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	20MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	27
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 ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-TQFP
Supplier Device Package	32-TQFP (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atmega48pb-aur

Address	Labels		Code Comments
0x0033	RESET: ldi	r16, high(RAMEND)	; Main program start
0x0034	out	SPH,r16	; Set Stack Pointer to top of RAM
0x0035	ldi	r16, low(RAMEND)	
0x0036	out	SPL,r16	
0x0037	sei		; Enable interrupts
0x0038	<instr>	xxx	
...	...		...

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 ATmega168PB 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 ATmega168PB is:

Address	Labels		Code Comments
.org	0x0002		
0x0002	jmp	EXT_INT0	; IRQ0 Handler
0x0004	jmp	EXT_INT1	; IRQ1 Handler
...	...	...	

Address	Labels		Code Comments
;			
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 ATmega168PB 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	

Related Links

placed in the Application section and Boot Lock bit BLB12 is programed, interrupts are disabled while executing from the Boot Loader section.

Bit 0 – IVCE: Interrupt Vector Change Enable

The IVCE bit must be written to logic one to enable change of the IVSEL bit. IVCE is cleared by hardware four cycles after it is written or when IVSEL is written. Setting the IVCE bit will disable interrupts, as explained in the IVSEL description above. See Code Example below.

Assembly Code Example

```
Move_interrupts:
; Get MCUCR
in    r16, MCUCR
mov   r17, r16
; Enable change of Interrupt Vectors
ori   r16, (1<<IVCE)
out   MCUCR, r16
; Move interrupts to Boot Flash section
ori   r17, (1<<IVSEL)
out   MCUCR, r17
ret
```

C Code Example

```
void Move_interrupts(void)
{
uchar temp;
/* GET MCUCR*/
temp = MCUCR;
/* Enable change of Interrupt Vectors */
MCUCR = temp|(1<<IVCE);
/* Move interrupts to Boot Flash section */
MCUCR = temp|(1<<IVSEL);
}
```

- PCINT4: Pin Change Interrupt source 4. The PB4 pin can serve as an external interrupt source.

- MOSI0/TXD1/OC2A/PCINT3 – Port B, Bit 3
  - MOSI0: SPI0 Master Data output, Slave Data input for SPI0 channel. When the SPI0 is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB3. When the SPI0 is enabled as a Master, the data direction of this pin is controlled by DDB3. When the pin is forced by the SPI0 to be an input, the pull-up can still be controlled by the PORTB3 bit.
  - TXD1: Transmit Data (Data output pin for the USART1). When the USART1 Transmitter is enabled, this pin is configured as an output regardless of the value of DDB3.
  - OC2A: Output Compare Match output. The PB3 pin can serve as an external output for the Timer/Counter2 Compare Match A. The PB3 pin has to be configured as an output (DDB3 set '1') to serve this function. The OC2A pin is also the output pin for the PWM mode timer function.
  - PCINT3: Pin Change Interrupt source 3. The PB3 pin can serve as an external interrupt source.

- SS0/OC1B/PCINT2 – Port B, Bit 2
  - SS0: Slave0 Select input. When the SPI0 is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB2. As a Slave, the SPI0 is activated when this pin is driven low. When the SPI0 is enabled as a Master, the data direction of this pin is controlled by DDB2. When the pin is forced by the SPI0 to be an input, the pull-up can still be controlled by the PORTB2 bit.
  - OC1B: Output Compare Match output. The PB2 pin can serve as an external output for the Timer/Counter1 Compare Match B. The PB2 pin has to be configured as an output (DDB2 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.
  - PCINT2: Pin Change Interrupt source 2. The PB2 pin can serve as an external interrupt source.

- OC1A/PCINT1 – Port B, Bit 1
  - OC1A: Output Compare Match output. The PB1 pin can serve as an external output for the Timer/Counter1 Compare Match A. The PB1 pin has to be configured as an output (DDB1 set (one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.
  - PCINT1: Pin Change Interrupt source 1. The PB1 pin can serve as an external interrupt source.

- ICP1/CLKO/PCINT0 – Port B, Bit 0
  - ICP1: Input Capture Pin. The PB0 pin can act as an Input Capture Pin for Timer/Counter1.
  - CLKO: Divided System Clock. The divided system clock can be output on the PB0 pin. The divided system clock will be output if the CKOUT Fuse is programmed, regardless of the PORTB0 and DDB0 settings. It will also be output during reset.
  - PCINT0: Pin Change Interrupt source 0. The PB0 pin can serve as an external interrupt source.

Table 19-3 Port B Pins Alternate Functions and Table 19-5 Overriding Signals for Alternate Functions in PB3...PB0 relate the alternate functions of Port B to the overriding signals shown in Figure 19-5 Alternate Port Functions(1). SPI MSTR INPUT and SPI SLAVE OUTPUT constitute the MISO signal, while MOSI is divided into SPI MSTR OUTPUT and SPI SLAVE INPUT.

