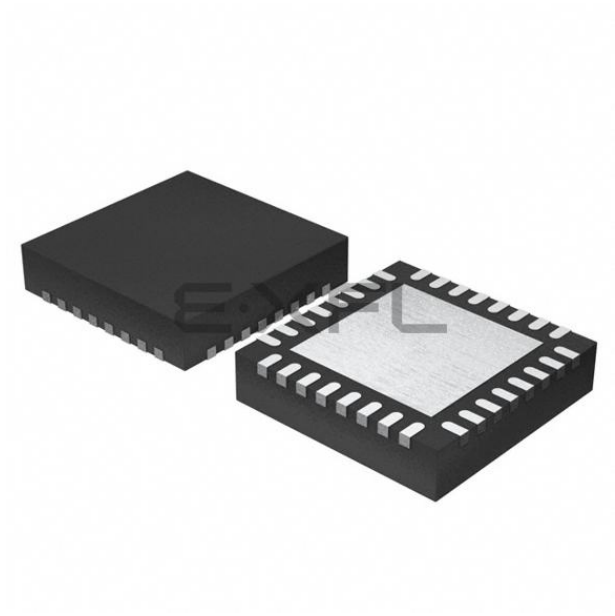


Welcome to E-XFL.COM

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, PWM, WDT
Number of I/O	20
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	256 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 ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-VQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at90pwm81-16mn

5. System Clock and Clock Options

The Atmel AT90PWM81/161 provides a large number of clock sources. Those can be divided in two categories: internal and external.

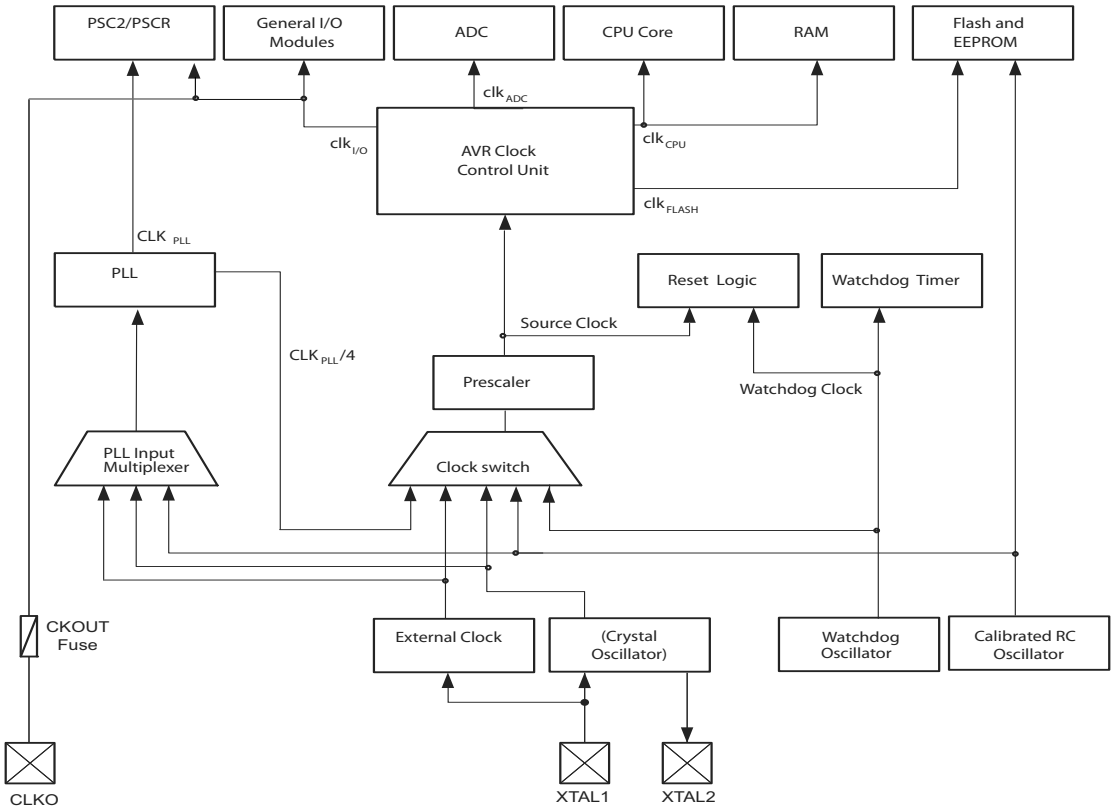
After reset, CKSEL fuses select one clock source. Once the device is running, software clock switching is available on any other clock sources.

Some hardware controls are provided for clock switching management but some specific procedures must be observed. Some settings may lead the user to program the device in an inadequate configuration.

5.1 Clock Systems and their Distribution

Figure 5-1 presents the principal clock systems in the AVR and their distribution. All of the clocks may not be active at a given time. In order to reduce power consumption, the clocks from modules not being used can be halted by using different sleep modes or by using features of the dynamic clock switch (“Power Management and Sleep Modes” on page 45 or “Dynamic Clock Switch” on page 35). The clock systems are detailed below.

Figure 5-1. Clock distribution.



5.1.1 clk_{CPU} - CPU Clock

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the

1. Clock stability before switching
   Once the new clock source is selected, the count procedure is running. The user (code) should wait for the setting of the CLKRDY flag in CLKSCR register before to perform a switching.
2. Clock available on request
   AT any time, the user (code) can ask for the availability of a clock source. The user (code) can request it writing the appropriate command in the CLKSCR register. A full status on clock sources then can be done.

5.3.6 Clock Switching

To drive the system clock, the user can switch from the current clock source to the following ones (one of them is the current clock source):

1. Calibrated internal RC oscillator 8.0MHz/1.0MHz,
2. Internal watchdog oscillator 128kHz,
3. External clock,
4. External Crystal/Ceramic Resonator,
5. PLL output divided by four.

The clock switching is performed in a sequence of commands. First, the user (code) must make sure that the new clock source is running. Then the switching command can be entered. At the end, the user (code) can stop the previous clock source. It will be better to run this sequence once the interrupts disabled. The user (code) has the responsibility of the clock switching sequence.

the clock frequency and, of course, if the clock is alive. The user's has itself to do the difference between '***no_clock_signal***' and '***clock_signal_not_yet_available***'.

