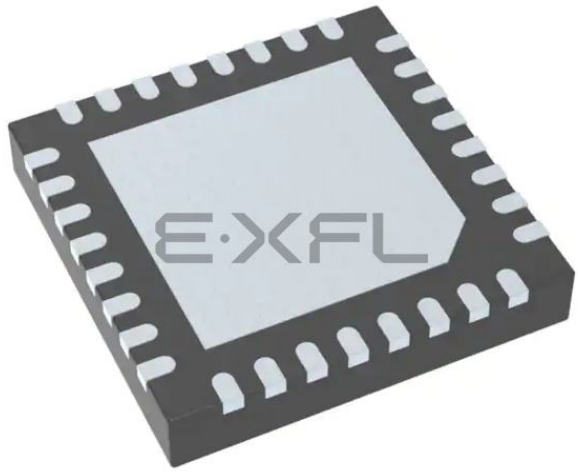


Welcome to E-XFL.COM

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"

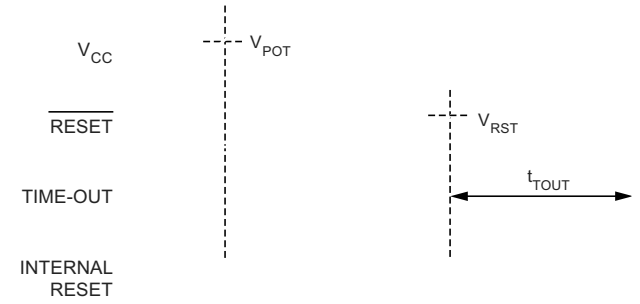
Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	23
Program Memory Size	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-QFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega48pa-15mz

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 selected 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 9-17. 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

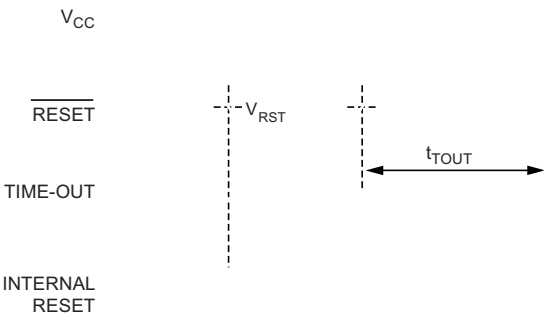
Figure 11-3. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally



11.4 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 Section 29.5 “System and Reset Characteristics” on page 272) 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. The external reset can be disabled by the RSTDISBL fuse, see Table 28-6 on page 253.

Figure 11-4. External Reset During Operation

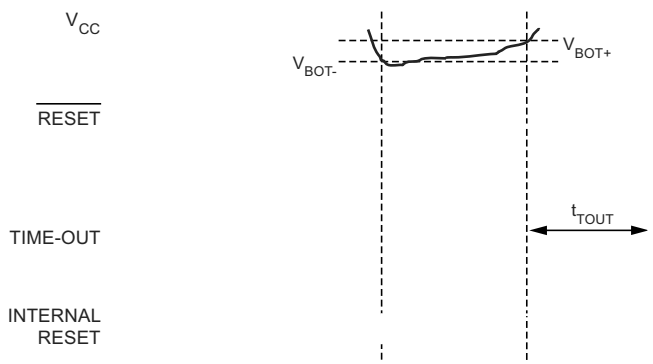


11.5 Brown-out Detection

The Atmel® ATmega48PA/88PA/168PA 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 $V_{BOT-} = V_{BOT} – V_{HYST}/2$.When the BOD is enabled, and V_{CC} decreases to a value below the trigger level (V_{BOT-} in Figure 11-5), the brown-out reset is immediately activated. When V_{CC} increases above the trigger level (V_{BOT+} in Figure 11-5), the delay counter starts the MCU after the time-out period t_{TOUT} has expired.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in Section 29.5 “System and Reset Characteristics” on page 272.

Figure 11-5. Brown-out Reset During Operation



12.2 Interrupt Vectors in the Atmel ATmega88PA

Table 12-2. Reset and Interrupt Vectors in the Atmel ATmega88PA

Vector No.	Program Address(2)	Source	Interrupt Definition
1	0x000(1)	RESET	External pin, power-on reset, brown-out reset and watchdog system reset
2	0x001	INT0	External interrupt request 0
3	0x002	INT1	External interrupt request 1
4	0x003	PCINT0	Pin change interrupt request 0
5	0x004	PCINT1	Pin change interrupt request 1
6	0x005	PCINT2	Pin change interrupt request 2
7	0x006	WDT	Watchdog time-out interrupt
8	0x007	TIMER2 COMPA	Timer/Counter2 compare match A
9	0x008	TIMER2 COMPB	Timer/Counter2 compare match B
10	0x009	TIMER2 OVF	Timer/Counter2 overflow
11	0x00A	TIMER1 CAPT	Timer/Counter1 capture event
12	0x00B	TIMER1 COMPA	Timer/Counter1 compare match A
13	0x00C	TIMER1 COMPB	Timer/coutner1 compare match B
14	0x00D	TIMER1 OVF	Timer/Counter1 overflow
15	0x00E	TIMER0 COMPA	Timer/Counter0 compare match A
16	0x00F	TIMER0 COMPB	Timer/Counter0 compare match B
17	0x010	TIMER0 OVF	Timer/Counter0 overflow
18	0x011	SPI, STC	SPI serial transfer complete
19	0x012	USART, RX	USART Rx complete
20	0x013	USART, UDRE	USART, data register empty
21	0x014	USART, TX	USART, Tx complete
22	0x015	ADC	ADC conversion complete
23	0x016	EE READY	EEPROM ready
24	0x017	ANALOG COMP	Analog comparator
25	0x018	TWI	2-wire serial interface
26	0x019	SPM READY	Store program memory ready

Notes:
1. When the BOOTRST fuse is programmed, the device will jump to the boot loader address at reset, see Section 27. “Boot Loader Support – Read-While-Write Self-Programming” on page 237.
2. When the IVSEL bit in MCUCR is set, interrupt vectors will be moved to the start of the boot flash section. The address of each interrupt vector will then be the address in this table added to the start address of the boot flash section.