19.4.13. Port E Input Pins Address

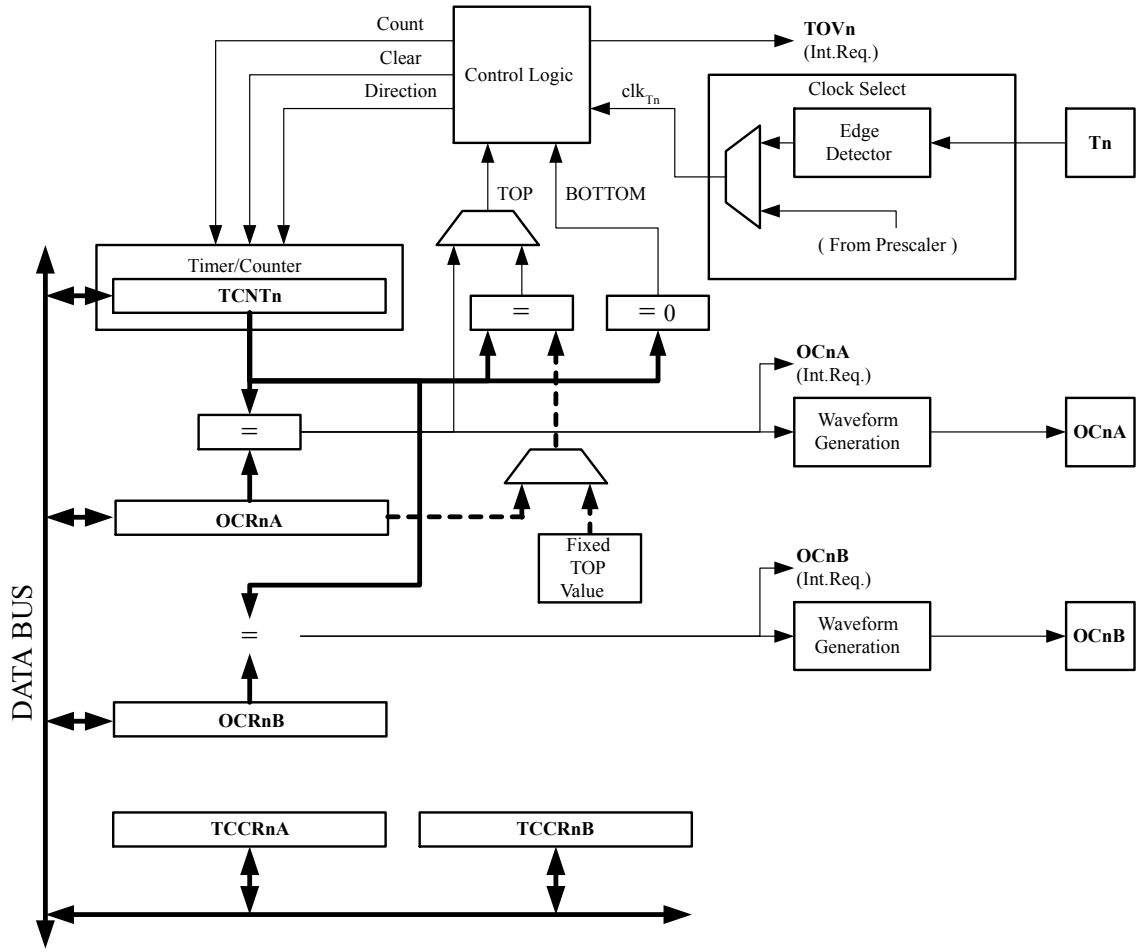
When addressing I/O Registers as data space using LD and ST instructions, the provided offset must be used. When using the I/O specific commands IN and OUT, the offset is reduced by 0x20, resulting in an I/O address offset within 0x00 - 0x3F.

Name: PINE
Offset: 0x2C
Reset: N/A
Property: When addressing as I/O Register: address offset is 0x0C

Bit	7	6	5	4	3	2	1	0
					PINE3	PINE2	PINE1	PINE0
Access					R/W	R/W	R/W	R/W
Reset					x	x	x	x

Bits 3:0 – PINEn: Port E Input Pins Address [n = 3:0]
Writing to the pin register provides toggle functionality for I/O.

Figure 20-1. 8-bit Timer/Counter Block Diagram



Related Links

Minimizing Power Consumption on page 63

Register Description on page 151

Pin Configurations on page 14

20.2.1. Definitions

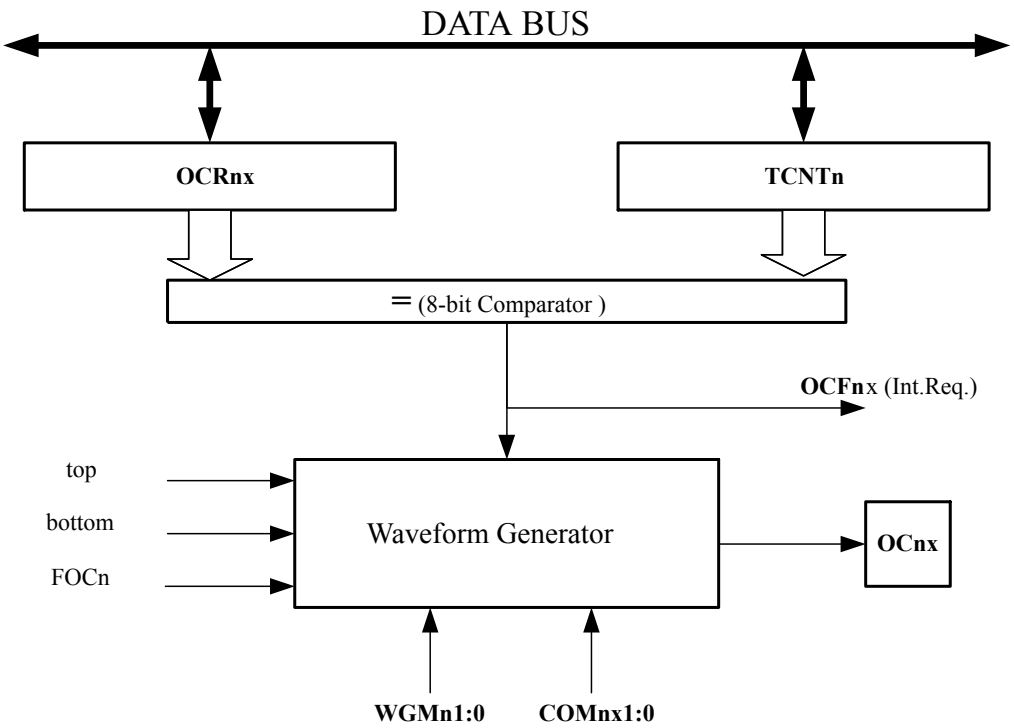
Many register and bit references in this section are written in general form:

- n=1 represents the Timer/Counter number
- x=A,B represents the Output Compare Unit A or B

However, when using the register or bit definitions in a program, the precise form must be used, i.e., TCNT1 for accessing Timer/Counter1 counter value.

