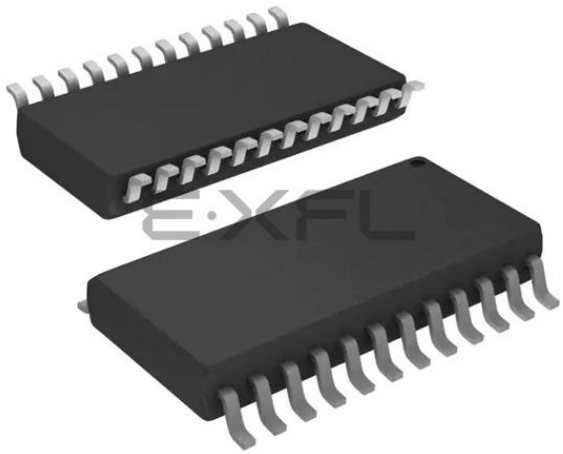


Welcome to E-XFL.COM

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	SPI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	19
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	24-SOIC (0.295", 7.50mm Width)
Supplier Device Package	24-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at90pwm1-16su

4.1 Block Diagram

Figure 4-1. Block Diagram

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The AT90PWM1 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 512 bytes SRAM, 53 general purpose I/O lines, 32 general purpose working registers, 2 Power Stage Controllers, two flexible Timer/Counters with compare modes and PWM, an 8-channel 10-bit ADC with two differential

6. Memories

This section describes the different memories in the AT90PWM1. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the AT90PWM1 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

6.1 In-System Reprogrammable Flash Program Memory

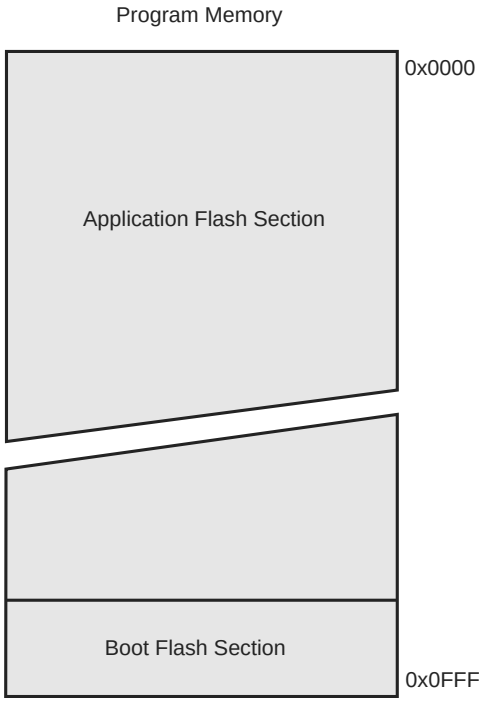
The AT90PWM1 contains 8K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 4K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The AT90PWM1 Program Counter (PC) is 12 bits wide, thus addressing the 4K program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in “Boot Loader Support – Read-While-Write Self-Programming” on page 205. “Memory Programming” on page 219 contains a detailed description on Flash programming in SPI or Parallel programming mode.

Constant tables can be allocated within the entire program memory address space (see the LPM – Load Program Memory.

Timing diagrams for instruction fetch and execution are presented in “Instruction Execution Timing” on page 13.

Figure 1. Program Memory Map



6.2 SRAM Data Memory

Figure 2 shows how the AT90PWM1 SRAM Memory is organized.

- **Bits 3..0 – CLKPS3..0: Clock Prescaler Select Bits 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 Table 7-11.

The CKDIV8 Fuse determines the initial value of the CLKPS bits. If CKDIV8 is unprogrammed, the CLKPS bits will be reset to “0000”. If CKDIV8 is programmed, CLKPS bits are reset to “0011”, giving a division factor of 8 at start up. This feature should be used if the selected clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. Note that any value can be written to the CLKPS bits regardless of the CKDIV8 Fuse setting. The Application software must ensure that a sufficient division factor is chosen if the selcted clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. The device is shipped with the CKDIV8 Fuse programmed.

Table 7-11. Clock Prescaler Select

CLKPS3	CLKPS2	CLKPS1	CLKPS0	Clock Division Factor
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	Reserved
1	0	1	0	Reserved
1	0	1	1	Reserved
1	1	0	0	Reserved
1	1	0	1	Reserved
1	1	1	0	Reserved
1	1	1	1	Reserved

8.1 Idle Mode

When the SM2..0 bits are written to 000, the SLEEP instruction makes the MCU enter Idle mode, stopping the CPU but allowing SPI, USART, Analog Comparator, ADC, Timer/Counters, Watchdog, and the interrupt system to continue operating. This sleep mode basically halt clk_{CPU} and clk_{FLASH}, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and USART Transmit Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the Analog Comparator can be powered down by setting the ACD bit in the Analog Comparator Control and Status Register – ACSR. This will reduce power consumption in Idle mode. If the ADC is enabled, a conversion starts automatically when this mode is entered.