When the BOOTRST fuse is unprogrammed, the boot section size set to 2Kbytes and the IVSEL bit in the MCUCR register is set before any interrupts are enabled, the most typical and general program setup for the reset and interrupt vector addresses in the Atmel® ATmega168PA is:

```
Address Labels      Code                Comments
0x0000 RESET:       ldi    r16,high(RAMEND); Main program start
0x0001              out    SPH,r16      ; Set Stack Pointer to top of RAM
0x0002              ldi    r16,low(RAMEND)
0x0003              out    SPL,r16
0x0004              sei                 ; Enable interrupts
0x0005              <instr> xxx
;
.org 0x1C02
0x1C02              jmp EXT_INT0        ; IRQ0 Handler
0x1C04              jmp EXT_INT1        ; IRQ1 Handler
...                 ...          ...    ;
0x1C32              jmp SPM_RDY         ; Store Program Memory Ready Handler
```

When the BOOTRST fuse is programmed and the boot section size set to 2Kbytes, the most typical and general program setup for the reset and interrupt vector addresses in the Atmel ATmega168PA is:

```
Address Labels      Code                Comments
.org 0x0002
0x0002              jmp    EXT_INT0     ; IRQ0 Handler
0x0004              jmp    EXT_INT1     ; IRQ1 Handler
...                 ...    ...          ;
0x0032              jmp    SPM_RDY      ; Store Program Memory Ready Handler
;
.org 0x1C00
0x1C00 RESET:       ldi    r16,high(RAMEND); Main program start
0x1C01              out    SPH,r16      ; Set Stack Pointer to top of RAM
0x1C02              ldi    r16,low(RAMEND)
0x1C03              out    SPL,r16
0x1C04              sei                 ; Enable interrupts
0x1C05              <instr> xxx
```

When the BOOTRST fuse is programmed, the boot section size set to 2Kbytes and the IVSEL bit in the MCUCR register is set before any interrupts are enabled, the most typical and general program setup for the reset and interrupt vector addresses in the Atmel ATmega168PA is:

```
Address Labels      Code                Comments
;
.org 0x1C00
0x1C00              jmp    RESET        ; Reset handler
0x1C02              jmp    EXT_INT0     ; IRQ0 Handler
0x1C04              jmp    EXT_INT1     ; IRQ1 Handler
...                 ...    ...          ;
0x1C32              jmp    SPM_RDY      ; Store Program Memory Ready Handler
;
0x1C33 RESET:       ldi    r16,high(RAMEND); Main program start
0x1C34              out    SPH,r16      ; Set Stack Pointer to top of RAM
0x1C35              ldi    r16,low(RAMEND)
0x1C36              out    SPL,r16
0x1C37              sei                 ; Enable interrupts
0x1C38              <instr> xxx
```

13.2 Register Description

13.2.1 EICRA – External Interrupt Control Register A

The external interrupt control register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:4 – Reserved**

These bits are unused bits in the Atmel® ATmega48PA/88PA/168PA, and will always read as zero.

- **Bit 3, 2 – ISC11, ISC10: Interrupt Sense Control 1 Bit 1 and Bit 0**

The external interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 13-1. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 13-1. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

- **Bit 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The external interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 13-2. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 13-2. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

14.2.5 Digital Input Enable and Sleep Modes

As shown in Figure 14-2 on page 65, the digital input signal can be clamped to ground at the input of the Schmitt Trigger. The signal denoted SLEEP in the figure, is set by the MCU sleep controller in power-down mode, power-save mode, and standby mode to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

SLEEP is overridden for port pins enabled as external interrupt pins. If the external interrupt request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in Section 14.3 “Alternate Port Functions” on page 69.

If a logic high level (“one”) is present on an asynchronous external interrupt pin configured as “Interrupt on rising edge, falling edge, or any logic change on pin” while the external interrupt is *not* enabled, the corresponding external interrupt flag will be set when resuming from the above mentioned Sleep mode, as the clamping in these sleep mode produces the requested logic change.

14.2.6 Unconnected Pins

If some pins are unused, it is recommended to ensure that these pins have a defined level. Even though most of the digital inputs are disabled in the deep sleep modes as described above, floating inputs should be avoided to reduce current consumption in all other modes where the digital inputs are enabled (reset, active mode and Idle mode).

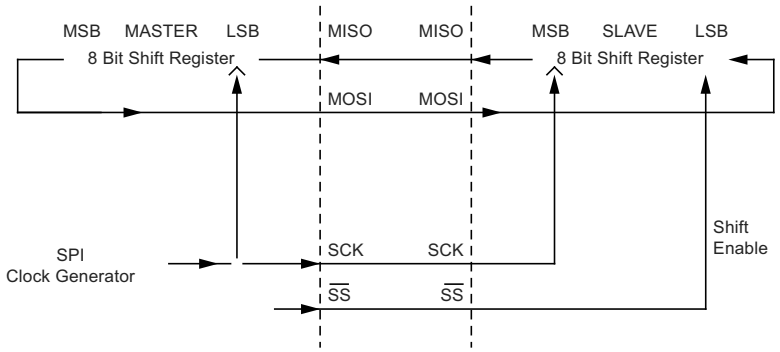
The simplest method to ensure a defined level of an unused pin, is to enable the internal pull-up. In this case, the pull-up will be disabled during reset. If low power consumption during reset is important, it is recommended to use an external pull-up or pull-down. Connecting unused pins directly to V_{CC} or GND is not recommended, since this may cause excessive currents if the pin is accidentally configured as an output.