- **Bits 3:0 – CLKC3:0: Clock Control Bits 3 - 0**

These bits define the command to provide to the '*Clock Switch'* module. The special write procedure must be followed to change the CLKC bits (see "Bit 7 – CLKCCE: Clock Control Change Enable" on page 42).

1. Write the Clock Control Change Enable (CLKCCE) bit to one and all other bits in CLKCSR to zero.
2. Within four cycles, write the desired value to CLKCSR register while clearing CLKCCE bit.

Interrupts should be disabled when setting CLKCSR register in order not to disturb the procedure.

Table 5-12. Clock command list.

Clock command	CLKC3..0
No command	0000_b
Disable clock source	0001_b
Enable clock source	0010_b
Request for clock availability	0011_b
Clock source switch	0100_b
Recover system clock source code	0101_b
CKOUT command	0111_b
No command	$1xxx_b$

5.5.6 CLKSELR - Clock Selection Register

Bit	7	6	5	4	3	2	1	0	
	-	COUT	CSUT1	CSUT0	CSEL3	CSEL2	CSEL1	CSEL0	CLKSELR
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	CKOUT fuse	SUT1..0 fuses			CKSEL3..0 fuses			

- **Bit 7– Res: Reserved Bit**

This bit is reserved bit in the AT90PWM81/161 and will always read as zero.

- **Bit 6 – COUT: Clock Out**

The COUT bit is initialized with CKOUT Fuse bit.
The COUT bit is only used in case of '*CKOUT*' command. Refer to Section 5.2.7 "Clock Output Buffer" on page 34 for using.
In case of '*Recover System Clock Source*' command, COUT it is not affected (no recovering of this setting).

- **Bits 5:4 – CSUT1:0: Clock Start-up Time**

CSUT bits are initialized with the values of SUT Fuse bits.
In case of '*Enable/Disable Clock Source*' command, CSUT field provides the code of the clock start-up time. Refer to subdivisions of Section 5.2 "Clock Sources" on page 28 for code of clock

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 "ADC Noise Canceler" on page 210 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 NOT automatically disabled, so it should be disabled if not used.

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 "Analog Comparator" on page 194 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 "Brown-out Detection" on page 53 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 "Internal Voltage Reference" on page 55 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 "Watchdog Timer" on page 56 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

7. System Control and Reset

7.1 System Control overview

7.1.1 Resetting the AVR

During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. The instruction placed at the Reset Vector must be a JMP – Absolute Jump – instruction to the reset handling routine. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa. The circuit diagram in Figure 7-1 on page 51 shows the reset logic. Table 7-1 on page 51 defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the SUT and CKSEL Fuses. The different selections for the delay period are presented in “Clock Sources” on page 28.

7.1.2 Reset Sources

The Atmel AT90PWM81/161 has four sources of reset:

- **Power-on Reset**. The MCU is reset when the supply voltage is below the Power-on Reset threshold (V_{POT}).
- **External Reset**. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length. The external reset pin can be disabled in 2 ways:
  - By the RSTDISBL fuse. In this case , the SPI programming is disabled.
  - By software using the RSTDIS bit in MCUCR register. In this case , the SPI programming is still active at power up time.
- **Watchdog Reset**. The MCU is reset when the Watchdog Timer period expires and the Watchdog is enabled.
- **Brown-out Reset**. The MCU is reset when the supply voltage V_{CC} is below the Brown-out Reset threshold (V_{BOT}) and the brown-out detector is enabled.

- **Bit 6 - WDIE: Watchdog Interrupt Enable**

When this bit is written to one and the I-bit in the Status Register is set, the Watchdog Interrupt is enabled. If WDE is cleared in combination with this setting, the Watchdog Timer is in Interrupt Mode, and the corresponding interrupt is executed if time-out in the Watchdog Timer occurs.

If WDE is set, the Watchdog Timer is in Interrupt and System Reset Mode. The first time-out in the Watchdog Timer will set WDIF. Executing the corresponding interrupt vector will clear WDIE and WDIF automatically by hardware (the Watchdog goes to System Reset Mode). This is useful for keeping the Watchdog Timer security while using the interrupt. To stay in Interrupt and System Reset Mode, WDIE must be set after each interrupt. This should however not be done within the interrupt service routine itself, as this might compromise the safety-function of the Watchdog System Reset mode. If the interrupt is not executed before the next time-out, a System Reset will be applied.

Table 7-5. Watchdog timer configuration.

WDTON (1)	WDE	WDIE	Mode	Action on time-out
0	0	0	Stopped	None
0	0	1	Interrupt mode	Interrupt
0	1	0	System reset mode	Reset
0	1	1	Interrupt and system reset mode	Interrupt, then go to system reset mode
1	x	x	System reset mode	Reset

Note: 1. For the WDTON fuse “1” means unprogrammed while “0” means programmed.

- **Bit 4 - WDCE: Watchdog Change Enable**

This bit is used in timed sequences for changing WDE and prescaler bits. To clear the WDE bit, and/or change the prescaler bits, WDCE must be set.

Once written to one, hardware will clear WDCE after four clock cycles.

- **Bit 3 - WDE: Watchdog System Reset Enable**

WDE is overridden by WDRF in MCUSR. This means that WDE is always set when WDRF is set. To clear WDE, WDRF must be cleared first. This feature ensures multiple resets during conditions causing failure, and a safe start-up after the failure.

