

Bit	15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0	
	-	-	-	-	-	RTRTAG	-	RB0TAG	CANIDT4
	-	-	-	-	-	-	-	-	CANIDT3
	IDT2	IDT1	IDT0	-	-	-	-	-	CANIDT2
	IDT10	IDT9	IDT8	IDT7	IDT6	IDT5	IDT4	IDT3	CANIDT1
Bit	31/23	30/22	29/21	28/20	27/19	26/18	25/17	24/16	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	-	-	-	-	-	-	-	-	

V2.0 part B

Bit	15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0	
	IDT4	IDT3	IDT2	IDT1	IDT0	RTRTAG	RB1TAG	RB0TAG	CANIDT4
	IDT12	IDT11	IDT10	IDT9	IDT8	IDT7	IDT6	IDT5	CANIDT3
	IDT20	IDT19	IDT18	IDT17	IDT16	IDT15	IDT14	IDT13	CANIDT2
	IDT28	IDT27	IDT26	IDT25	IDT24	IDT23	IDT22	IDT21	CANIDT1
Bit	31/23	30/22	29/21	28/20	27/19	26/18	25/17	24/16	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	-	-	-	-	-	-	-	-	

16.11.3.1 V2.0 part A

- **Bit 31:21 – IDT10:0: Identifier Tag**

Identifier field of the remote or data frame to send.

This field is updated with the corresponding value of the remote or data frame received.

- **Bit 20:3 – Reserved Bits**

These bits are reserved for future use. For compatibility with future devices, they must be written to zero when CANIDTn are written.

When a remote or data frame is received, these bits do not operate in the comparison but they are updated with un-predicted values.

- **Bit 2 – RTRTAG: Remote Transmission Request Tag**

RTR bit of the remote or data frame to send.

This tag is updated with the corresponding value of the remote or data frame received. In case of Automatic Reply mode, this bit is automatically reset before sending the response.

- **Bit 1 – Reserved Bit**

This bit is reserved for future use. For compatibility with future devices, it must be written to zero when CANIDTn are written.

When a remote or data frame is received, this bit does not operate in the comparison but it is updated with un-predicted values.

- **Bit 0 – RB0TAG: Reserved Bit 0 Tag**

RB0 bit of the remote or data frame to send.

This tag is updated with the corresponding value of the remote or data frame received.