The interconnection between master and slave CPUs with SPI is shown in Figure 19-2 on page 145. The system consists of two shift registers, and a master clock generator. The SPI master initiates the communication cycle when pulling low the slave select $\overline{SS}$ pin of the desired slave. master and slave prepare the data to be sent in their respective shift registers, and the master generates the required clock pulses on the SCK line to interchange data. Data is always shifted from master to slave on the master out – slave in, MOSI, line, and from slave to master on the master in – slave out, MISO, line. After each data packet, the master will synchronize the slave by pulling high the slave select, $\overline{SS}$, line.

When configured as a master, the SPI interface has no automatic control of the $\overline{SS}$ line. This must be handled by user software before communication can start. When this is done, writing a byte to the SPI data register starts the SPI clock generator, and the hardware shifts the eight bits into the slave. After shifting one byte, the SPI clock generator stops, setting the end of transmission Flag (SPIF). If the SPI interrupt enable bit (SPIE) in the SPCR register is set, an interrupt is requested. The master may continue to shift the next byte by writing it into SPDR, or signal the end of packet by pulling high the slave select, $\overline{SS}$ line. The last incoming byte will be kept in the buffer register for later use.

When configured as a slave, the SPI interface will remain sleeping with MISO tri-stated as long as the $\overline{SS}$ pin is driven high. In this state, software may update the contents of the SPI data register, SPDR, but the data will not be shifted out by incoming clock pulses on the SCK pin until the $\overline{SS}$ pin is driven low. As one byte has been completely shifted, the end of transmission flag, SPIF is set. If the SPI interrupt enable bit, SPIE, in the SPCR register is set, an interrupt is requested. The slave may continue to place new data to be sent into SPDR before reading the incoming data. The last incoming byte will be kept in the buffer register for later use.

Figure 19-2. SPI Master-slave Interconnection



The system is single buffered in the transmit direction and double buffered in the receive direction. This means that bytes to be transmitted cannot be written to the SPI Data Register before the entire shift cycle is completed. When receiving data, however, a received character must be read from the SPI Data Register before the next character has been completely shifted in. Otherwise, the first byte is lost.

In SPI Slave mode, the control logic will sample the incoming signal of the SCK pin. To ensure correct sampling of the clock signal, the minimum low and high periods should be:

Low periods: Longer than 2 CPU clock cycles.

High periods: Longer than 2 CPU clock cycles.

When the SPI is enabled, the data direction of the MOSI, MISO, SCK, and $\overline{SS}$ pins is overridden according to Table 19-1 on page 145. For more details on automatic port overrides, refer to Section 14.3 “Alternate Port Functions” on page 69.

Table 19-1. SPI Pin Overrides(1)

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

Note: 1. See Section 14.3.1 “Alternate Functions of Port B” on page 71 for a detailed description of how to define the direction of the user defined SPI pins.

22.2.2 Electrical Interconnection

As depicted in Figure 22-1, both bus lines are connected to the positive supply voltage through pull-up resistors. The bus drivers of all TWI-compliant devices are open-drain or open-collector. This implements a wired-AND function which is essential to the operation of the interface. A low level on a TWI bus line is generated when one or more TWI devices output a zero. A high level is output when all TWI devices tri-state their outputs, allowing the pull-up resistors to pull the line high. Note that all AVR devices connected to the TWI bus must be powered in order to allow any bus operation.

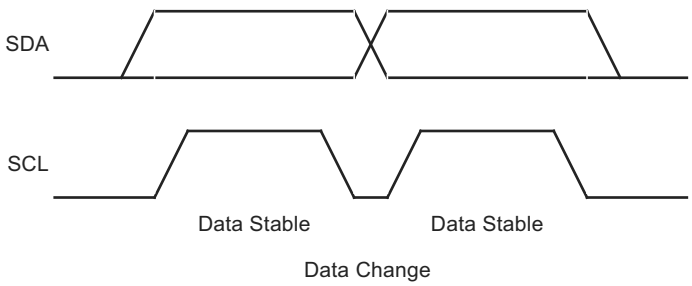
The number of devices that can be connected to the bus is only limited by the bus capacitance limit of 400pF and the 7-bit slave address space. A detailed specification of the electrical characteristics of the TWI is given in “Two-wire Serial Interface Characteristics” on page 275. Two different sets of specifications are presented there, one relevant for bus speeds below 100kHz, and one valid for bus speeds up to 400kHz.

22.3 Data Transfer and Frame Format

22.3.1 Transferring Bits

Each data bit transferred on the TWI bus is accompanied by a pulse on the clock line. The level of the data line must be stable when the clock line is high. The only exception to this rule is for generating start and stop conditions.

Figure 22-2. Data Validity



22.3.2 START and STOP Conditions

The Master initiates and terminates a data transmission. The transmission is initiated when the Master issues a START condition on the bus, and it is terminated when the master issues a STOP condition. Between a START and a STOP condition, the bus is considered busy, and no other master should try to seize control of the bus. A special case occurs when a new START condition is issued between a START and STOP condition. This is referred to as a REPEATED START condition, and is used when the Master wishes to initiate a new transfer without relinquishing control of the bus. After a REPEATED START, the bus is considered busy until the next STOP. This is identical to the START behavior, and therefore START is used to describe both START and REPEATED START for the remainder of this datasheet, unless otherwise noted. As depicted below, START and STOP conditions are signalled by changing the level of the SDA line when the SCL line is high.