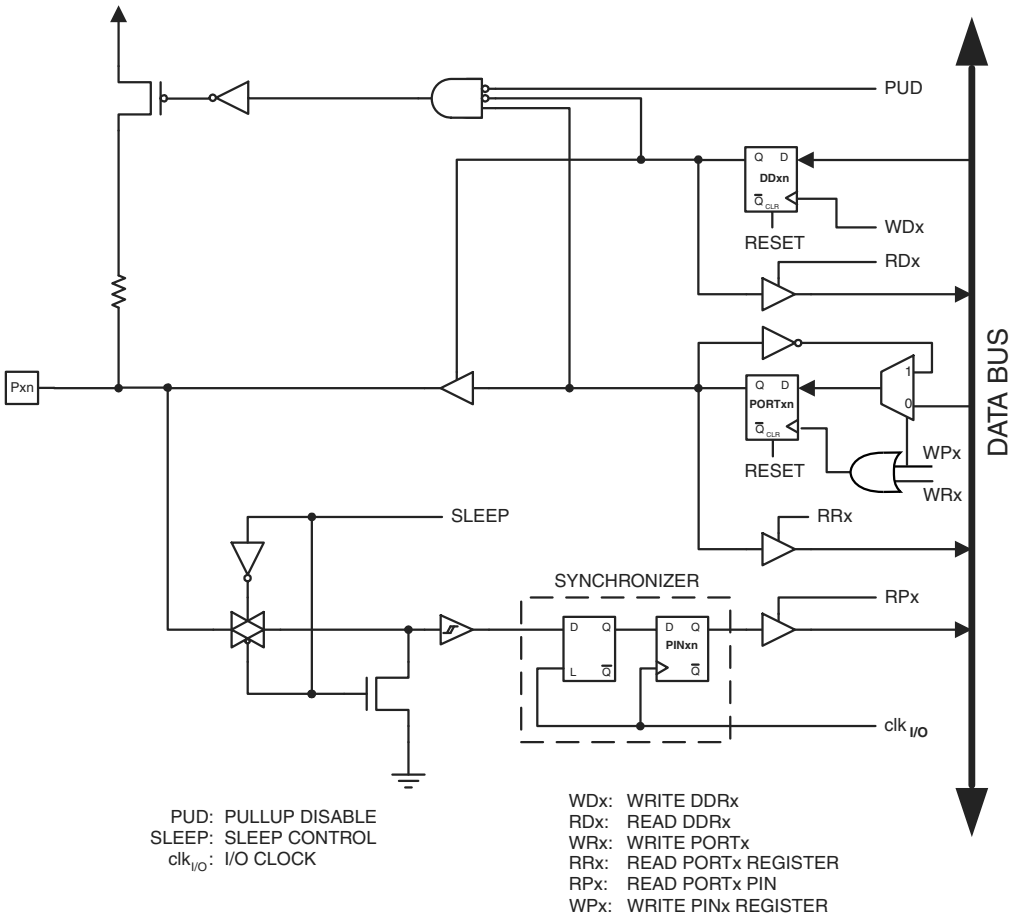
- **Bit 5, 2..0 - WDP3..0: Watchdog Timer Prescaler 3, 2, 1 and 0**

The WDP3..0 bits determine the Watchdog Timer prescaling when the Watchdog Timer is running. The different prescaling values and their corresponding time-out periods are shown in Table 7-6 on page 61.

9.2 Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 9-2 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 9-2. 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.

9.2.1 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 81, 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 9-8. Overriding signals for alternate functions in PD3..PD0.

Signal name	PD3/ADC1/ ACMP2_OUT	PD2/ADC0/ ACMP1	PD1/PSCOUTR0/ PSCINrB	PD0/ACMP3_OUT/$\overline{SS}$/ CLKO
PUOE	0	0	0	SPE.$\overline{MSTR}$
PUOV	0	0	0	PD0.$\overline{PUD}$
DDOE	ACE2EN	0	PSCen00	ACMP3D|(SPE.$\overline{MSTR}$)
DDOV	1	0	1	AC3EN
PVOE	AC2EN	0	PSCen00	AC3EN
PVOV	ACMP2_OUT	0	PSCOUT00	AVCMP3_OUT
DIEOE	ADC1D	ADC0D	0	0
DIEOV	0	0	0	0
DI	-	-	PSCINrB	$\overline{SS}$
AIO	ADC1	ADC0/ACMP1	-	-

9.3.4 Alternate Functions of Port E

The Port E pins with alternate functions are shown in Table 9-9.

Table 9-9. Port E pins alternate functions.

Port pin	Alternate function
PE3	AREF (Analog reference voltage) ADC6 (Analog input channel 6)
PE2	XTAL2: XTAL Output ACMP1M (Analog Comparator 1 Negative Input) PCSINr (PSCR Digital Input)
PE1	XTAL1: XTAL Input PCSIN2 (PSC 2 Digital Input) ACMP1_OUT (Analog Comparator 1 Output)
PE0	$\overline{RESET}$ (Reset Input) OCD (On Chip Debug I/O) INT2 (External Interrupt 2 Input)

The alternate pin configuration is as follows:

- **AREF/ADC6, Bit 3**

AREF: Analog reference voltage. See Table 17-3 on page 218 for the definition of this pin.

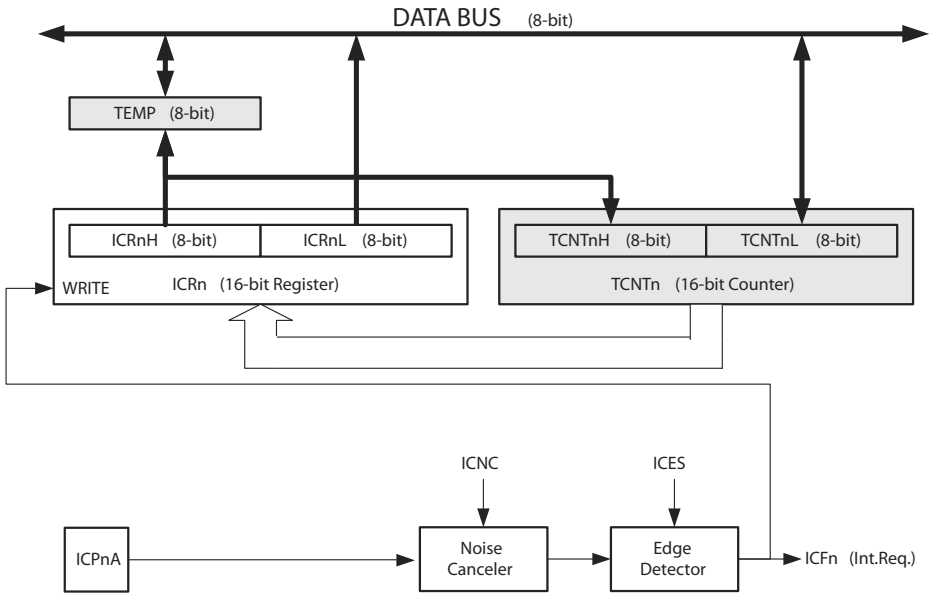
ADC6: Analog to Digital Converter, input channel 6.

This pin can only be used as a digital output pin. It cannot be read as a digital input.