17.4.4 LIN/UART Command Overview

Figure 17-5. LIN/UART Command Dependencies

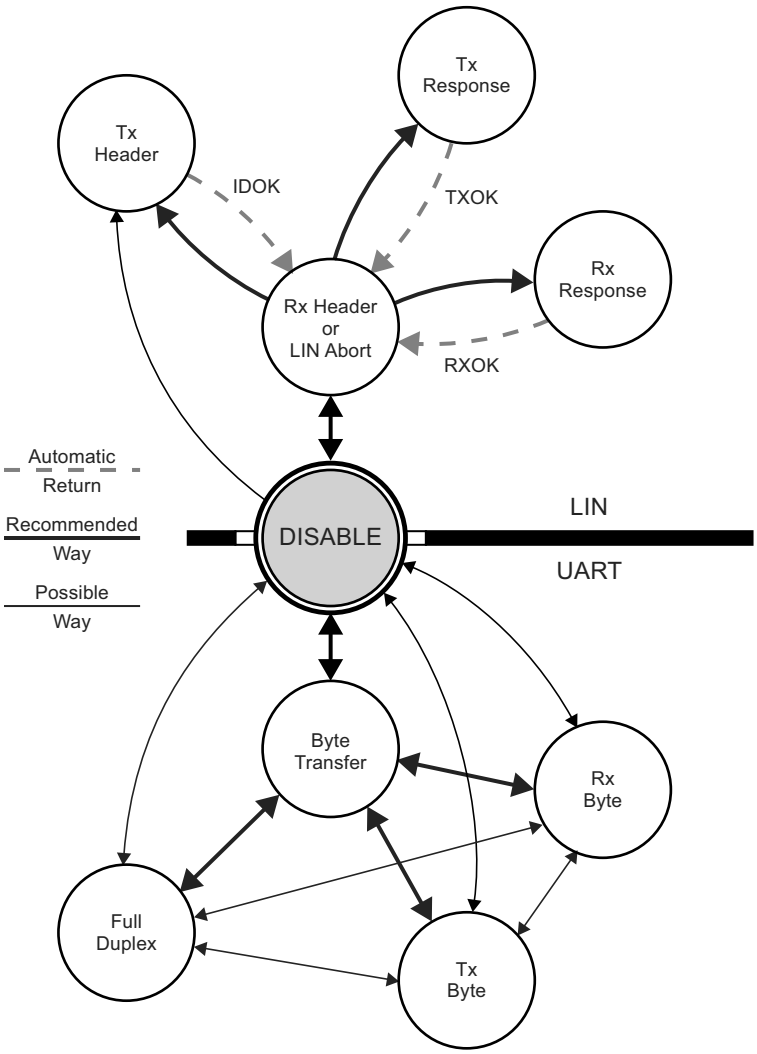
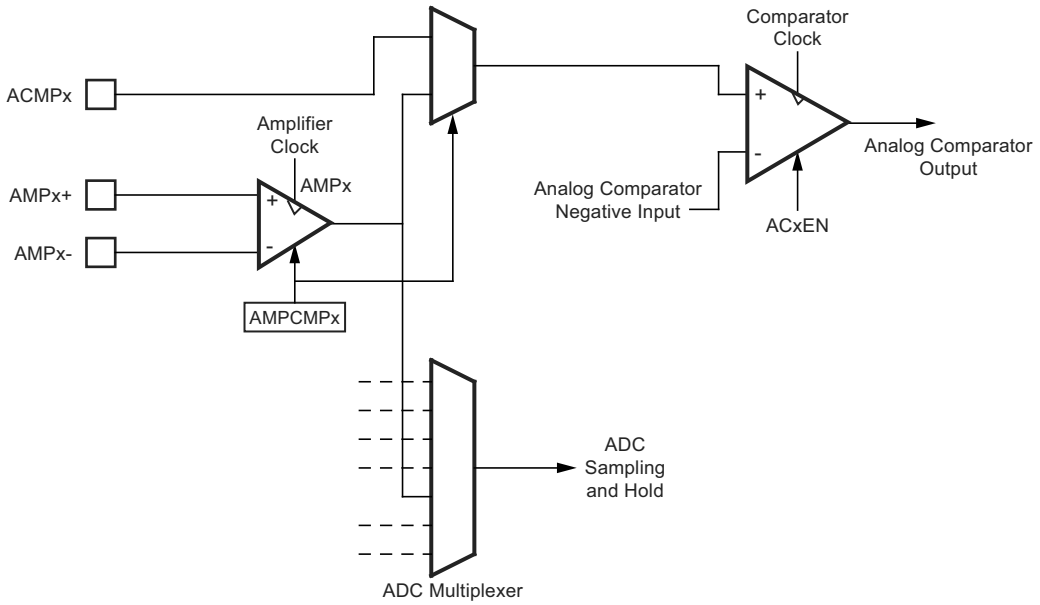


Table 17-1. LIN/UART Command List

LENA	LCMD[2]	LCMD[1]	LCMD[0]	Command	Comment
0	x	x	x	Disable peripheral	
1	0	0	0	Rx Header - LIN abort	LIN withdrawal
			1	Tx Header	LCMD[2..0]=000 after Tx
		1	0	Rx response	LCMD[2..0]=000 after Rx
			1	Tx response	LCMD[2..0]=000 after Tx
	1	0	0	Byte transfer	no CRC, no time out LTXDL=LRXDL=0 (LINDLR: read only register)
		1	0	Rx Byte	
		0	1	Tx Byte	
		1	1	Full duplex	

Figure 22-2. Amplifier, Comparator and ADC



22.4 Analog Peripheral Clock Sources

22.4.1 ADC Clock

The ADC clock comes from the clock system (CLKio) and it is divided by the ADC Prescaler. See Section 18-6 “ADC Prescaler Selection” on page 212 The bits described in the ADC Prescaler Selection determine the division factor between the system clock frequency and input clock of the ADC.