8.2 ADC Noise Reduction Mode

When the SM2..0 bits are written to 001, the SLEEP instruction makes the MCU enter ADC Noise Reduction mode, stopping the CPU but allowing the ADC, the External Interrupts, Timer/Counter (if their clock source is external - T0 or T1) and the Watchdog to continue operating (if enabled). This sleep mode basically halts $clk_{I/O}$, clk_{CPU}, and clk_{FLASH}, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart from the ADC Conversion Complete interrupt, only an External Reset, a Watchdog Reset, a Brown-out Reset, a Timer/Counter interrupt, an SPM/EEPROM ready interrupt, an External Level Interrupt on INT3:0 can wake up the MCU from ADC Noise Reduction mode.

8.3 Power-down Mode

When the SM2..0 bits are written to 010, the SLEEP instruction makes the MCU enter Power-down mode. In this mode, the External Oscillator is stopped, while the External Interrupts and the Watchdog continue operating (if enabled). Only an External Reset, a Watchdog Reset, a Brown-out Reset, a PSC Interrupt, an External Level Interrupt on INT3:0 can wake up the MCU. This sleep mode basically halts all generated clocks, allowing operation of asynchronous modules only.

Note that if a level triggered interrupt is used for wake-up from Power-down mode, the changed level must be held for some time to wake up the MCU. Refer to “External Interrupts” on page 74 for details.

When waking up from Power-down mode, there is a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is defined by the same CKSEL fuses that define the Reset Time-out period, as described in “Clock Sources” on page 28.

8.4 Standby Mode

When the SM2..0 bits are 110 and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Standby mode. This mode is identical to Power-down

with the exception that the Oscillator is kept running. From Standby mode, the device wakes up in six clock cycles.

Table 4. Active Clock Domains and Wake-up Sources in the Different Sleep Modes.

	Active Clock Domains					Oscillators	Wake-up Sources					
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{PLL}	Main Clock Source Enabled	INT3..0	PSC	SPM/EEPROM Ready	ADC	WDT	OtherI/O
Idle			X	X	X	X	X	X	X	X	X	X
ADC Noise Reduction				X	X	X	X(2)	X	X	X	X	
Power-down							X(2)	X			X	
Standby(1)						X	X(2)				X	

Notes: 1. Only recommended with external crystal or resonator selected as clock source.
2. Only level interrupt.

8.5 Power Reduction Register

The Power Reduction Register, PRR, provides a method to stop the clock to individual peripherals to reduce power consumption. The current state of the peripheral is frozen and the I/O registers can not be read or written. Resources used by the peripheral when stopping the clock will remain occupied, hence the peripheral should in most cases be disabled before stopping the clock. Waking up a module, which is done by clearing the bit in PRR, puts the module in the same state as before shutdown.

A full predictible behaviour of a peripheral is not guaranteed during and after a cycle of stopping and starting of its clock. So its recommended to stop a peripheral before stopping its clock with PRR register.

Module shutdown can be used in Idle mode and Active mode to significantly reduce the overall power consumption. In all other sleep modes, the clock is already stopped.

8.5.1 Power Reduction Register - PRR

Bit	7	6	5	4	3	2	0	
	PRPSC2	**PRPSC1**	**PRPSC0**	**PRTIM1**	**PRTIM0**	**PRSPI**	**PRADC**	**PRR**
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	

- **Bit 7 - PRPSC2: Power Reduction PSC2**

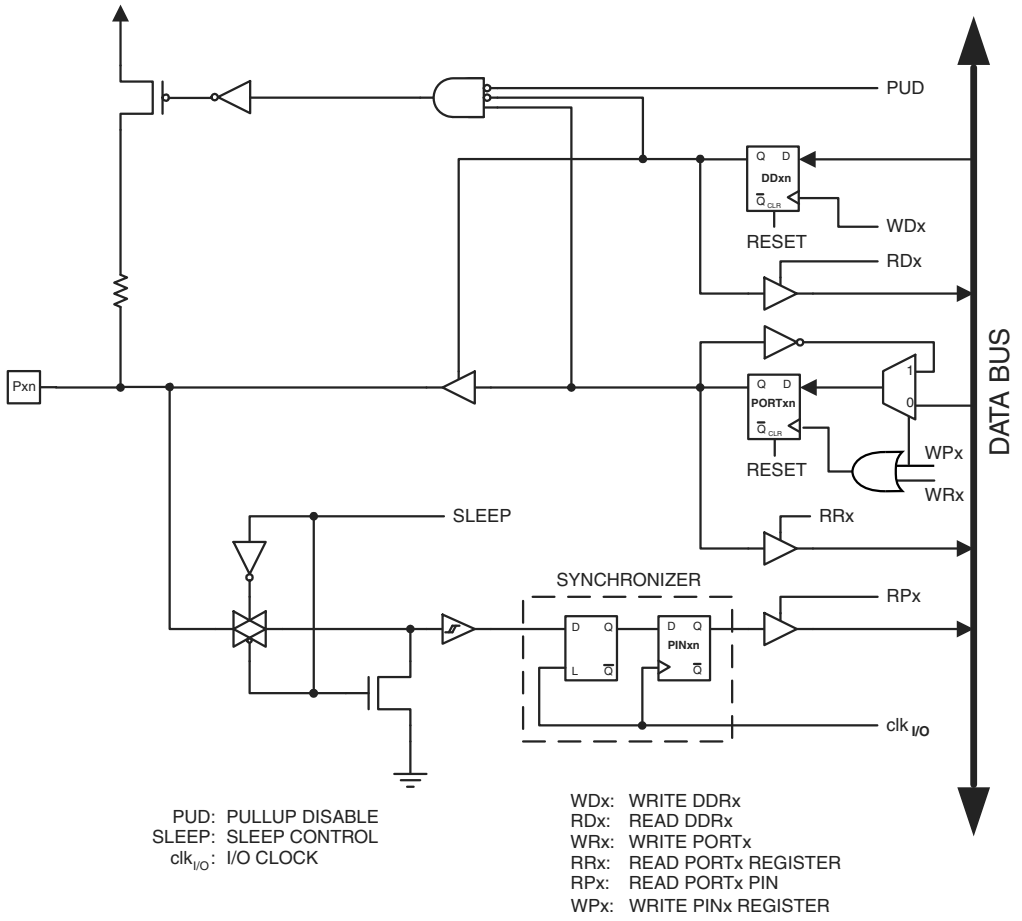
Writing a logic one to this bit reduces the consumption of the PSC2 by stopping the clock to this module. When waking up the PSC2 again, the PSC2 should be re initialized to ensure proper operation.