- **XTAL2/ACMP1M/PSCINr – Bit 2**

XTAL2: Chip clock Oscillator pin 2. Used as clock pin for crystal Oscillator or Low-frequency crystal Oscillator. When used as a clock pin, the pin can not be used as an I/O pin.

Figure 11-4. Input capture unit block diagram.



When a change of the logic level (an event) occurs on the *Input Capture pin* (ICP1), alternatively on the *Analog Comparator output* (ACO), and this change confirms to the setting of the edge detector, a capture will be triggered. When a capture is triggered, the 16-bit value of the counter (TCNT1) is written to the *Input Capture Register* (ICR1). The *Input Capture Flag* (ICF1) is set at the same system clock as the TCNT1 value is copied into ICR1 Register. If enabled (ICIE1 = 1), the Input Capture Flag generates an Input Capture interrupt. The ICF1 Flag is automatically cleared when the interrupt is executed. Alternatively the ICF1 Flag can be cleared by software by writing a logical one to its I/O bit location.

Reading the 16-bit value in the *Input Capture Register* (ICR1) is done by first reading the low byte (ICR1L) and then the high byte (ICR1H). When the low byte is read the high byte is copied into the high byte temporary register (TEMP). When the CPU reads the ICR1H I/O location it will access the TEMP Register.

The ICR1 Register can only be written when using a Waveform Generation mode that utilizes the ICR1 Register for defining the counter's TOP value. In these cases the *Waveform Generation mode* (WGM13) bits must be set before the TOP value can be written to the ICR1 Register. When writing the ICR1 Register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

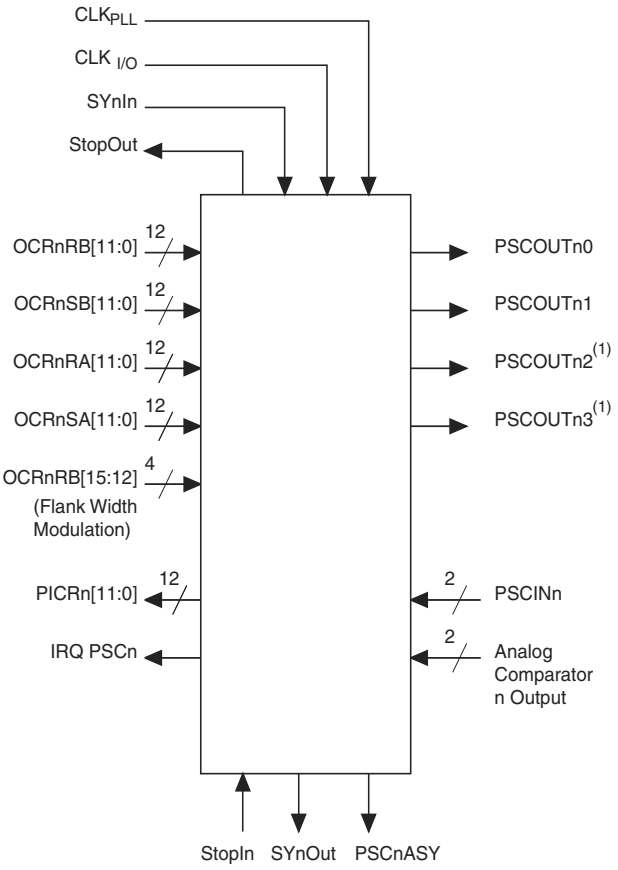
For more information on how to access the 16-bit registers refer to "Accessing 16-bit Registers" on page 87.

11.5.1 Input Capture Trigger Source

The main trigger source for the Input Capture unit is the *Input Capture pin* (ICP1). Timer/Counter1 can alternatively use the Analog Comparator output as trigger source for the Input Capture unit. The Analog Comparator is selected as trigger source by setting the *Analog Comparator Input Capture* (AC1ICE) bit in the *Analog Comparator Extended Control Register* (AC1ECON). Be aware that changing trigger source can trigger a capture. The Input Capture Flag must therefore be cleared after the change.

12.4 Signal Description

Figure 12-3. PSC external block view.



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

12.4.1 Input Description

Table 12-1. Internal inputs.

Name	Description	Type width
OCRnRB[11:0]	Compare value which reset signal on Part B (PSCOUTn1)	Register 12 bits
OCRnSB[11:0]	Compare value which set signal on Part B (PSCOUTn1)	Register 12 bits
OCRnRA[11:0]	Compare value which reset signal on Part A (PSCOUTn0)	Register 12 bits
OCRnSA[11:0]	Compare value which set signal on Part A (PSCOUTn0)	Register 12 bits
OCRnRB[15:12]	Frequency resolution enhancement value (flank width modulation)	Register 4 bits
CLK I/O	Clock input from I/O clock	Signal
CLK PLL	Clock input from PLL	Signal
SYnIn	Synchronization in (from adjacent PSC) (1)	Signal
StopIn	Stop input (for synchronized mode)	Signal

Note: 1. See Figure 12-41 on page 132

12.7 Enhanced Resolution

Lamp Ballast applications need an enhanced resolution down to 50Hz. The method to improve the normal resolution is based on Flank Width Modulation (also called Fractional Divider). Cycles are grouped into frames of 16 cycles. Cycles are modulated by a sequence given by the fractional divider number. The resulting output frequency is the average of the frequencies in the frame. The fractional divider (d) is given by OCRnRB[15:12].

The PSC output period is directly equal to the PSCOUTn0 On Time + Dead Time (OT0+DT0) and PSCOUTn1 On Time + Dead Time (OT1+DT1) values. These values are 12 bits numbers. The frequency adjustment can only be done in steps like the dedicated counters. The step width is defined as the frequency difference between two neighboring PSC frequencies.

It is possible to apply the Flank Width Modulation (FWM) on RB, RB+RA, SB, SB+SA. The selection is done bit the bits PBFMn0 and PBFMn.

According to the ramp mode and the enhanced resolution mode (defined by PBFMn1:0), the frequency difference Df can take three different values:

$$\Delta f = 0$$

$$\Delta f1 = |f1 - f2| = \left|\frac{f_{PSC}}{k} - \frac{f_{PSC}}{k+1}\right| = f_{PSC} \times \frac{1}{k(k+1)}$$

$$\Delta f2 = |f1 - f2| = \left|\frac{f_{PSC}}{k} - \frac{f_{PSC}}{k+2}\right| = f_{PSC} \times \frac{2}{k(k+2)}$$

with k is the number of CLK_{PSC} period in a PSC cycle and is given by the following formula:

$$k = \frac{f_{PSC}}{f_{OP}}$$

with f_{OP} is the output operating frequency.