Figure 22-3. START, REPEATED START and STOP conditions

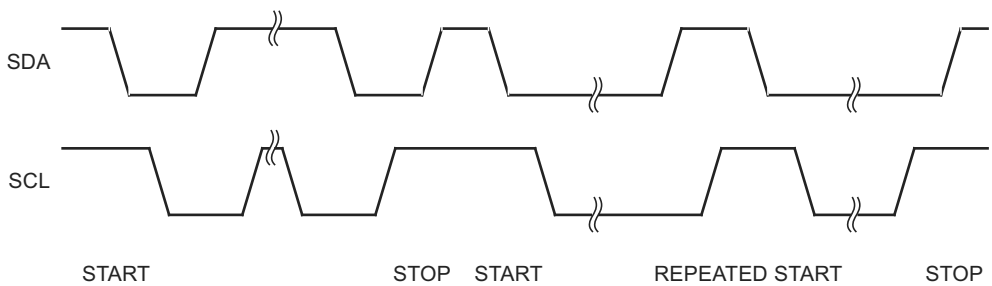


Table 22-3. Status codes for Master Transmitter Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0x08	A START condition has been transmitted	Load SLA+W	0	0	1	X	SLA+W will be transmitted; ACK or NOT ACK will be received
0x10	A repeated START condition has been transmitted	Load SLA+W or	0	0	1	X	SLA+W will be transmitted; ACK or NOT ACK will be received
		Load SLA+R	0	0	1	X	SLA+R will be transmitted; Logic will switch to Master Receiver mode
0x18	SLA+W has been transmitted; ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x20	SLA+W has been transmitted; NOT ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x28	Data byte has been transmitted; ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x30	Data byte has been transmitted; NOT ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x38	Arbitration lost in SLA+W or data bytes	No TWDR action or	0	0	1	X	2-wire Serial Bus will be released and not addressed Slave mode entered
		No TWDR action	1	0	1	X	A START condition will be transmitted when the bus becomes free

22.9.3 TWSR – TWI Status Register

Bit	7	6	5	4	3	2	1	0	
(0xB9)	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	TWSR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	1	1	1	1	1	0	0	0	

- **Bits 7:3 – TWS: TWI Status**

These 5 bits reflect the status of the TWI logic and the 2-wire serial bus. The different status codes are described later in this section. Note that the value read from TWSR contains both the 5-bit status value and the 2-bit prescaler value. The application designer should mask the prescaler bits to zero when checking the Status bits. This makes status checking independent of prescaler setting. This approach is used in this datasheet, unless otherwise noted.

- **Bit 2 – Reserved**

This bit is reserved and will always read as zero.

- **Bits 1:0 – TWPS: TWI Prescaler Bits**

These bits can be read and written, and control the bit rate prescaler.

Table 22-8. TWI Bit Rate Prescaler

TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

To calculate bit rates, see Section 22.5.2 “Bit Rate Generator Unit” on page 187. The value of TWPS1...0 is used in the equation.

22.9.4 TWDR – TWI Data Register

Bit	7	6	5	4	3	2	1	0	
(0xBB)	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0	TWDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	1	1	1	1	1	1	1	

In transmit mode, TWDR contains the next byte to be transmitted. In receive mode, the TWDR contains the last byte received. It is writable while the TWI is not in the process of shifting a byte. This occurs when the TWI Interrupt Flag (TWINT) is set by hardware. Note that the data register cannot be initialized by the user before the first interrupt occurs. The data in TWDR remains stable as long as TWINT is set. While data is shifted out, data on the bus is simultaneously shifted in. TWDR always contains the last byte present on the bus, except after a wake up from a sleep mode by the TWI interrupt. In this case, the contents of TWDR is undefined.

In the case of a lost bus arbitration, no data is lost in the transition from master to slave. Handling of the ACK bit is controlled automatically by the TWI logic, the CPU cannot access the ACK bit directly.

- **Bits 7:0 – TWD: TWI Data Register**

These eight bits constitute the next data byte to be transmitted, or the latest data byte received on the 2-wire serial bus.

24.6 ADC Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode to reduce noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC noise reduction and Idle mode. To make use of this feature, the following procedure should be used:

a. Make sure that the ADC is enabled and is not busy converting. Single conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
b. Enter ADC noise reduction mode (or idle mode). The ADC will start a conversion once the CPU has been halted.
c. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC conversion complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC conversion complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not be automatically turned off when entering other sleep modes than Idle mode and ADC noise reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption.

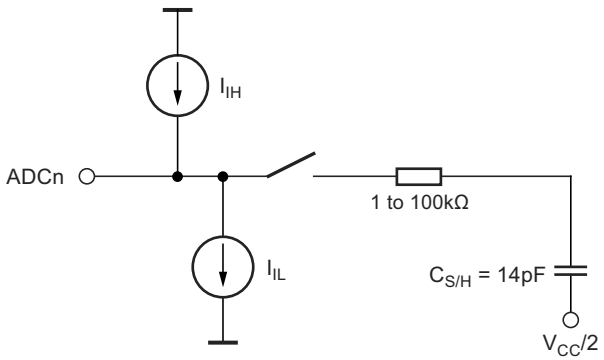
24.6.1 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 24-8 An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long time the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedance sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

Signal components higher than the Nyquist frequency ($f_{ADC}/2$) should not be present for either kind of channels, to avoid distortion from unpredictable signal convolution. The user is advised to remove high frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 24-8. Analog Input Circuitry



24.9 Register Description

24.9.1 ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – REFS[1:0]: Reference Selection Bits**

These bits select the voltage reference for the ADC, as shown in Table 24-3. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set). The internal voltage reference options may not be used if an external reference voltage is being applied to the AREF pin.

Table 24-3. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal V_{ref} turned off
0	1	AV_{CC} with external capacitor(1) at AREF pin
1	0	Reserved
1	1	Internal 1.1V voltage reference with external capacitor(1) at AREF pin

Note: 1. Note the value used for the external ARef capacitor (e.g. 10nF) should be very much smaller than the decoupling capacitor used on the AVcc pin (e.g. 100nF) to prevent possible switching glitches.