- **Bit 6 - PRPSC1: Power Reduction PSC1**

11.2 Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 11-2 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 11-2. General Digital I/O(1)



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

11.2.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in “Register Description for I/O-Ports” on page 72, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin

The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

TABLE 3.

Assembly Code Examples[1]

```
...
; Set TCNTn to 0x01FF
ldi r17,0x01
ldi r16,0xFF
out TCNTnH,r17
out TCNTnL,r16
; Read TCNTn into r17:r16
in  r16,TCNTnL
in  r17,TCNTnH
...
```

C Code Examples[1]

```
unsigned int i;
...
/* Set TCNTn to 0x01FF */
TCNTn = 0x1FF;
/* Read TCNTn into i */
i = TCNTn;
...
```

Note: 1. The example code assumes that the part specific header file is included.
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The assembly code example returns the TCNTn value in the r17:r16 register pair.

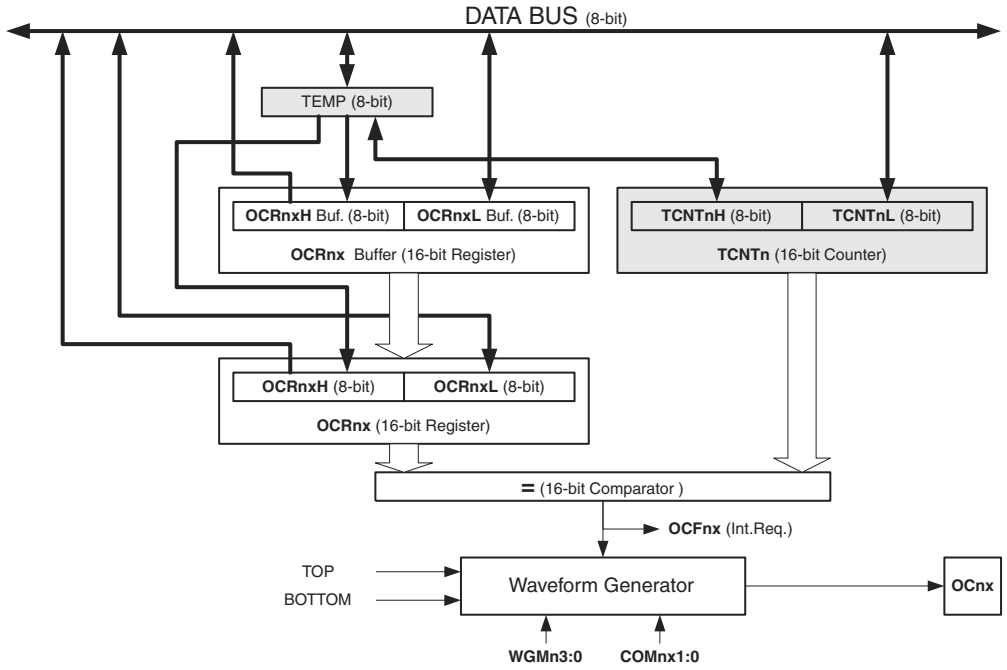
It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit Timer Registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

ing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the *Waveform Generation mode* (WGMn3:0) bits and *Compare Output mode* (COMnx1:0) bits. The TOP and BOTTOM signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (See “16-bit Timer/Counter1 with PWM” on page 96.)

A special feature of Output Compare unit A allows it to define the Timer/Counter TOP value (i.e., counter resolution). In addition to the counter resolution, the TOP value defines the period time for waveforms generated by the Waveform Generator.