Example, in normal mode, with maximum operating frequency 160kHz and f_{PLL} = 64Mhz, k equals 400. The resulting resolution is Delta F equals 64MHz / 400 / 401 = 400Hz.

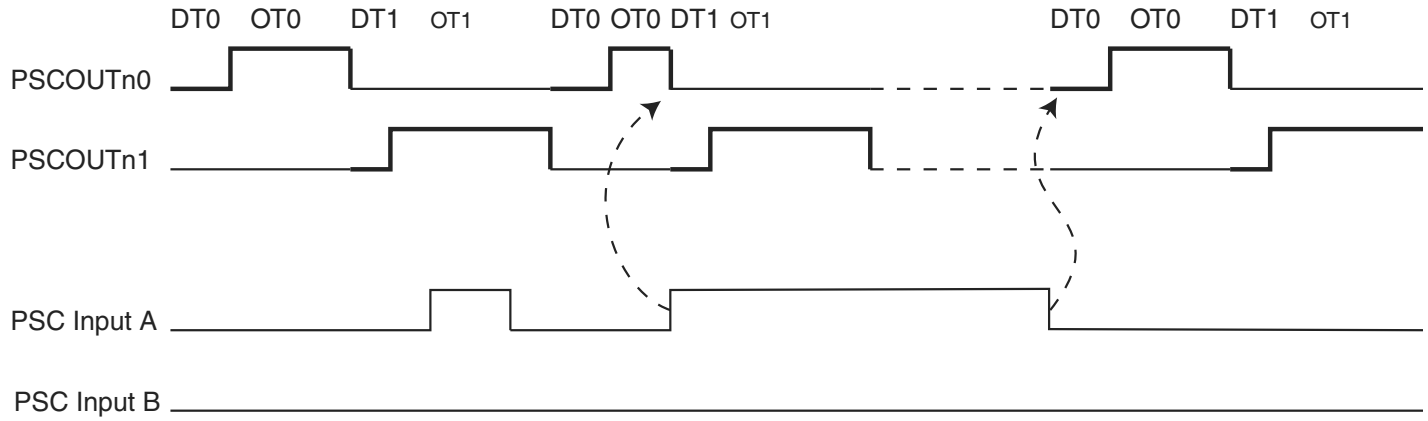
In enhanced mode, the output frequency is the average of the frame formed by the 16 consecutive cycles.

f_{b1} and f_{b2} are two neighboring base frequencies.

$$f_{AVERAGE} = \frac{16-d}{16} \times f_{b1} + \frac{d}{16} \times f_{b2}$$

12.10 PSC Input Mode 2: Stop signal, Execute Opposite Pulse and Wait

Figure 12-24. PSCn behavior versus PSCn Input A in Fault Mode 2.

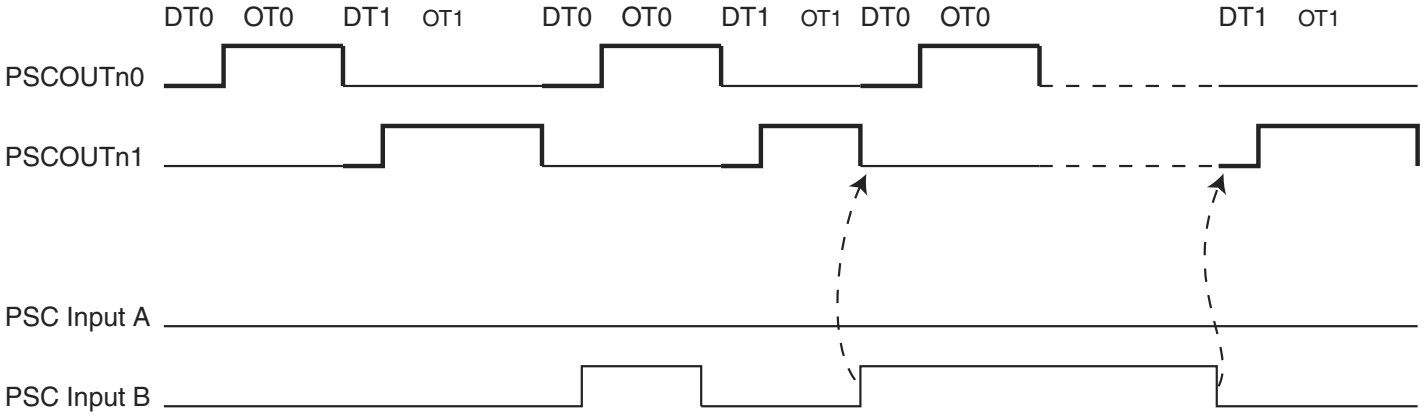


PSC Input A is take into account during DT0 and OT0 only. It has no effect during DT1 and OT1.

When PSCn Input A event occurs, PSC releases PSCOUTn0, jumps and executes DT1 plus OT1 and then waits for PSC Input A inactive state.

Even if PSC Input A is released during DT1 or OT1, DT1 plus OT1 sub-cycle is always completely executed.

Figure 12-25. PSCn behavior versus PSCn Input B in Fault Mode 2.



PSC Input B is take into account during DT1 and OT1 only. It has no effect during DT0 and OT0.

When PSC Input B event occurs, PSC releases PSCOUTn1, jumps and executes DT0 plus OT0 and then waits for PSC Input B inactive state.

Even if PSC Input B is released during DT0 or OT0, DT0 plus OT0 sub-cycle is always completely executed.

12.12 PSC Input Mode 4: Deactivate outputs without changing timing

Figure 12-28. PSC behavior versus PSCn Input A or Input B in Mode 4.