See Section 18.4 “Prescaling and Conversion Timing” on page 200 for a complete description of the ADC clock system.

22.4.2 Comparator Clock

While it is not connected to an amplifier, a comparator is clocked by the comparator clock which is configured thanks to the ACCKSEL bit in AC0CON register, see Section 20.4.1 “Analog Comparator 0 Control Register – AC0CON” on page 227. One can select between the 16MHz PLL output and the CLKio.

When it is connected to an amplifier, a comparator is clock by twice the amplifier clock.

22.4.3 Amplifier Clock

When the amplifier uses the ADC clock, this clock is divided by 8. This insures a maximum frequency of 250kHz for the amplifier when the ADC clock is 2MHz. When the ADC is clocked with a frequency higher than 2MHz the amplifier cannot be clocked by the ADC clock.

See Section 18.10 “Amplifier” on page 214 for a complete description of the Amplifier clock system.

25.2 Fuse Bits

The ATmega16/32/64/M1/C1 has three Fuse bytes. Table 25-4 to Table 25-7 on page 257 describe briefly the functionality of all the fuses and how they are mapped into the Fuse bytes. Note that the fuses are read as logical zero, “0”, if they are programmed.

Table 25-4. Extended Fuse Byte

Extended Fuse Byte	Bit No	Description	Default Value
-	7	-	1 (unprogrammed)
-	6	-	1 (unprogrammed)
PSCRB	5	PSC reset behavior	1 (unprogrammed)
PSCRVA	4	PSCOUTnA reset value	1 (unprogrammed)
PSCRVB	3	PSCOUTnB reset value	1 (unprogrammed)
BODLEVEL2(1)	2	Brown-out detector trigger level	1 (unprogrammed)
BODLEVEL1(1)	1	Brown-out detector trigger level	1 (unprogrammed)
BODLEVEL0(1)	0	Brown-out detector trigger level	1 (unprogrammed)

Note: 1. See Table 7-2 on page 40 for BODLEVEL fuse decoding.

25.3 PSC Output Behavior during Reset

For external component safety reason, the state of PSC outputs during reset can be programmed by fuses PSCRB, PSCARV and PSCBRV. These fuses are located in the extended fuse byte (see Table 25-4 on page 256).

If PSCRB fuse equals 1 (unprogrammed), all PSC outputs keep a standard port behavior. If PSC0RB fuse equals 0 (programmed), all PSC outputs are forced at reset to low level or high level according to PSCARV and PSCBRV fuse bits. In this second case, the PSC outputs keep the forced state until POC register is written. Section 5.10.1 “Clock Prescaler Register – CLKPR” on page 33

PSCARV (PSCOUTnA reset value) gives the state low or high which will be forced on PSCOUT0A, PSCOUT1A and PSCOUT2A outputs when PSCRB is programmed. If PSCARV fuse equals 0 (programmed), the PSCOUT0A, PSCOUT1A and PSCOUT2A outputs will be forced to high state. If PSCRV fuse equals 1 (unprogrammed), the PSCOUT0A, PSCOUT1A and PSCOUT2A outputs will be forced to low state.

PSCBRV (PSCOUTnB Reset Value) gives the state low or high which will be forced on PSCOUT0B, PSCOUT1B and PSCOUT2B outputs when PSCRB is programmed. If PSCBRV fuse equals 0 (programmed), the PSCOUT0B, PSCOUT1B and PSCOUT2B outputs will be forced to high state. If PSCRV fuse equals 1 (unprogrammed), the PSCOUT0B, PSCOUT1B and PSCOUT2B outputs will be forced to low state.

Table 25-5. PSC Output Behavior during and after Reset until POC Register is Written

PSCRB	PSCARV	PSCBRV	PSCOUTnA	PSCOUTnB
Unprogrammed	X	X	Normal port	Normal port
Programmed	Unprogrammed	Unprogrammed	Forced low	Forced low
Programmed	Unprogrammed	Programmed	Forced low	Forced high
Programmed	Programmed	Unprogrammed	Forced high	Forced low
Programmed	Programmed	Programmed	Forced high	Forced high
BODLEVEL2(1)		2	Brown-out detector trigger level	1 (unprogrammed)
BODLEVEL1(1)		1	Brown-out detector trigger level	1 (unprogrammed)
BODLEVEL0(1)		0	Brown-out detector trigger level	1 (unprogrammed)

27. ATmega16/32/64/M1/C1 Typical Characteristics

All DC characteristics contained in this datasheet are based on simulations and characterization of similar devices in the same process and design methods. These values are preliminary representing design targets, and will be updated after characterization of actual automotive silicon data.