- **Bit 5 – ADLAR: ADC Left Adjust Result**

The ADLAR bit affects the presentation of the ADC conversion result in the ADC data register. Write one to ADLAR to left adjust the result. Otherwise, the result is right adjusted. Changing the ADLAR bit will affect the ADC data register immediately, regardless of any ongoing conversions. For a complete description of this bit, see Section 24.9.3 “ADCL and ADCH – The ADC Data Register” on page 227.

- **Bit 4 – Reserved**

This bit is an unused bit in the Atmel® ATmega48PA/88PA/168PA, and will always read as zero.

- **Bits 3:0 – MUX[3:0]: Analog Channel Selection Bits**

The value of these bits selects which analog inputs are connected to the ADC. See Table 24-4 for details. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set).

```
        ;       re-enable the RWW section
        ldi     spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
        rcall   Do_spm

        ;       read back and check, optional
        ldi     looplo, low(PAGESIZEB)      ;init loop variable
        ldi     loophi, high(PAGESIZEB)     ;not required for PAGESIZEB<=256
        subi    YL, low(PAGESIZEB)          ;restore pointer
        sbci    YH, high(PAGESIZEB)
Rdloop:
        lpm     r0, Z+
        ld      r1, Y+
        cpse    r0, r1
        rjmp    Error
        sbiw    loophi:looplo, 1            ;use subi for PAGESIZEB<=256
        brne    Rdloop

        ;       return to RWW section
        ;       verify that RWW section is safe to read
Return:
        in      temp1, SPMCSR
        sbrs    temp1, RWWSB                ; If RWWSB is set, the RWW section is not
ready yet
        ret
        ;       re-enable the RWW section
        ldi     spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
        rcall   Do_spm
        rjmp    Return

Do_spm:
        ;       check for previous SPM complete
Wait_spm:
        in      temp1, SPMCSR
        sbrc    temp1, SELFPRGEN
        rjmp    Wait_spm
        ;       input: spmcrval determines SPM action
        ;       disable interrupts if enabled, store status
        in      temp2, SREG
        cli
        ;       check that no EEPROM write access is present
Wait_ee:
        sbic    EECR, EEPE
        rjmp    Wait_ee
        ;       SPM timed sequence
        out     SPMCSR, spmcrval
        spm
        ;       restore SREG (to enable interrupts if originally enabled)
        out     SREG, temp2
        ret
```

26.3 Register Description

26.3.1 SPMCSR – Store Program Memory Control and Status Register

The store program memory control and status register contains the control bits needed to control the program memory operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	**SPMIE**	**RWWSB**	SIGRD	RWWSRE	**BLBSET**	**PGWRT**	**PGERS**	**SELFPRGEN**	**SPMCSR**
Read/Write	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPMIE: SPM Interrupt Enable**

When the SPMIE bit is written to one, and the I-bit in the status register is set (one), the SPM ready interrupt will be enabled. The SPM ready Interrupt will be executed as long as the SELFPRGEN bit in the SPMCSR register is cleared. The interrupt will not be generated during EEPROM write or SPM.

- **Bit 6 – RWWSB: Read-While-Write Section Busy**

This bit is for compatibility with devices supporting read-while-write. It will always read as zero in Atmel® ATmega48PA.

- **Bit 5 – SIGRD: Signature Row Read**

If this bit is written to one at the same time as SELFPRGEN, the next LPM instruction within three clock cycles will read a byte from the signature row into the destination register. see Section 27.8.10 “Reading the Signature Row from Software” on page 244 for details. An SPM instruction within four cycles after SIGRD and SELFPRGEN are set will have no effect. This operation is reserved for future use and should not be used.

- **Bit 4 – RWWSRE: Read-While-Write Section Read Enable**

The functionality of this bit in Atmel ATmega48PA is a subset of the functionality in the Atmel ATmega48PA/88PA/168PA. If the RWWSRE bit is written while filling the temporary page buffer, the temporary page buffer will be cleared and the data will be lost.

- **Bit 3 – BLBSET: Boot Lock Bit Set**

The functionality of this bit in Atmel ATmega48PA is a subset of the functionality in the Atmel ATmega48PA/88PA/168PA. An LPM instruction within three cycles after BLBSET and SELFPRGEN are set in the SPMCSR register, will read either the lock bits or the fuse bits (depending on Z0 in the Z-pointer) into the destination register. See Section 26.2.2 “Reading the Fuse and Lock Bits from Software” on page 233 for details.

- **Bit 2 – PGWRT: Page Write**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes page write, with the data stored in the temporary buffer. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGWRT bit will auto-clear upon completion of a page write, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire page write operation.

- **Bit 1 – PGERS: Page Erase**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes page erase. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGERS bit will auto-clear upon completion of a page erase, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire page write operation.

- **Bit 0 – SELFPRGEN: Self Programming Enable**

This bit enables the SPM instruction for the next four clock cycles. If written to one together with either RWWSRE, BLBSET, PGWRT, or PGERS, the following SPM instruction will have a special meaning, see description above. If only SELFPRGEN is written, the following SPM instruction will store the value in R1:R0 in the temporary page buffer addressed by the Z-pointer. The LSB of the Z-pointer is ignored. The SELFPRGEN bit will auto-clear upon completion of an SPM instruction, or if no SPM instruction is executed within four clock cycles. During page erase and page write, the SELFPRGEN bit remains high until the operation is completed. Writing any other combination than “10001”, “01001”, “00101”, “00011” or “00001” in the lower five bits will have no effect.

27.8.5 Consideration While Updating BLS

Special care must be taken if the user allows the Boot Loader section to be updated by leaving boot lock bit11 unprogrammed. An accidental write to the boot loader itself can corrupt the entire boot loader, and further software updates might be impossible. If it is not necessary to change the boot loader software itself, it is recommended to program the boot lock bit11 to protect the boot loader software from any internal software changes.