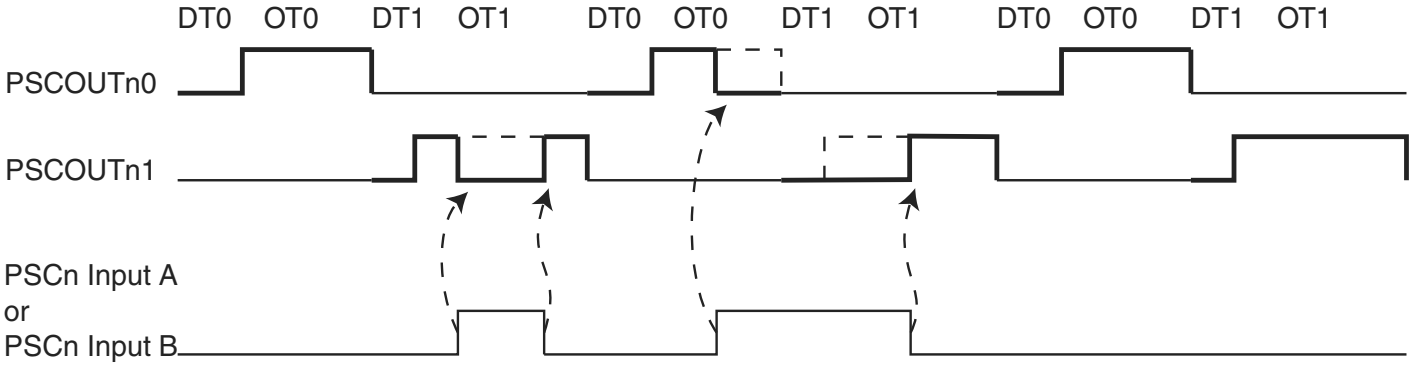
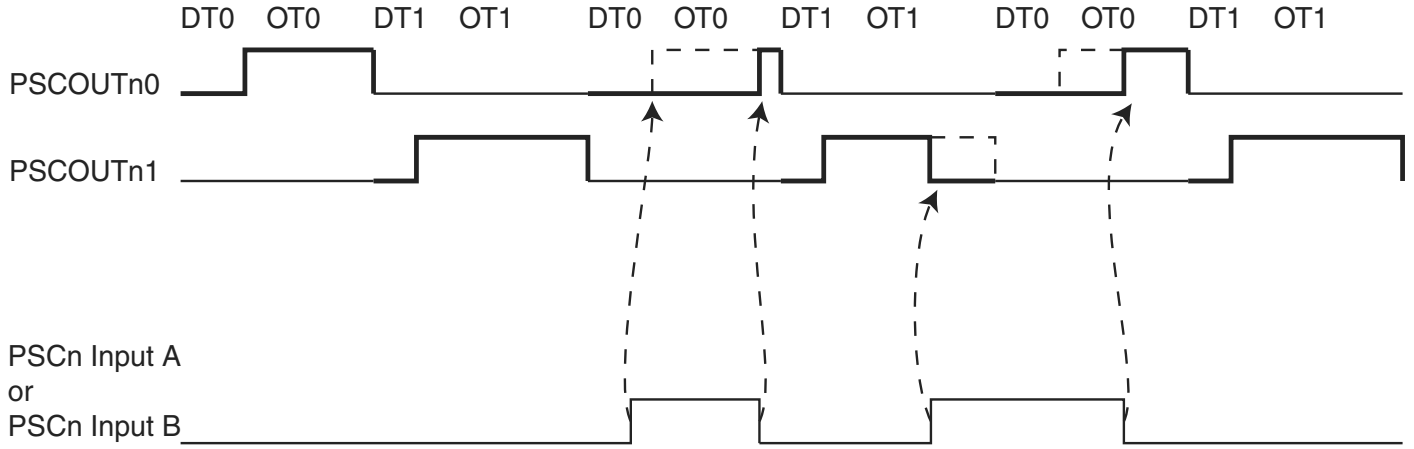


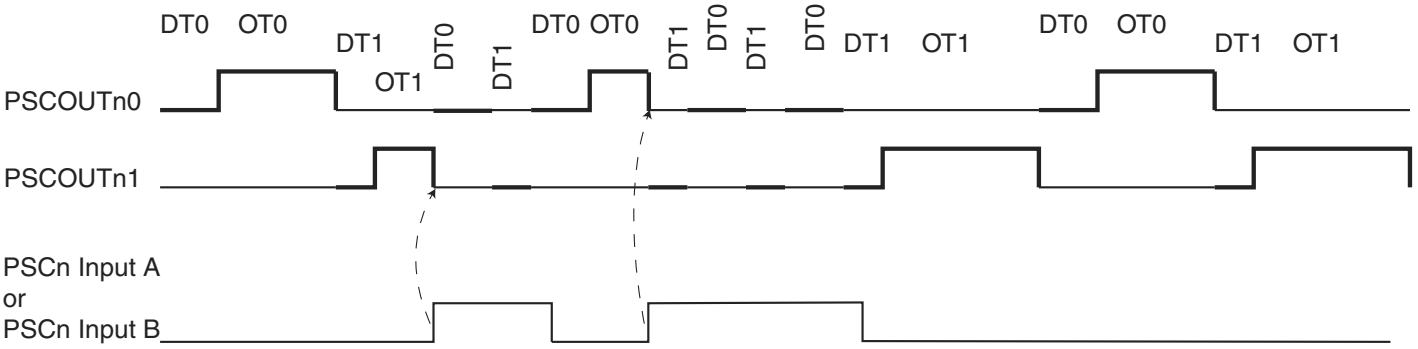
Figure 12-29. PSC behavior versus PSCn Input A or Input B in Fault Mode 4.



PSCn Input A or PSCn Input B act indifferently on On-Time0/Dead-Time0 or on On-Time1/Dead-Time1.

12.13 PSC Input Mode 5: Stop signal and Insert Dead-Time

Figure 12-30. PSC behavior versus PSCn Input A in Fault Mode 5.



Used in Fault mode 5, PSCn Input A or PSCn Input B act indifferently on On-Time0/Dead-Time0 or on On-Time1/Dead-Time1.