Figure 15-4 shows a block diagram of the Output Compare unit. The small “n” in the register and bit names indicates the device number (n = n for Timer/Counter n), and the “x” indicates Output Compare unit (x). The elements of the block diagram that are not directly a part of the Output Compare unit are gray shaded.

Figure 15-4. 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

I/O pin it is connected to. However, note that the *Data Direction Register* (DDR) bit corresponding to the OCnA or OCnB pin must be set in order to enable the output driver.

When the OCnA or OCnB is connected to the pin, the function of the COMnx1:0 bits is dependent of the WGMn3:0 bits setting. Table 29 shows the COMnx1:0 bit functionality when the WGMn3:0 bits are set to a Normal or a CTC mode (non-PWM).

Table 29. Compare Output Mode, non-PWM

COMnA1/COMnB1	COMnA0/COMnB0	Description
0	0	Normal port operation, OCnA/OCnB disconnected.
0	1	Toggle OCnA/OCnB on Compare Match.
1	0	Clear OCnA/OCnB on Compare Match (Set output to low level).
1	1	Set OCnA/OCnB on Compare Match (Set output to high level).

Table 30 shows the COMnx1:0 bit functionality when the WGMn3:0 bits are set to the fast PWM mode.

Table 30. Compare Output Mode, Fast PWM(1)

COMnA1/COMnB1	COMnA0/COMnB0	Description
0	0	Normal port operation, OCnA/OCnB disconnected.
0	1	WGMn3:0 = 14 or 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OCnA/OCnB on Compare Match, set OCnA/OCnB at TOP
1	1	Set OCnA/OCnB on Compare Match, clear OCnA/OCnB at TOP

Note: 1. A special case occurs when OCRnA/OCRnB equals TOP and COMnA1/COMnB1 is set. In this case the compare match is ignored, but the set or clear is done at TOP. See “Fast PWM Mode” on page 109. for more details.

Table 31 shows the COMnx1:0 bit functionality when the WGMn3:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

Table 31. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM(1)

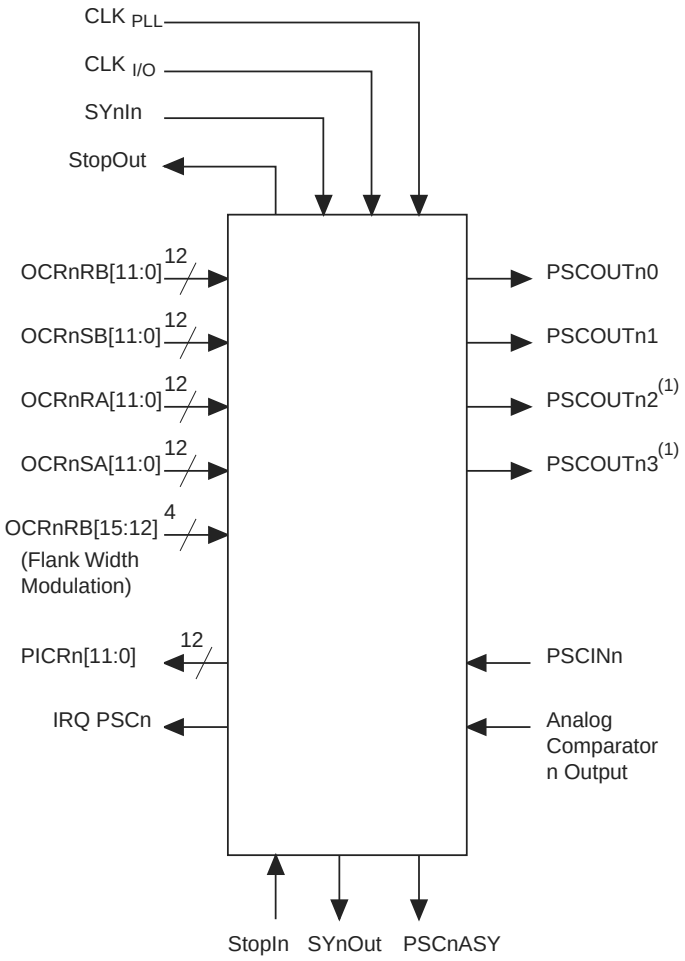
COMnA1/COMnB1	COMnA0/COMnB0	Description
0	0	Normal port operation, OCnA/OCnB disconnected.
0	1	WGMn3:0 = 8, 9 10 or 11: Toggle OCnA on Compare Match, OCnB disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OCnA/OCnB on Compare Match when up-counting. Set OCnA/OCnB on Compare Match when downcounting.
1	1	Set OCnA/OCnB on Compare Match when up-counting. Clear OCnA/OCnB on Compare Match when downcounting.

16.3.2 Output Polarity

The polarity “active high” or “active low” of the PSC outputs is programmable. All the timing diagrams in the following examples are given in the “active high” polarity.