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

All active- and idle current consumption measurements are done with all bits in the PRR register set and thus, the corresponding I/O modules are turned off. Also the analog comparator is disabled during these measurements. Table 27-1 on page 287 shows the additional current consumption compared to I_{CC} active and I_{CC} idle for every I/O module controlled by the power reduction register. See Section 6.6 “Power Reduction Register” on page 36 for details.

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

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

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

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

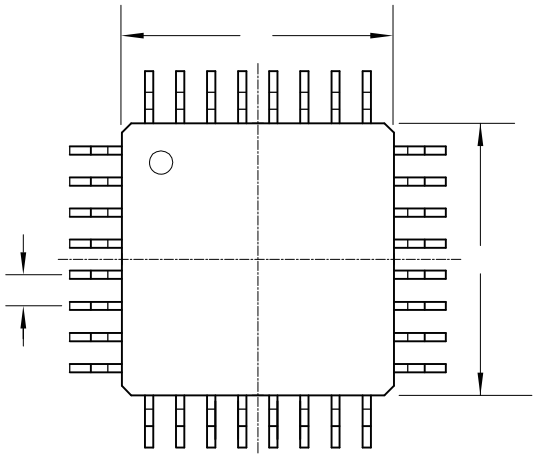
The difference between current consumption in power-down mode with watchdog timer enabled and power-down mode with watchdog timer disabled represents the differential current drawn by the watchdog timer.

27.1 Active Supply Current

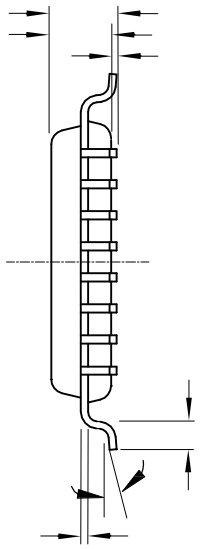
Figure 27-1. Active Supply Current versus Frequency (0.1 to 1.0MHz)

Figure 27-2. Active Supply Current versus Frequency (1 to 24MHz)

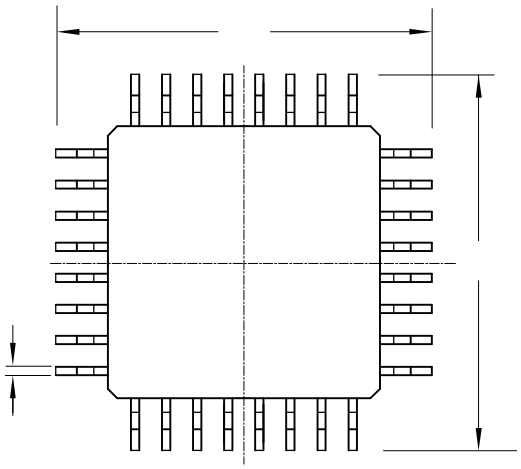
Figure 32-1. MA



Top View



Side View



Bottom View

COMMON DIMENSIONS (Unit of Measure = mm)				
Symbol	**MIN**	**NOM**	**MAX**	**NOTE**
A	----	----	1.20	
A1	0.05	----	0.15	
A2	0.95	1.00	1.05	
D/E	8.75	9.00	9.25	
D1/E1	6.90	7.00	7.10	2
C	0.09	----	0.20	
L	0.45	----	0.75	
b	0.30	----	0.45	
e	0.80 TYP.			
n	32			

02/29/12

Atmel Package Drawing Contact: packagedrawings@atmel.com	TITLE	GPC	DRAWING NO.	REV.
	MA, 32 Lds - 0.80mm Pitch, 7x7x1.00mm Body size Thin Profile Plastic Quad Flat Package (TQFP)	AUT	MA	C