27.8.6 Prevent Reading the RWW Section during Self-programming

During self-programming (either page erase or page write), the RWW section is always blocked for reading. The user software itself must prevent that this section is addressed during the self programming operation. The RWWSB in the SPMCSR will be set as long as the RWW section is busy. During self-programming the interrupt vector table should be moved to the BLS as described in Section 11.8 “Watchdog Timer” on page 45, or the interrupts must be disabled. Before addressing the RWW section after the programming is completed, the user software must clear the RWWSB by writing the RWWSRE. See Section 27.8.13 “Simple Assembly Code Example for a Boot Loader” on page 245 for an example.

27.8.7 Setting the Boot Loader Lock Bits by SPM

To set the Boot Loader Lock bits and general Lock Bits, write the desired data to R0, write “X0001001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR.

Bit	7	6	5	4	3	2	1	0
R0	**1**	**1**	**BLB12**	**BLB11**	**BLB02**	**BLB01**	**LB2**	**LB1**

See Table 27-2 and Table 27-3 for how the different settings of the boot loader bits affect the flash access.

If bits 5...0 in R0 are cleared (zero), the corresponding Lock bit will be programmed if an SPM instruction is executed within four cycles after BLBSET and SELFPRGEN are set in SPMCSR. The Z-pointer is don’t care during this operation, but for future compatibility it is recommended to load the Z-pointer with 0x0001 (same as used for reading the IO_{ck} bits). For future compatibility it is also recommended to set bits 7 and 6 in R0 to “1” when writing the Lock bits. When programming the lock bits the entire flash can be read during the operation.

27.8.8 EEPROM Write Prevents Writing to SPMCSR

Note that an EEPROM write operation will block all software programming to flash. Reading the fuses and lock bits from software will also be prevented during the EEPROM write operation. It is recommended that the user checks the status bit (EEPE) in the EECR register and verifies that the bit is cleared before writing to the SPMCSR register.

27.8.9 Reading the Fuse and Lock Bits from Software

It is possible to read both the fuse and lock bits from software. To read the lock bits, load the Z-pointer with 0x0001 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM instruction is executed within three CPU cycles after the BLBSET and SELFPRGEN bits are set in SPMCSR, the value of the lock bits will be loaded in the destination register. The BLBSET and SELFPRGEN bits will auto-clear upon completion of reading the lock bits or if no LPM instruction is executed within three CPU cycles or no SPM instruction is executed within four CPU cycles. When BLBSET and SELFPRGEN are cleared, LPM will work as described in the instruction set manual.

Bit	7	6	5	4	3	2	1	0
Rd	**–**	**–**	**BLB12**	**BLB11**	**BLB02**	**BLB01**	**LB2**	**LB1**

The algorithm for reading the fuse low byte is similar to the one described above for reading the lock bits. To read the fuse low byte, load the Z-pointer with 0x0000 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the fuse low byte (FLB) will be loaded in the destination register as shown below. Refer to Table 28-5 on page 253 for a detailed description and mapping of the fuse low byte.

Bit	7	6	5	4	3	2	1	0
Rd	**FLB7**	**FLB6**	**FLB5**	**FLB4**	**FLB3**	**FLB2**	**FLB1**	**FLB0**

27.9 Register Description

27.9.1 SPMCSR – Store Program Memory Control and Status Register

The store program memory control and status register contains the control bits needed to control the boot loader operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	**SPMIE**	**RWWSB**	–	RWWSRE	**BLBSET**	**PGWRT**	**PGERS**	**SELFPRGEN**	**SPMCSR**
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPMIE: SPM Interrupt Enable**

When the SPMIE bit is written to one, and the I-bit in the status register is set (one), the SPM ready interrupt will be enabled. The SPM ready Interrupt will be executed as long as the SELFPRGEN bit in the SPMCSR register is cleared.

- **Bit 6 – RWWSB: Read-While-Write Section Busy**

When a self-programming (page erase or page write) operation to the RWW section is initiated, the RWWSB will be set (one) by hardware. When the RWWSB bit is set, the RWW section cannot be accessed. The RWWSB bit will be cleared if the RWWSRE bit is written to one after a self-programming operation is completed. Alternatively the RWWSB bit will automatically be cleared if a page load operation is initiated.

- **Bit 5 – Reserved**

This bit is a reserved bit in the Atmel® ATmega48PA/88PA/168PA and always read as zero.

- **Bit 4 – RWWSRE: Read-While-Write Section Read Enable**

When programming (page erase or page write) to the RWW section, the RWW section is blocked for reading (the RWWSB will be set by hardware). To re-enable the RWW section, the user software must wait until the programming is completed (SELFPRGEN will be cleared). Then, if the RWWSRE bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles re-enables the RWW section. The RWW section cannot be re-enabled while the flash is busy with a page erase or a page write (SELFPRGEN is set). If the RWWSRE bit is written while the flash is being loaded, the flash load operation will abort and the data loaded will be lost.

- **Bit 3 – BLBSET: Boot Lock Bit Set**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles sets boot lock bits and memory lock bits, according to the data in R0. The data in R1 and the address in the Z-pointer are ignored. The BLBSET bit will automatically be cleared upon completion of the Lock bit set, or if no SPM instruction is executed within four clock cycles.

An LPM instruction within three cycles after BLBSET and SELFPRGEN are set in the SPMCSR register, will read either the lock bits or the fuse bits (depending on Z0 in the Z-pointer) into the destination register. See Section 27.8.9 “Reading the Fuse and Lock Bits from Software” on page 243 for details.

- **Bit 2 – PGWRT: Page Write**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes page write, with the data stored in the temporary buffer. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGWRT bit will auto-clear upon completion of a page write, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire Page Write operation if the NRWW section is addressed.

- **Bit 1 – PGERS: Page Erase**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes page erase. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGERS bit will auto-clear upon completion of a page erase, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire page write operation if the NRWW section is addressed.