The following definitions are used throughout the section:

Figure 20-3. Output Compare Unit, Block Diagram



Note: The “n” in the register and bit names indicates the device number (n = 0 for Timer/Counter 0), and the “x” indicates Output Compare unit (A/B).

The OCR0x Registers are double buffered when using any of the Pulse Width Modulation (PWM) modes. When double buffering is enabled, the CPU has access to the OCR0x Buffer Register. The double buffering synchronizes the update of the OCR0x Compare Registers 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 double buffering is disabled for the normal and Clear Timer on Compare (CTC) modes of operation, and the CPU will access the OCR0x directly.

Related Links

Modes of Operation on page 145

20.5.1. Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a '1' to the Force Output Compare (FOCnx) bit. Forcing compare match will not set the OCFnx Flag or reload/clear the timer, but the OCnx pin will be updated as if a real compare match had occurred (the COMnx[1:0] bits define whether the OCnx pin is set, cleared or toggled).

20.5.2. Compare Match Blocking by TCNTn Write

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

20.5.3. Using the Output Compare Unit

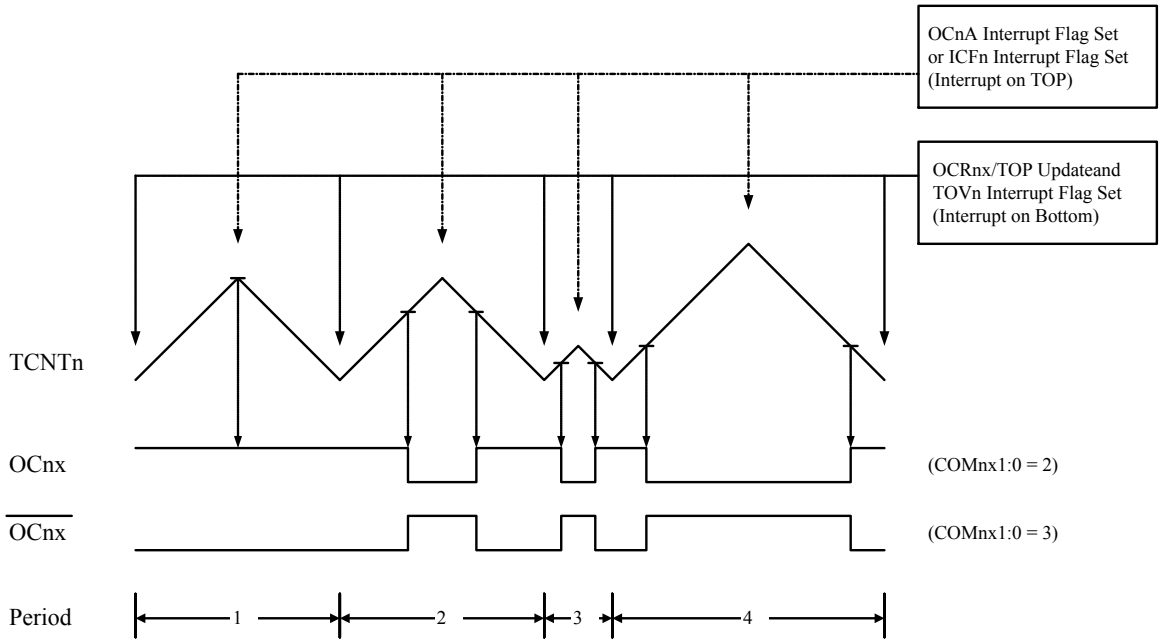
Since writing TCNTn in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNTn when using the Output Compare Unit, independently of whether the Timer/Counter is running or not. If the value written to TCNTn equals the OCRnx value, the

resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated using the following equation:

$$R_{\text{PFCPWM}} = \frac{\log(\text{TOP}+1)}{\log(2)}$$

In phase and frequency correct PWM mode the counter is incremented until the counter value matches either the value in ICR1 (WGM1[3:0]=0x8), or the value in OCR1A (WGM1[3:0]=0x9). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct and frequency correct PWM mode is shown below. The figure shows phase and frequency correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x Interrupt Flag will be set when a compare match occurs.

Figure 21-9. Phase and Frequency Correct PWM Mode, Timing Diagram



Note: The “n” in the register and bit names indicates the device number (n = 1 for Timer/Counter 1), and the “x” indicates Output Compare unit (A/B).

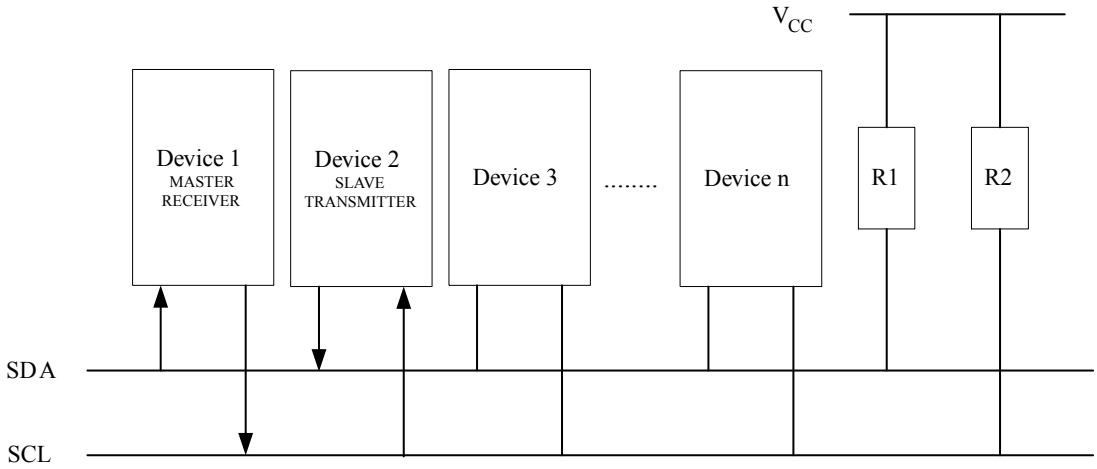
The Timer/Counter Overflow Flag (TOV1) is set at the same timer clock cycle as the OCR1x Registers are updated with the double buffer value (at BOTTOM). When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 Flag set when TCNT1 has reached TOP. The Interrupt Flags can then be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x.