16.4 Signal Description

Figure 16-3. PSC External Block View



Note: 1. available only for PSC2
2. n = 0, 1 or 2

16.20 Analog Synchronization

PSC generates a signal to synchronize the sample and hold; synchronisation is mandatory for measurements.

This signal can be selected between all falling or rising edge of PSCn0 or PSCn1 outputs.

In center aligned mode, OCRnRAH/L is not used, so it can be used to specified the synchronization of the ADC. It this case, it's minimum value is 1.

16.21 Interrupt Handling

As each PSC can be dedicated for one function, each PSC has its own interrupt system (vector ...)

List of interrupt sources:

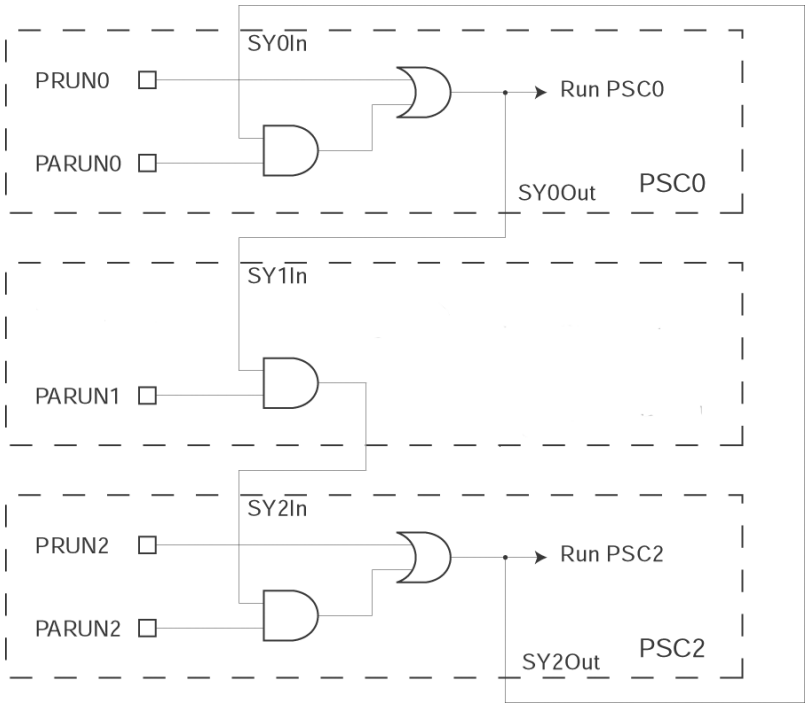
- Counter reload (end of On Time 1)
- PSC Input event (active edge or at the beginning of level configured event)
- PSC Mutual Synchronization Error

16.22 PSC Synchronization

2 or 3 PSC can be synchronized together. In this case, two waveform alignments are possible:

- The waveforms are center aligned in the Center Aligned mode if master and slaves are all with the same PSC period (which is the natural use).
- The waveforms are edge aligned in the 1, 2 or 4 ramp mode

Figure 16-38. PSC Run Synchronization



- **Bit 2 – POENnB: PSC n OUT Part B Output Enable**

When this bit is clear, I/O pin affected to PSCOUTn1 acts as a standard port.

When this bit is set, I/O pin affected to PSCOUTn1 is connected to the PSC waveform generator B output and is set and clear according to the PSC operation.

- **Bit 1 – POEN2C : PSCOUT22 Output Enable (PSC2 only)**

When this bit is clear, second I/O pin affected to PSCOUT22 acts as a standard port.

When this bit is set, second I/O pin affected to PSCOUT22 is connected to the PSC waveform generator A output and is set and clear according to the PSC operation.

- **Bit 0 – POENnA: PSC n OUT Part A Output Enable**

When this bit is clear, I/O pin affected to PSCOUTn0 acts as a standard port.

When this bit is set, I/O pin affected to PSCOUTn0 is connected to the PSC waveform generator A output and is set and clear according to the PSC operation.

16.25.3 Output Compare SA Register – OCRnSAH and OCRnSAL

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	OCRnSA[11:8]				OCRnSAH
	OCRnSA[7:0]								OCRnSAL
Read/Write	W	W	W	W	W	W	W	W	
Initial Value	0	0	0	0	0	0	0	0	

16.25.4 Output Compare RA Register – OCRnRAH and OCRnRAL

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	OCRnRA[11:8]				OCRnRAH
	OCRnRA[7:0]								OCRnRAL
Read/Write	W	W	W	W	W	W	W	W	
Initial Value	0	0	0	0	0	0	0	0	

TABLE 2.

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.

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

- **Bit 5 – ADATE: ADC Auto trigger Enable Bit**

Set this bit to enable the auto triggering mode of the ADC.
Clear it to return in single conversion mode.
In auto trigger mode the trigger source is selected by the ADTS bits in the ADCSRB register. See Table 19-1 on page 197.

- **Bit 4– ADIF: ADC Interrupt Flag**

Set by hardware as soon as a conversion is complete and the Data register are updated with the conversion result.
Cleared by hardware when executing the corresponding interrupt handling vector.
Alternatively, ADIF can be cleared by writing it to logical one.