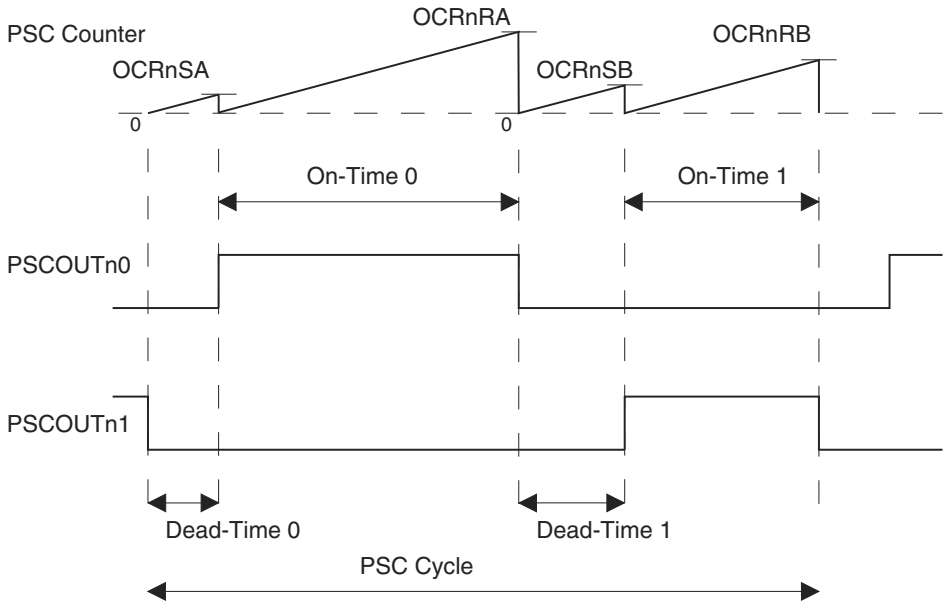
The waveform frequency is defined by the following equation:

$$f_{PSCn} = \frac{1}{PSCnCycle} = \frac{f_{CLK\_PSCn}}{(OT0 + OT1 + DT0 + DT1)}$$

13.5.2.1 Four Ramp Mode

In Four Ramp mode, each time in a cycle has its own definition.

Figure 13-4. PSCr0 & PSCr1 basic waveforms in Four Ramp mode.



The input clock of PSCR is given by CLKPSC.

PSCOUTr0 and PSCOUTr1 signals are defined by On-Time 0, Dead-Time 0, On-Time 1 and Dead-Time 1 values with:

On-Time 0 = OCRrRAH/L × 1/Fclkpsc

On-Time 1 = OCRrRBH/L × 1/Fclkpsc

Dead-Time 0 = (OCRrSAH/L + 2) × 1/Fclkpsc

Dead-Time 1 = (OCRrSBH/L + 2) × 1/Fclkpsc

Note: Minimal value for Dead-Time 0 and Dead-Time 1 = 2 × 1/Fclkpsc.

Table 13-15. PSCR ramp number description.

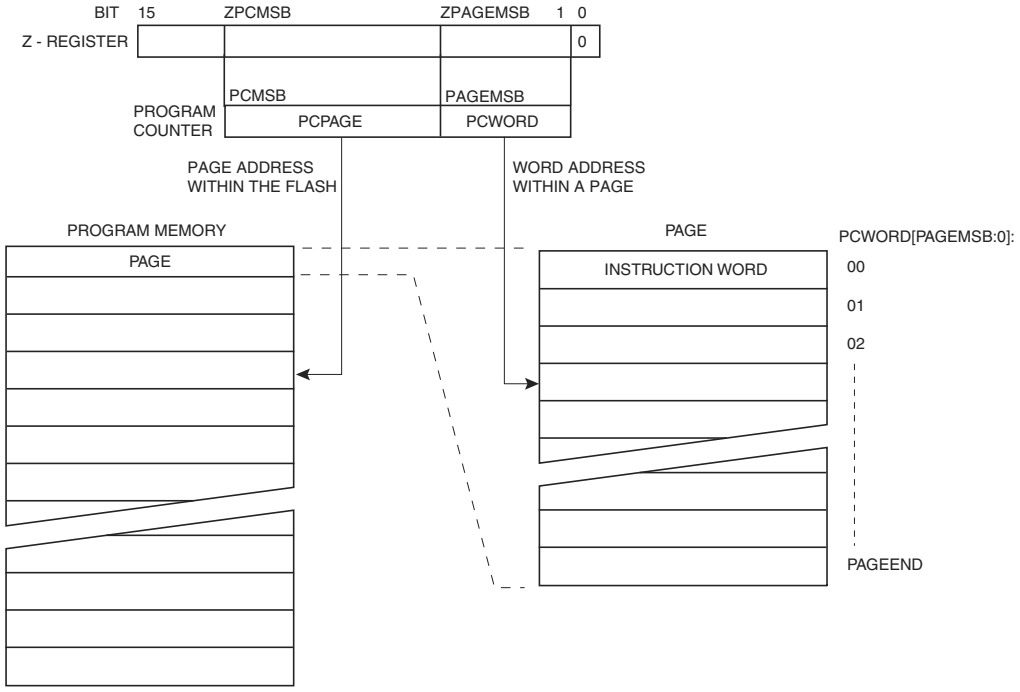
PRN01	PRN00	Description
0	0	The last event which has generated an interrupt occurred during ramp 1
0	1	The last event which has generated an interrupt occurred during ramp 2
1	0	The last event which has generated an interrupt occurred during ramp 3
1	1	The last event which has generated an interrupt occurred during ramp 4

- **Bit 0 – PEOP0: End Of PSCR Interrupt**

This bit is set by hardware when PSCR achieves its whole cycle.

Must be cleared by software by writing a one to its location.

Figure 20-3. Addressing the flash during SPM [1].



Note: 1. The different variables used in Figure 20-3 are listed in Table 20-10 on page 247.

20.7 Self-Programming the Flash

The program memory is updated in a page by page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM and the buffer can be filled either before the Page Erase command or between a Page Erase and a Page Write operation:

Alternative 1, fill the buffer before a Page Erase:

- Fill temporary page buffer
- Perform a Page Erase
- Perform a Page Write

Alternative 2, fill the buffer after Page Erase:

- Perform a Page Erase
- Fill temporary page buffer
- Perform a Page Write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be rewritten. When using Alternative 1, the Boot Loader provides an effective Read-Modify-Write feature which allows the user software to first read the page, do the necessary changes, and then write back the modified data. If Alternative 2 is used, it is not possible to read the old data while loading since the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the Page Erase and Page Write operation is addressing the same page. See “Simple Assembly Code Example for a Boot Loader” on page 245 for an assembly code example.

Table 21-3. Lock bit protection modes [1][2]. Only ATmega88/168.