As shown in the timing diagram above, the output generated is, in contrast to the phase correct mode, symmetrical in all periods. Since the OCR1x Registers are updated at BOTTOM, the length of the rising and the falling slopes will always be equal. This gives symmetrical output pulses and is therefore frequency correct.

Using the ICR1 Register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A Register is free to be used for generating a PWM output on OC1A. However, if the base PWM

Figure 27-13. Data Transfer in Master Receiver Mode



A START condition is sent by writing to the TWI Control register (TWCRn) a value of the type TWCRn=1x10x10x:

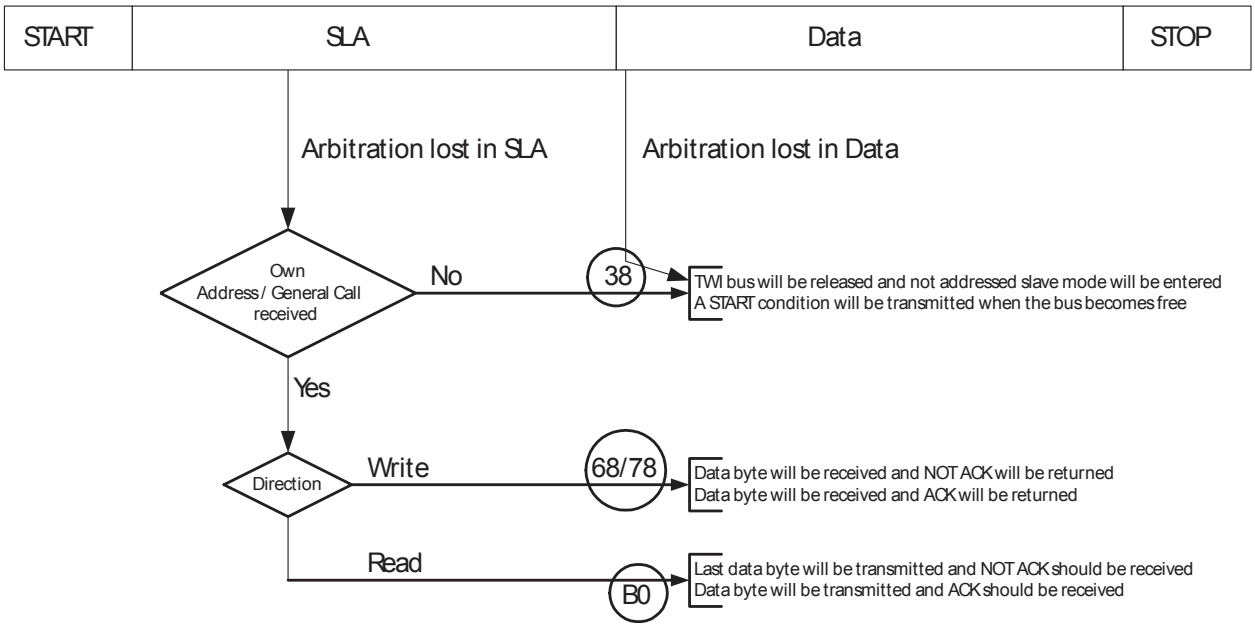
- TWCRn.TWEN must be written to '1' to enable the 2-wire Serial Interface
- TWCRn.TWSTA must be written to '1' to transmit a START condition
- TWCRn.TWINT must be cleared by writing a '1' to it.

The TWI will then test the 2-wire Serial Bus and generate a START condition as soon as the bus becomes free. After a START condition has been transmitted, the TWINT Flag is set by hardware, and the status code in TWSRn will be 0x08 (see Status Code table below). In order to enter MR mode, SLA +R must be transmitted. This is done by writing SLA+R to TWDR. Thereafter, the TWINT flag should be cleared (by writing '1' to it) to continue the transfer. This is accomplished by writing the a value to TWCRn of the type TWCRn=1x00x10x.

When SLA+R have been transmitted and an acknowledgment bit has been received, TWINT is set again and a number of status codes in TWSRn are possible. Possible status codes in Master mode are 0x38, 0x40, or 0x48. The appropriate action to be taken for each of these status codes is detailed in the table below. Received data can be read from the TWDR Register when the TWINT Flag is set high by hardware. This scheme is repeated until the last byte has been received. After the last byte has been received, the MR should inform the ST by sending a NACK after the last received data byte. The transfer is ended by generating a STOP condition or a repeated START condition. A repeated START condition is sent by writing to the TWI Control register (TWCRn) a value of the type TWCRn=1x10x10x again. A STOP condition is generated by writing TWCRn=1xx01x10x:

After a repeated START condition (status code 0x10) the 2-wire Serial Interface can access the same Slave again, or a new Slave without transmitting a STOP condition. Repeated START enables the Master to switch between Slaves, Master Transmitter mode and Master Receiver mode without losing control over the bus.