- **Bit 3– ADIE: ADC Interrupt Enable Bit**

Set this bit to activate the ADC end of conversion interrupt.
Clear it to disable the ADC end of conversion interrupt.

- **Bit 2, 1, 0– ADPS2, ADPS1, ADPS0: ADC Prescaler Selection Bits**

These 3 bits determine the division factor between the system clock frequency and input clock of the ADC.
The different setting are shown in Table 64.

Table 64. ADC Prescaler Selection

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

19.8.3 ADC Control and Status Register B– ADCSRB

Bit	7	6	5	4	3	2	1	0	
	ADHSM	-	-	ADASCR	ADTS3	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	-	-	-	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ADHSM: ADC High Speed Mode**

Writing this bit to one enables the ADC High Speed mode. Set this bit if you wish to convert with an ADC clock frequency higher than 200KHz.

- **Bit 3, 2, 1, 0– ADTS3:ADTS0: ADC Auto Trigger Source Selection Bits**

These bits are only necessary in case the ADC works in auto trigger mode. It means if ADATE bit in ADCSRA register is set.

In accordance with the Table 19-1, these 3 bits select the interrupt event which will generate the trigger of the start of conversion. The start of conversion will be generated by the rising edge of the selected interrupt flag whether the interrupt is enabled or not. In case of trig on PSCnASY

These 2 bits determine the gain of the amplifier 0.
The different setting are shown in Table 19-3.

Table 19-3. Amplifier 0 Gain Selection

AMP0G1	AMP0G0	Description
0	0	Gain 5
0	1	Gain 10
1	0	Gain 20
1	1	Gain 40

To ensure an accurate result, after the gain value has been changed, the amplifier input needs to have a quite stable input value during at least 4 Amplifier synchronization clock periods.

- **Bit 1, 0– AMP0TS1, AMP0TS0: Amplifier 0 Trigger Source Selection Bits**

In accordance with the Table 19-4, these 2 bits select the event which will generate the trigger for the amplifier 0. This trigger source is necessary to start the conversion on the amplified channel.

Table 19-4. AMP0 Auto Trigger Source Selection

AMP0TS1	AMP0TS0	Description
0	0	Auto synchronization on ADC Clock/8
0	1	Trig on PSC0ASY
1	0	Reserved
1	1	Trig on PSC2ASY

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

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I_{CC}	Power Supply Current	Active 8 MHz, V_{CC} = 3V, RC osc, PRR = 0xFF		3.8	7	mA
		Active 16 MHz, V_{CC} = 5V, Ext Clock, PRR = 0xFF		14	24	mA
		Idle 8 MHz, V_{CC} = 3V, RC Osc		1.5	3	mA
		Idle 16 MHz, V_{CC} = 5V, Ext Clock		5.5	10	mA
	Power-down mode(5)	WDT enabled, V_{CC} = 3V t0 < 90°C		5	15	µA
		WDT enabled, V_{CC} = 3V t0 < 105°C		9	20	µA
		WDT disabled, V_{CC} = 3V t0 < 90°C		2	3	µA
		WDT disabled, V_{CC} = 3V t0 < 105°C		5	10	µA
V_{ACIO}	Analog Comparator Input Offset Voltage AT90PWM2/3	V_{CC} = 5V, V_{in} = 3V		20	50	mV
V_{hysr}	Analog Comparator Hysteresis Voltage AT90PWM2B/3B	V_{CC} = 5V, V_{in} = 3V Rising Edge Falling Edge	 33 34	 46 62	 71 110	 mV mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACID}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 5.0V		(6) (6)		ns

Note:
1. "Max" means the highest value where the pin is guaranteed to be read as low
2. "Min" means the lowest value where the pin is guaranteed to be read as high
3. Although each I/O port can sink more than the test conditions (20 mA at V_{CC} = 5V, 10 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   SO32, SO24 and TQFN Package:
   1] The sum of all IOL, for all ports, should not exceed 400 mA.
   2] The sum of all IOL, for ports B6 - B7, C0 - C1, D0 - D3, E0 should not exceed 100 mA.
   3] The sum of all IOL, for ports B0 - B1, C2 - C3, D4, E1 - E2 should not exceed 100 mA.
   4] The sum of all IOL, for ports B3 - B5, C6 - C7 should not exceed 100 mA.
   5] The sum of all IOL, for ports B2, C4 - C5, D5 - D7 should not exceed 100 mA.
   If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (20 mA at Vcc = 5V, 10 mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:
   SO32, SO24 and TQFN Package:
   1] The sum of all IOH, for all ports, should not exceed 400 mA.
   2] The sum of all IOH, for ports B6 - B7, C0 - C1, D0 - D3, E0 should not exceed 150 mA.
   3] The sum of all IOH, for ports B0 - B1, C2 - C3, D4, E1 - E2 should not exceed 150 mA.
   4] The sum of all IOH, for ports B3 - B5, C6 - C7 should not exceed 150 mA.
   5] The sum of all IOH, for ports B2, C4 - C5, D5 - D7 should not exceed 150 mA.
   If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.