BLB0 mode	BLB02	BLB01	
1	1	1	No restrictions for SPM or LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
BLB1 mode	**BLB12**	**BLB11**	
1	1	1	No restrictions for SPM or LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Notes: 1. Program the Fuse bits and Boot Lock bits before programming the LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed.

21.2 Fuse Bits

The AT90PWM81/161 has three Fuse bytes. Table 21-4 to Table 21-6 on page 251 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 21-4. Extended Low Fuse byte.

Extended fuse byte	Bit No	Description	Default value
PSC2RB	7	PSC2 reset behavior	1
PSC2RBA	6	PSC2 reset behavior for OUT22 & 23	1
PSCRRB	5	PSC reduced reset behavior	1
PSCRV	4	PSCOUT & PSCOUTR reset value	1
PSCINRB	3	PSC & PSCR inputs reset behavior	1
BODLEVEL2 [1]	2	Brown-out detector trigger level	1 (unprogrammed)
BODLEVEL1 [1]	1	Brown-out detector trigger level	0 (programmed)
BODLEVEL0 [1]	0	Brown-out detector trigger level	1 (unprogrammed)

Notes: 1. See Table 7-2 on page 53 for BODLEVEL Fuse decoding.

21.2.1 PSC Output Behavior During Reset

For external component safety reason, the state of PSC outputs during Reset can be programmed by fuses PSCRV, PSCRRB & PSC2RB.

These fuses are located in the Extended Fuse Byte (see Table 21-4).

1. Set XA1, XA0 to “01”. This enables data loading.
2. Set DATA = Data low byte (0x00 - 0xFF).
3. Give XTAL1 a positive pulse. This loads the data byte.

D. Load Data High Byte

1. Set BS1 to “1”. This selects high data byte.
2. Set XA1, XA0 to “01”. This enables data loading.
3. Set DATA = Data high byte (0x00 - 0xFF).
4. Give XTAL1 a positive pulse. This loads the data byte.

E. Latch Data

1. Set BS1 to “1”. This selects high data byte.
2. Give PAGEL a positive pulse. This latches the data bytes (see Figure 21-3 on page 257 for signal waveforms).

F. Repeat B through E until the entire buffer is filled or until all data within the page is loaded

While the lower bits in the address are mapped to words within the page, the higher bits address the pages within the FLASH. This is illustrated in Figure 21-2 on page 257. Note that if less than eight bits are required to address words in the page (pagesize <256), the most significant bit(s) in the address low byte are used to address the page when performing a Page Write.

G. Load Address High byte

1. Set XA1, XA0 to “00”. This enables address loading.
2. Set BS1 to “1”. This selects high address.
3. Set DATA = Address high byte (0x00 - 0xFF).
4. Give XTAL1 a positive pulse. This loads the address high byte.

H. Program Page

1. Give $\overline{WR}$ a negative pulse. This starts programming of the entire page of data. RDY/$\overline{BSY}$ goes low.
2. Wait until RDY/$\overline{BSY}$ goes high (see Figure 21-3 on page 257 for signal waveforms).

I. Repeat B through H until the entire Flash is programmed or until all data has been programmed

J. End Page Programming

1. 1. Set XA1, XA0 to “10”. This enables command loading.
2. Set DATA to “0000 0000”. This is the command for No Operation.
3. Give XTAL1 a positive pulse. This loads the command, and the internal write signals are reset.

26. Ordering Information

Speed (MHz)	Power supply	Ordering code	Package	Operation range
16	2.7V - 5.5V	AT90PWM81-16ME	QFN32 (1)	Extended (-40°C to 105°C)
		AT90PWM81-16SE	SO20	
		AT90PWM81-16MF	QFN32 (2)	Extended (-40°C to 125°C)
		AT90PWM81-16SF	SO20	

16	2.7V - 5.5V	AT90PWM161-16MN	QFN32 (3)	Extended (-40°C to 105°C)
		AT90PWM161-16SN	SO20	
		AT90PWM161-16MF	QFN32 (4)	Extended (-40°C to 125°C)
		AT90PWM161-16SF	SO20	

Note: All packages are Pb free, fully LHF

Note: This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

Note: Parts numbers are for shipping in sticks (SO) or in trays (QFN). Thes devices can also be supplied in Tape and Reel. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

Note:
1. Marking on the package is PWM81-MN.
2. Marking on the package is PWM81-MF.
3. Marking on the package is PWM161-MN.
4. Marking on the package is PWM161-MF.

Package type	
SO20	TG, 20-lead, 0.300” body width plastic gull wing small outline package (SOIC)
QFN32	PN, 32-lead, 5.0 x 5.0mm body, 0.50mm pitch quad flat no lead package (QFN)

27. Errata

27.1 Errata AT90PWM81 revA

- **Available on request**

27.2 Errata AT90PWM81 revB

- **Clock Switch disable**
- **Crystal oscillator control with Clock Switch**
- **BOD disable fuse**
- **PSC output at reset**
- **Flash and EEPROM programming failure if CPU clock is switched**
- **ADC AMPlifier measurement is unstable**
- **ADC measurement reports abnormal values with PSC2-synchronized conversions**
- **Over-consumption in power down sleep mode**

1. **Clock Switch enable & disable**

   After a “Enable Clock Source” or a “Disable Clock Source” command, the command is still active until the next access of CLKCSR register. If CLKSEL is written with a new value, the corresponding clock will be unintentionnaly enabled or disabled.

   Work around:

   After the Enable or Disable command, write CLKCSR with value 1<<CLKCCE

2. **Crystal oscillator control with Clock Switch**

   When a Xtal oscillator is active and CLKSELR is written with a new value for the selection of another clock source (for instance RC or WD) , the Xtal oscillator gain is not correct.

   Work around:

   After the commands “Enable Clock Source” and “Clock Source Switching”, write back CLKSELR with the values corresponding to the active Xtal oscillator

3. **BOD disable fuse**

   It is strongly advised to keep the BOD active. Indeed, the RC oscillator may lock if it is activated when the power suppy goes at a low voltage.

   Work around:

   If it is mandatory to disable the BOD, do not set the RC oscillator as clock source during reset and makes sure the RC oscillator is never active when the power supply is below the lowest POR voltage (2.6V).

4. **PSC output at Reset**

   At Reset, the PSC outputs may be set at a value different from the PSC Fuse configuration (Bit 4 of Extended Fuse Byte).

   Work around:

   Initiate PSC output states from source code.