Figure 27-21. Possible Status Codes Caused by Arbitration



27.9. Register Description

27.9.2. TWI Status Register

Name: TWSR
Offset: 0xB9
Reset: 0xF8
Property: -

Bit	7	6	5	4	3	2	1	0
	TWS7	TWS6	TWS5	TWS4	TWS3		TWPS1	TWPS0
Access	R	R	R	R	R		R/W	R/W
Reset	1	1	1	1	1		0	0

Bit 7 – TWS7: TWI Status Bit 7
The TWS[7:3] 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 6 – TWS6: TWI Status Bit 6

Bit 5 – TWS5: TWI Status Bit 5

Bit 4 – TWS4: TWI Status Bit 4

Bit 3 – TWS3: TWI Status Bit 3

Bits 1:0 – TWPSn: TWI Prescaler [n = 1:0]
These bits can be read and written, and control the bit rate prescaler.

Table 27-8. TWI Bit Rate Prescaler

TWS[1:0]	Prescaler Value
00	1
01	4
10	16
11	64

To calculate bit rates, refer to Bit Rate Generator Unit. The value of TWPS1...0 is used in the equation.

27.9.5. TWI Control Register

The TWCR is used to control the operation of the TWI. It is used to enable the TWI, to initiate a Master access by applying a START condition to the bus, to generate a Receiver acknowledge, to generate a stop condition, and to control halting of the bus while the data to be written to the bus are written to the TWDR. It also indicates a write collision if data is attempted written to TWDR while the register is inaccessible.

Name: TWCR
Offset: 0xBC
Reset: 0x00
Property: -

Bit	7	6	5	4	3	2	1	0
	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN		TWIE
Access	R/W	R/W	R/W	R/W	R/W	R/W		R/W
Reset	0	0	0	0	0	0		0

Bit 7 – TWINT: TWI Interrupt Flag

This bit is set by hardware when the TWI has finished its current job and expects application software response. If the I-bit in SREG and TWIE in TWCR are set, the MCU will jump to the TWI Interrupt Vector. While the TWINT Flag is set, the SCL low period is stretched. The TWINT Flag must be cleared by software by writing a logic one to it.

Note that this flag is not automatically cleared by hardware when executing the interrupt routine. Also note that clearing this flag starts the operation of the TWI, so all accesses to the TWI Address Register (TWAR), TWI Status Register (TWSR), and TWI Data Register (TWDR) must be complete before clearing this flag.

Bit 6 – TWEA: TWI Enable Acknowledge

The TWEA bit controls the generation of the acknowledge pulse. If the TWEA bit is written to one, the ACK pulse is generated on the TWI bus if the following conditions are met:

1. The device's own slave address has been received.
2. A general call has been received, while the TWGCE bit in the TWAR is set.
3. A data byte has been received in Master Receiver or Slave Receiver mode.

By writing the TWEA bit to zero, the device can be virtually disconnected from the 2-wire Serial Bus temporarily. Address recognition can then be resumed by writing the TWEA bit to one again.

Bit 5 – TWSTA: TWI START Condition

The application writes the TWSTA bit to one when it desires to become a Master on the 2-wire Serial Bus. The TWI hardware checks if the bus is available, and generates a START condition on the bus if it is free. However, if the bus is not free, the TWI waits until a STOP condition is detected, and then generates a new START condition to claim the bus Master status. TWSTA must be cleared by software when the START condition has been transmitted.

Bit 4 – TWSTO: TWI STOP Condition

Writing the TWSTO bit to one in Master mode will generate a STOP condition on the 2-wire Serial Bus. When the STOP condition is executed on the bus, the TWSTO bit is cleared automatically. In Slave mode, setting the TWSTO bit can be used to recover from an error condition. This will not generate a

29.6.2. Analog Noise Canceling Techniques

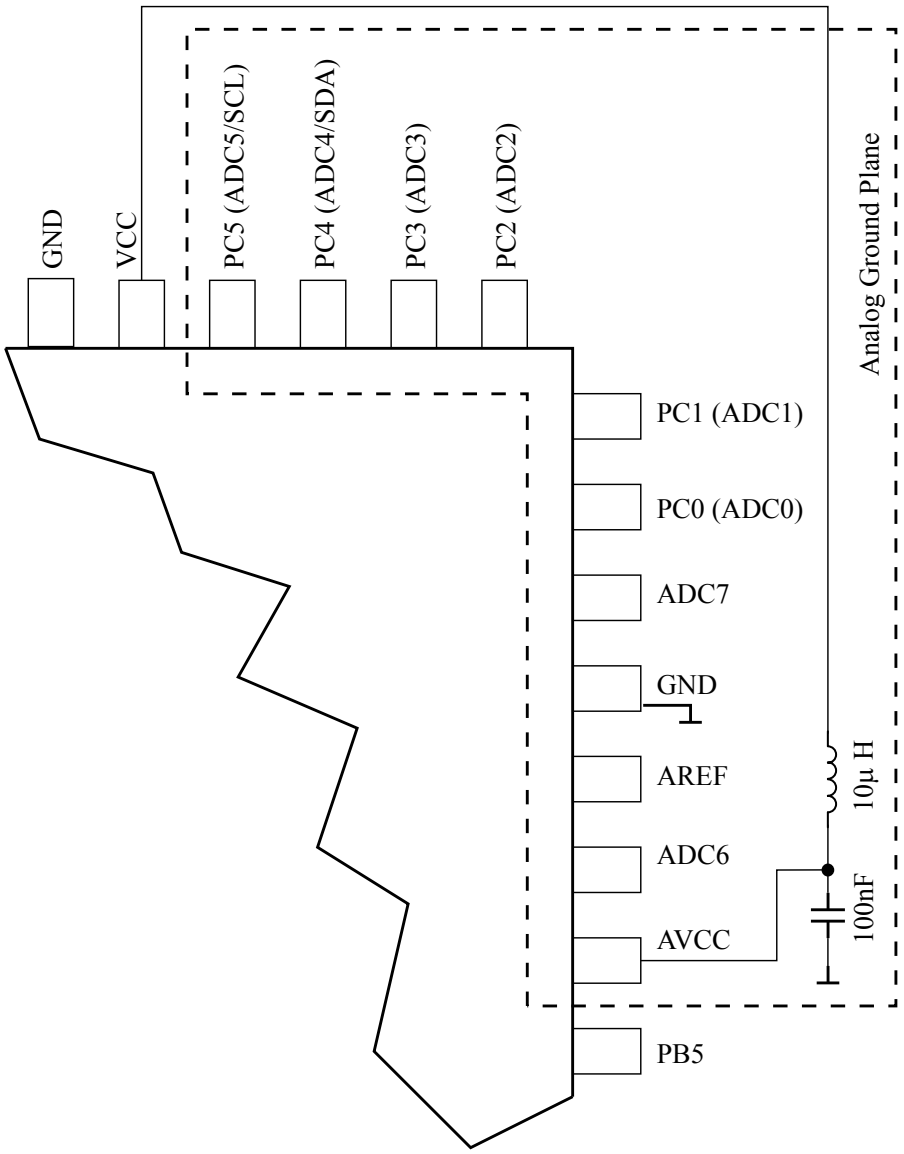
Digital circuitry inside and outside the device generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

1. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.
   1.1. The AV_{CC} pin on the device should be connected to the digital V_{CC} supply voltage via an LC network as shown in the figure below.

   1.2. Use the ADC noise canceler function to reduce induced noise from the CPU.

   1.3. If any ADC [3:0] port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress. However, using the 2-wire Interface (ADC4 and ADC5) will only affect the conversion on ADC4 and ADC5 and not the other ADC channels.

Figure 29-9. ADC Power Connections



```
    rjmp Wait_ee
    ; SPM timed sequence
    out SPMCSR, spmcrval
    spm
    ; restore SREG (to enable interrupts if originally enabled)
    out SREG, temp2
    ret
```

32.8.14. ATmega88PB Boot Loader Parameters

The following tables are the parameters used in the description of the self programming are given.

Table 32-7. Boot Size Configuration, ATmega88PB

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	128 words	4	0x000 - 0xF7F	0xF80 - 0xFFF	0xF7F	0xF80
1	0	256 words	8	0x000 - 0xEFF	0xF00 - 0xFFF	0xEFF	0xF00
0	1	512 words	16	0x000 - 0xDFF	0xE00 - 0xFFF	0xDFF	0xE00
0	0	1024 words	32	0x000 - 0xBFF	0xC00 - 0xFFF	0xBFF	0xC00

Note: The different BOOTSZ Fuse configurations are shown in Figure 32-2 Memory Sections.

Table 32-8. Read-While-Write Limit, ATmega88PB

Section	Pages	Address
Read-While-Write section (RWW)	96	0x000 - 0xBFF
No Read-While-Write section (NRWW)	32	0xC00 - 0xFFF

For details about these two section, please refer to NRWW – No Read-While-Write Section and RWW – Read-While-Write Section.

Table 32-9. Explanation of Different Variables used in Figure 32-3 Addressing the Flash During SPM

Variable		Corresponding Z-value(1)	Description
PCMSB	11		Most significant bit in the Program Counter. (The Program Counter is 12 bits PC[11:0])
PAGEMSB	4		Most significant bit which is used to address the words within one page (32 words in a page requires 5 bits PC [4:0]).

Table 32-12. Explanation of Different Variables used in Figure 32-3 Addressing the Flash During SPM ATmega168PB

Variable		Corresponding Z-value(1)	Description
PCMSB	12		Most significant bit in the Program Counter. (The Program Counter is 13 bits PC[12:0])
PAGEMSB	5		Most significant bit which is used to address the words within one page (64 words in a page requires 6 bits PC [5:0])
ZPCMSB		Z13	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMSB		Z6	Bit in Z-register that is mapped to PAGEMSB. Because Z0 is not used, the ZPAGEMSB equals PAGEMSB + 1.
PCPAGE	PC[12:6]	Z13:Z7	Program counter page address: Page select, for page erase and page write
PCWORD	PC[5:0]	Z6:Z1	Program counter word address: Word select, for filling temporary buffer (must be zero during page write operation)