24.7 BOD Thresholds and Analog Comparator Offset

Figure 24-29. BOD Thresholds vs. Temperature (BODLEVEL Is 4.3V)

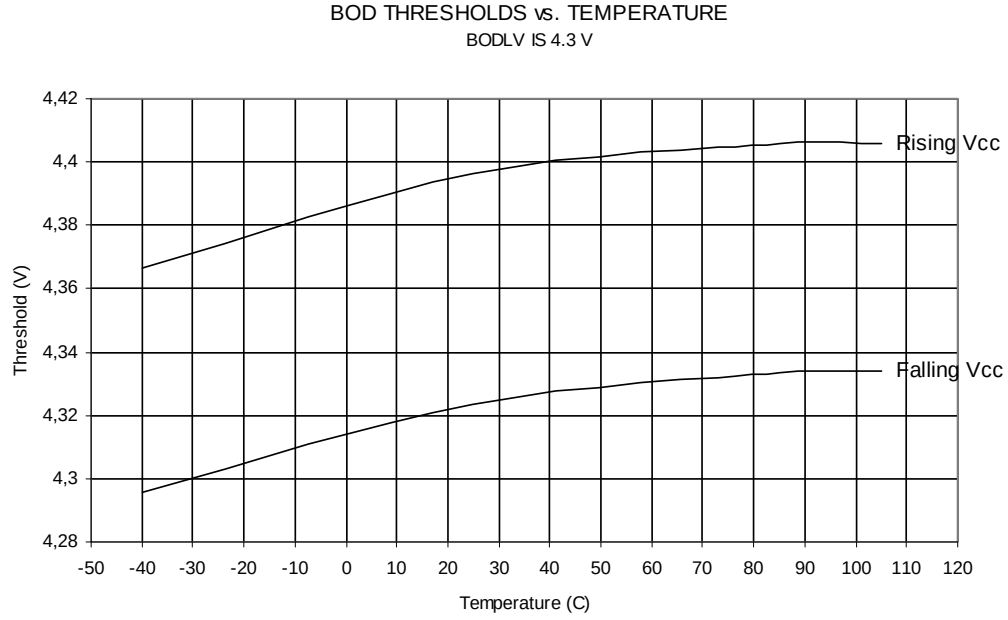
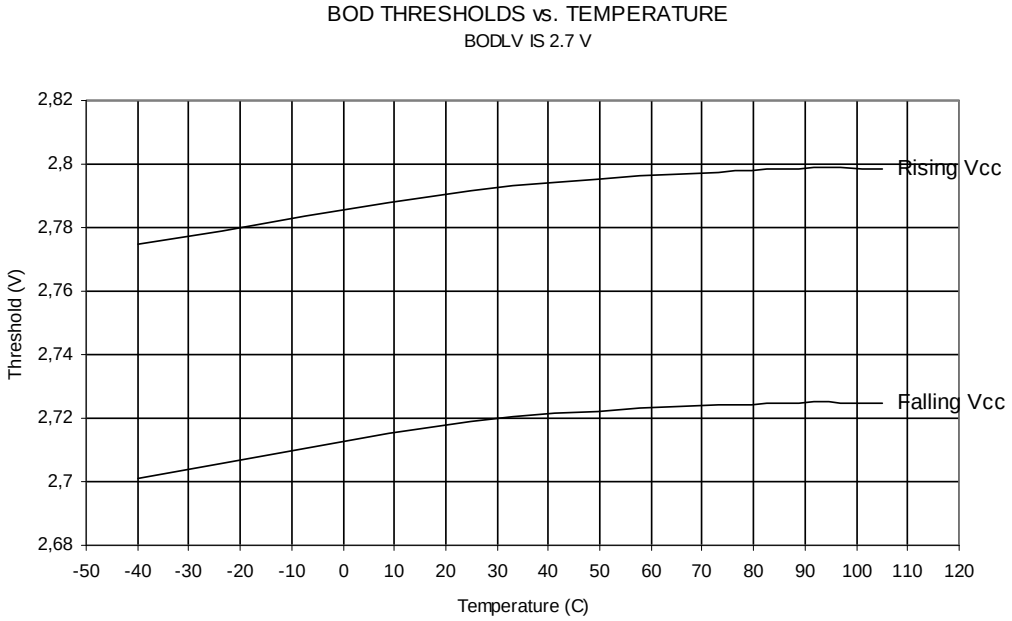
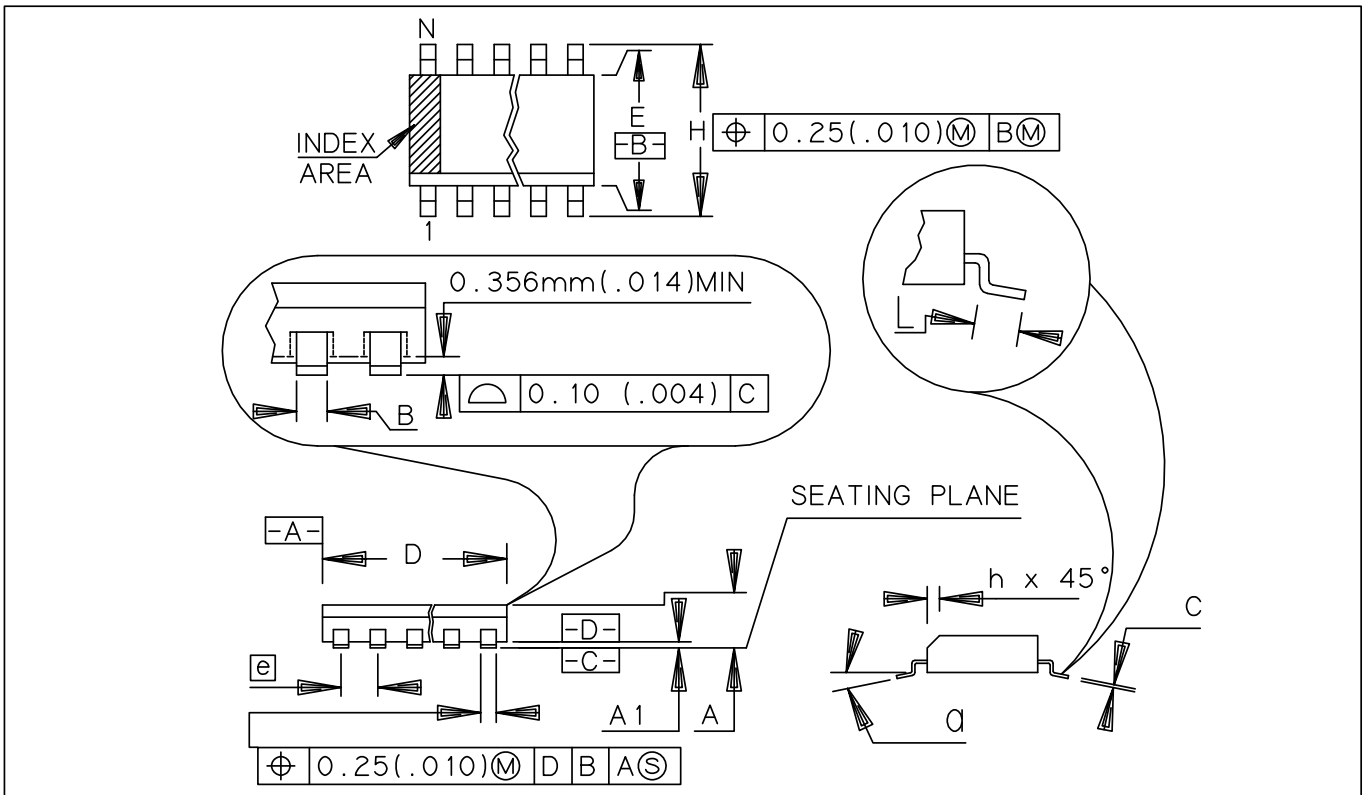