28.5 Page Size

Table 28-9. No. of Words in a Page and No. of Pages in the Flash

Device	Flash Size	Page Size	PCWORD	No. of Pages	PCPAGE	PCMSB
Atmel ATmega48PA/ 88PA/168PA	2K words (4K bytes)	32 words	PC[4:0]	64	PC[10:5]	10
Atmel ATmega88PA	4K words (8K bytes)	32 words	PC[4:0]	128	PC[11:5]	11
Atmel ATmega168PA	8K words (16K bytes)	64 words	PC[5:0]	128	PC[12:6]	12

Table 28-10. No. of Words in a Page and No. of Pages in the EEPROM

Device	EEPROM Size	Page Size	PCWORD	No. of Pages	PCPAGE	EEAMSB
Atmel ATmega48PA/ 88PA/168PA	256 bytes	4 bytes	EEA[1:0]	64	EEA[7:2]	7
Atmel ATmega88PA	512 bytes	4 bytes	EEA[1:0]	128	EEA[8:2]	8
Atmel ATmega168PA	512 bytes	4 bytes	EEA[1:0]	128	EEA[8:2]	8

28.6 Parallel Programming Parameters, Pin Mapping, and Commands

This section describes how to parallel program and verify flash program memory, EEPROM data memory, memory lock bits, and fuse bits in the Atmel® ATmega48PA/88PA/168PA. Pulses are assumed to be at least 250 ns unless otherwise noted.

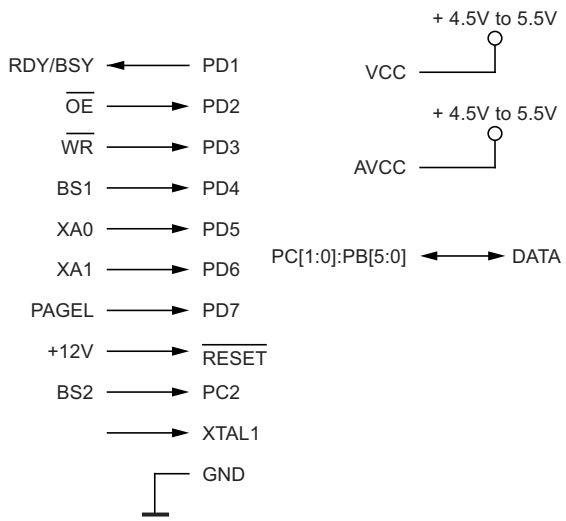
28.6.1 Signal Names

In this section, some pins of the Atmel ATmega48PA/88PA/168PA are referenced by signal names describing their functionality during parallel programming, see Figure 28-1 and Table 28-11. Pins not described in the following table are referenced by pin names.

The XA1/XA0 pins determine the action executed when the XTAL1 pin is given a positive pulse. The bit coding is shown in Table 28-13.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The different commands are shown in Table 28-14.

Figure 28-1. Parallel Programming



Note: $V_{CC} - 0.3V < AV_{CC} < V_{CC} + 0.3V$, however, AV_{CC} should always be within 4.5 to 5.5V

28.8.3 Serial Programming Instruction set

Table 28-17 on page 266 and Figure 28-8 on page 267 describes the Instruction set.

Table 28-17. Serial Programming Instruction Set (Hexadecimal values)

Instruction/Operation	Instruction Format			
	Byte 1	Byte 2	Byte 3	Byte4
Programming enable	$AC	$53	$00	$00
Chip erase (program memory/EEPROM)	$AC	$80	$00	$00
Poll RDY/$\overline{\text{BSY}}$	$F0	$00	$00	data byte out
Load Instructions				
Load extended address byte(1)	$4D	$00	Extended adr	$00
Load program memory page, high byte	$48	$00	adr LSB	high data byte in
Load program memory page, low byte	$40	$00	adr LSB	low data byte in
Load EEPROM memory page (page access)	$C1	$00	0000 000aa	data byte in
Read Instructions				
Read program memory, high byte	$28	adr MSB	adr LSB	high data byte out
Read program memory, low byte	$20	adr MSB	adr LSB	low data byte out
Read EEPROM memory	$A0	0000 00aa	aaaa aaaa	data byte out
Read lock bits	$58	$00	$00	data byte out
Read signature byte	$30	$00	0000 000aa	data byte out
Read fuse bits	$50	$00	$00	data byte out
Read fuse high bits	$58	$08	$00	data byte out
Read extended fuse bits	$50	$08	$00	data byte out
Read calibration byte	$38	$00	$00	data byte out
Write Instructions(6)				
Write program memory page	$4C	adr MSB(8)	adr LSB(8)	$00
Write EEPROM memory	$C0	0000 00aa	aaaa aaaa	data byte in
Write EEPROM memory page (page access)	$C2	0000 00aa	aaaa aa00	$00
Write lock bits	$AC	$E0	$00	data byte in
Write fuse bits	$AC	$A0	$00	data byte in
Write fuse high bits	$AC	$A8	$00	data byte in
Write extended fuse bits	$AC	$A4	$00	data byte in

Notes:
1. Not all instructions are applicable for all parts.
2. a = address.
3. Bits are programmed ‘0’, unprogrammed ‘1’.
4. To ensure future compatibility, unused fuses and lock bits should be unprogrammed (‘1’).
5. Refer to the corresponding section for fuse and lock bits, calibration and signature bytes and page size.
6. Instructions accessing program memory use a word address. This address may be random within the page range.
7. See http://www.atmel.com/avr for Application Notes regarding programming and programmers.
8. WORDS

30.2.8 BOD Threshold

Figure 30-43. BOD Thresholds versus Temperature (BODLEVEL is 1.8V)