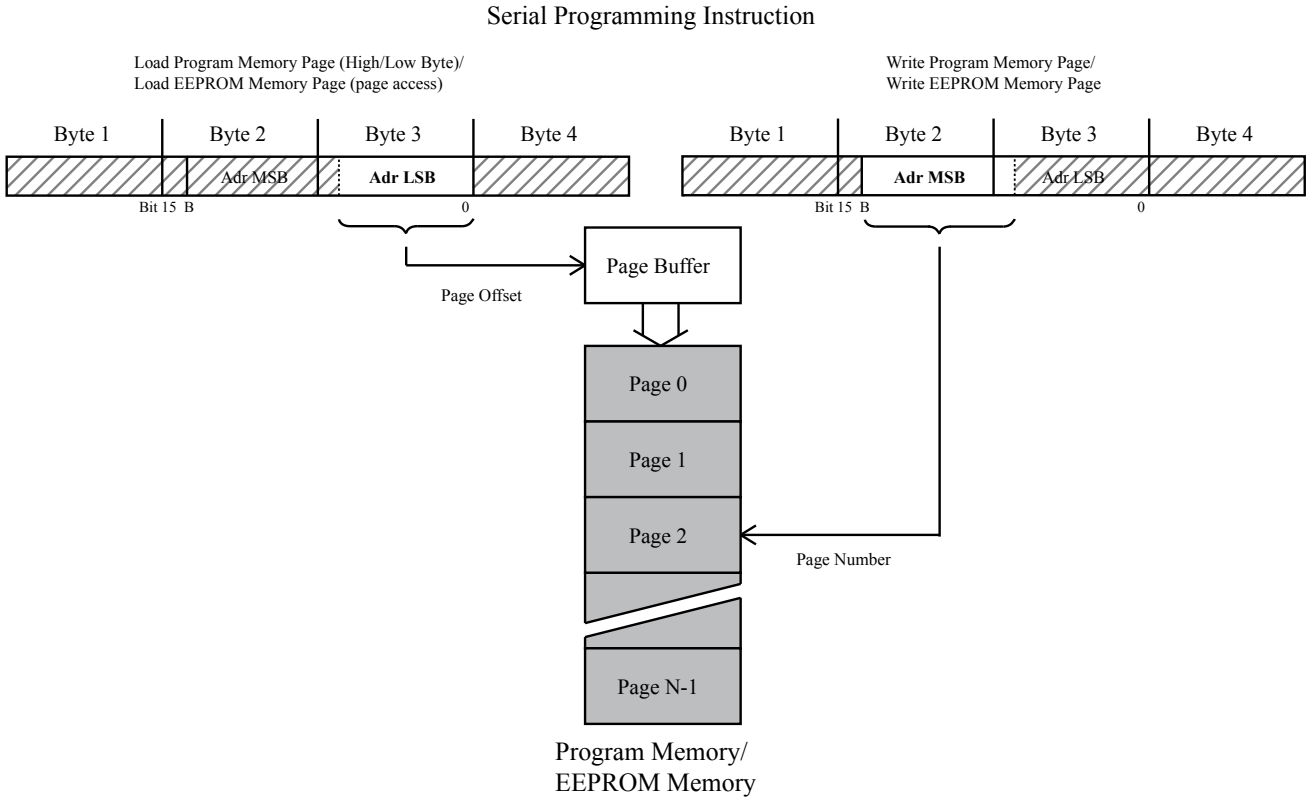
Note: 1. Z15:Z14: always ignored

Z0: should be zero for all SPM commands, byte select for the LPM instruction.

Please refer to Addressing the Flash During Self-Programming or details about the use of Z-pointer during Self- Programming.

32.9. Register Description

Figure 33-7. Serial Programming Instruction example



33.8.4. SPI Serial Programming Characteristics

Figure 33-8. Serial Programming Waveforms

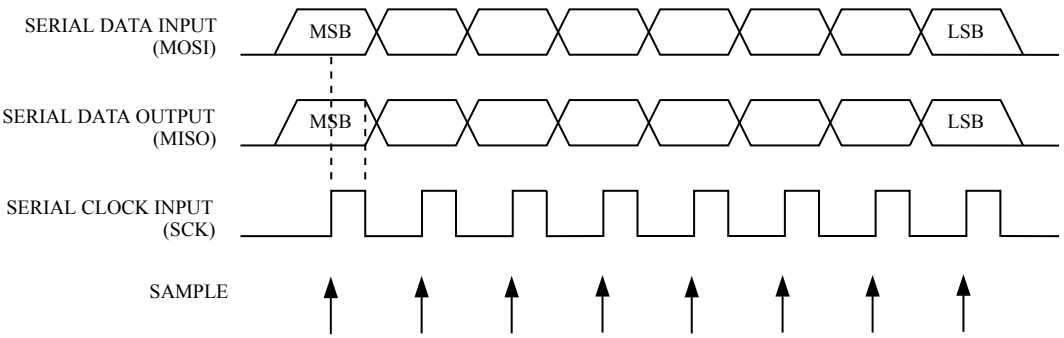
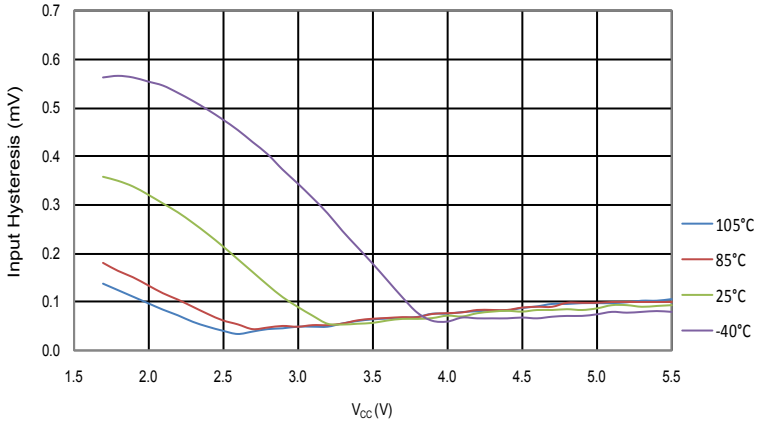


Figure 35-30. ATmega48PB/88PB: Reset Pin Input Hysteresis vs. V_{CC}



35.1.10. BOD Threshold

Figure 35-31. ATmega48PB/88PB: BOD Thresholds vs. Temperature (BODLEVEL is 1.8V)

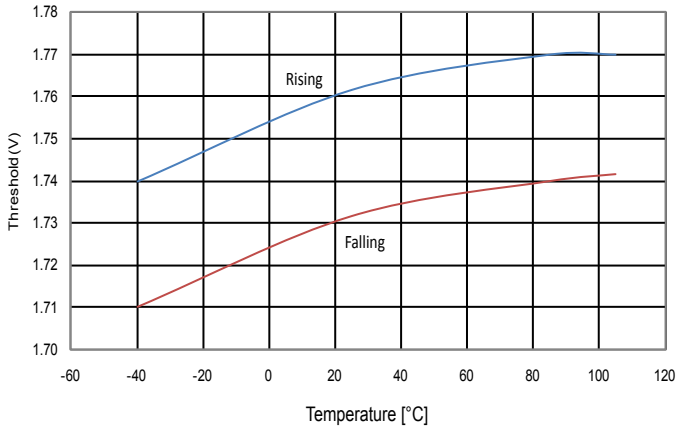


Figure 35-32. ATmega48PB/88PB: BOD Thresholds vs. Temperature (BODLEVEL is 2.7V)