Figure 24-30. BOD Thresholds vs. Temperature (BODLEVEL Is 2.7V)



Mnemonics	Operands	Description	Operation	Flags	#Clocks
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
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
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
MCU CONTROL INSTRUCTIONS					

28.1 SO24



	MM		INCH	
A	2.35	2.65	.093	.104
A1	0.10	0.30	.004	.012
B	0.35	0.49	.014	.019
C	0.23	0.32	.009	.013
D	15.20	15.60	.599	.614
E	7.40	7.60	.291	.299
e	1.27	BSC	.050	BSC
H	10.00	10.65	.394	.419
h	0.25	0.75	.010	.029
L	0.40	1.27	.016	.050
N	24		24	
a	0°		8°	

07/27/07

Atmel Nantes S.A. La Chantrerie - BP 70602 44306 Nantes Cedex 3 - France	TITLE TD, 24 - Lead, 0.300" Body Width Plastic Gull Wing Small Outline Package (SOIC)	DRAWING No. TD	REV. A

29. Errata

- **PSC : Double End-Of-Cycle Interrupt Request in Centered Mode**
- **ADC : Conversion accuracy**
-

1. **PSC : Double End-Of-Cycle Interrupt Request in Centered Mode**

   In centered mode, after the “expected” End-Of-Cycle Interrupt, a second unexpected Interrupt occurs 1 PSC cycle after the previous interrupt.

   Work around:

   While CPU cycle is lower than PSC clock, the CPU sees only one interrupt request. For PSC clock period greater than CPU cycle, the second interrupt request must be cleared by software.

2. **ADC : Conversion accuracy**

   The conversion accuracy degrades when the ADC clock is 2 MHz.

   Work around:

   When a 10 bit conversion accuracy is required, use an ADC clock of 1 MHz or below.

   At 2 Mhz the ADC can be used as a 7 bits ADC.