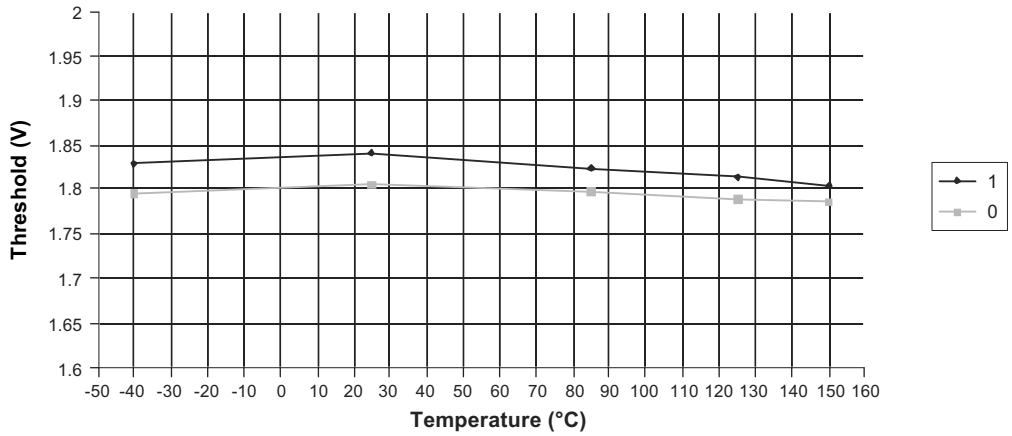


Figure 30-44. BOD Thresholds versus Temperature (BODLEVEL is 2.7V)

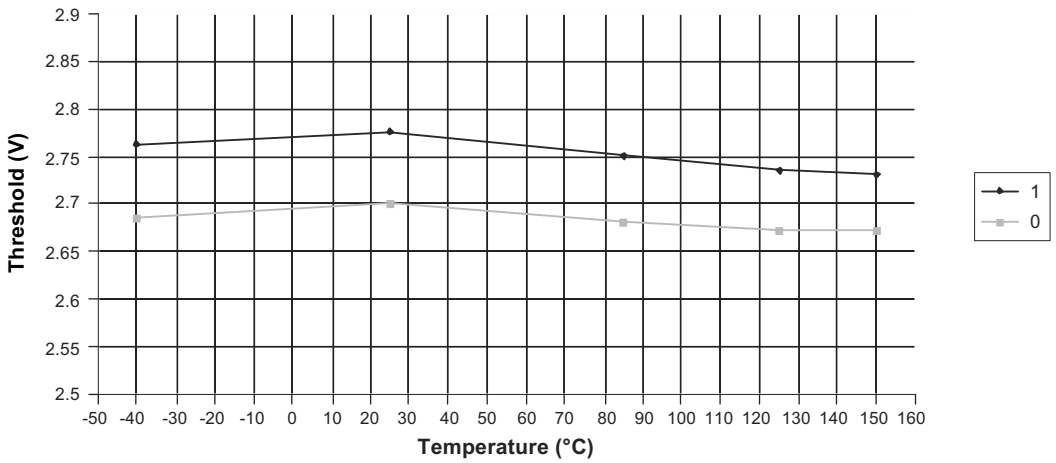
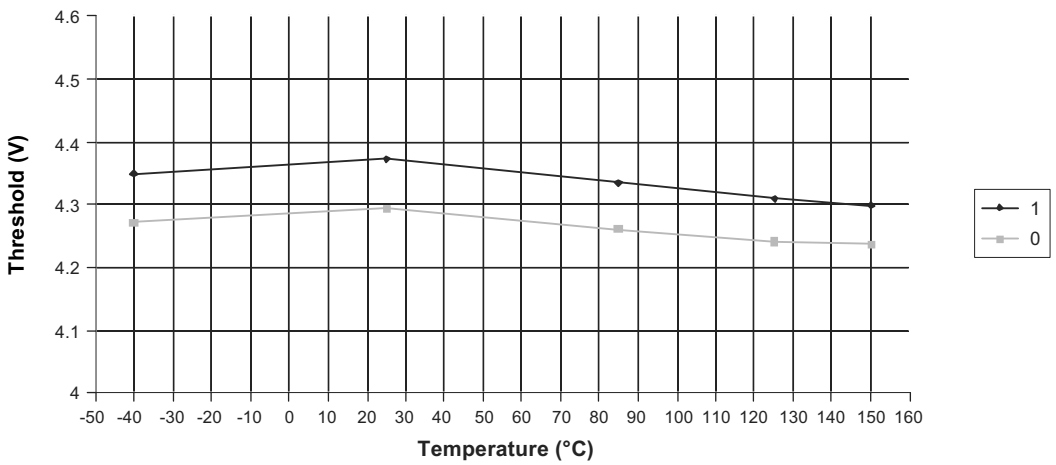


Figure 30-45. BOD Thresholds versus Temperature (BODLEVEL is 4.3V)



33. Ordering Information

33.1 ATmega48PA/88PA/168PA

Speed (MHz)	Power Supply (V)	Ordering Code	Package[1]	Operational Range
16[2]	2.7 - 5.5	ATmega48PA-15AZ	MA	Automotive (–40°C to 125°C)
		ATmega48PA-15MZ	PN	
		ATmega88PA-15AZ	MA	
		ATmega88PA-15MZ	PN	
		ATmega168PA-15AZ	MA	
		ATmega168PA-15MZ	PN	

Notes:
1. Pb-free packaging complies to the european directive for restriction of hazardous substances (RoHS directive).Also halide free and fully green.
2. See Section 29.3 “Speed Grades” on page 270.

Package Type	
MA	MA, 32 - Lead, 7x7mm body size, 1.0mm body thickness 0.8mm lead pitch, thin profile plastic quad flat package (TQFP)
PN	PN, 32-lead, 5.0x5.0mm body, 0.50mm, quad flat no lead package (QFN)