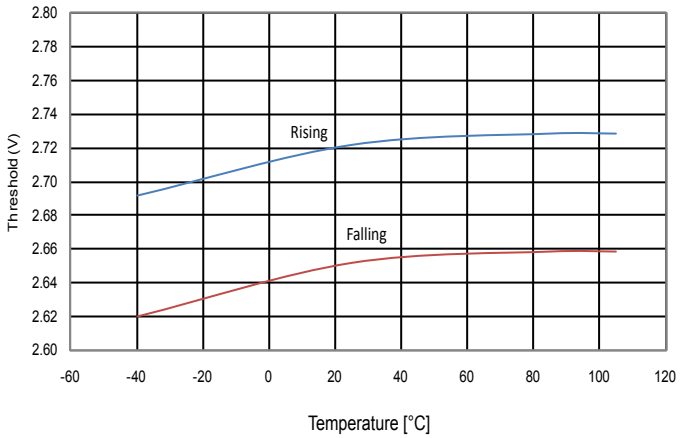
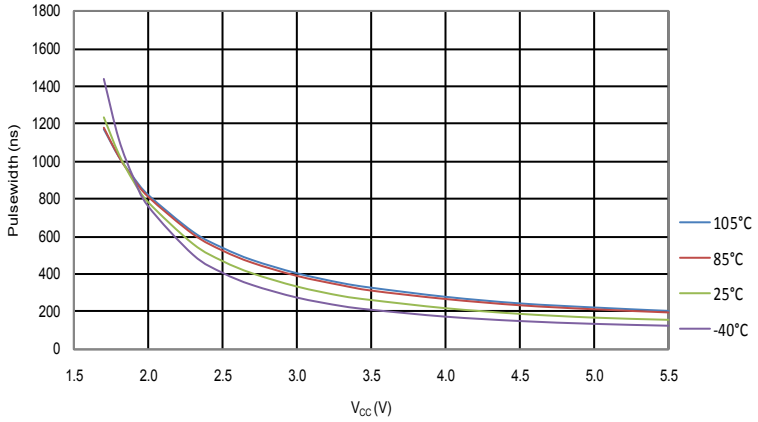


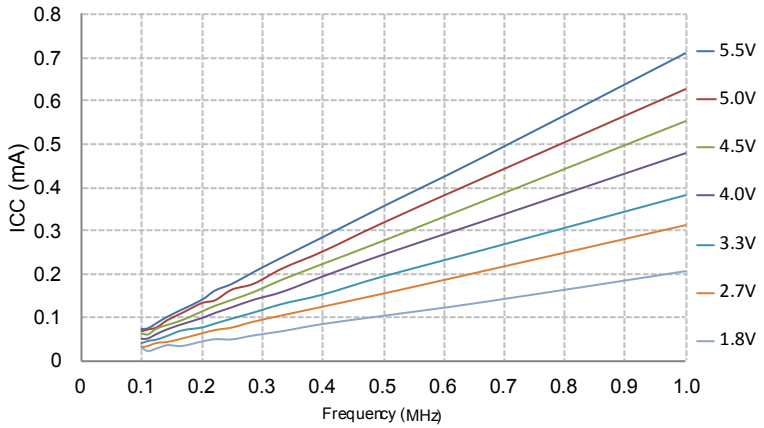
Figure 35-48. ATmega48PB_88PB: Minimum Reset Pulse Width vs. Vcc



35.2. ATmega168PB Typical Characteristics

35.2.1. Active Supply Current

Figure 35-49. ATmega168PB: Active Supply Current vs. Low Frequency (0.1-1.0MHz)



Offset	Name	Bit Pos.								
0x86	ICR1L	7:0	ICR1L[7:0]							
0x87	ICR1H	7:0	ICR1H[7:0]							
0x88	OCR1AL	7:0	OCR1AL[7:0]							
0x89	OCR1AH	7:0	OCR1AH[7:0]							
0x8A	OCR1BL	7:0	OCR1BL[7:0]							
0x8B	OCR1BH	7:0	OCR1BH[7:0]							
0x8C ... 0xAF	Reserved									
0xB0	TCCR2A	7:0	COM2A1	COM2A0	COM2B1	COM2B0			WGM21	WGM20
0xB1	TCCR2B	7:0	FOC2A	FOC2B			WGM22	CS22	CS21	CS20
0xB2	TCNT2	7:0	TCNT2[7:0]							
0xB3	OCR2A	7:0	OCR2A[7:0]							
0xB4	OCR2B	7:0	OCR2B[7:0]							
0xB5	Reserved									
0xB6	ASSR	7:0		EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
0xB7	Reserved									
0xB8	TWBR	7:0	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0
0xB9	TWSR	7:0	TWS7	TWS6	TWS5	TWS4	TWS3		TWPS1	TWPS0
0xBA	TWAR	7:0	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
0xBB	TWDR	7:0	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0
0xBC	TWCR	7:0	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN		TWIE
0xBD	TWAMR	7:0	TWAM[6:0]							
0xBE ... 0xBF	Reserved									
0xC0	UCSR0A	7:0	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0
0xC1	UCSR0B	7:0	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80
0xC2	UCSR0C	7:0	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 / UDORD0	UCSZ00 / UCPHA0	UCPOL0
0xC3	UCSR0D	7:0	RXIE	RXS	SFDE					
0xC4	UBRR0L	7:0	UBRR0[7:0]							
0xC5	UBRR0H	7:0					UBRR0[3:0]			
0xC6	UDR0	7:0	TXB / RXB[7:0]							