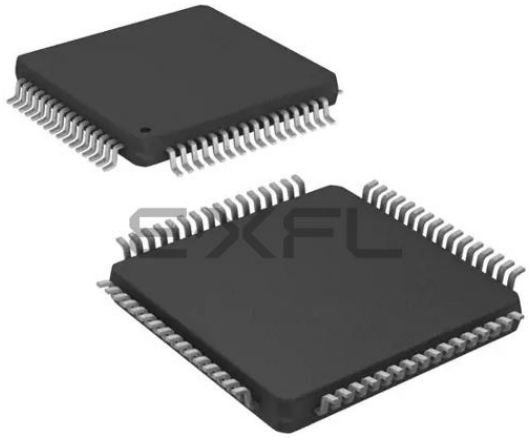


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

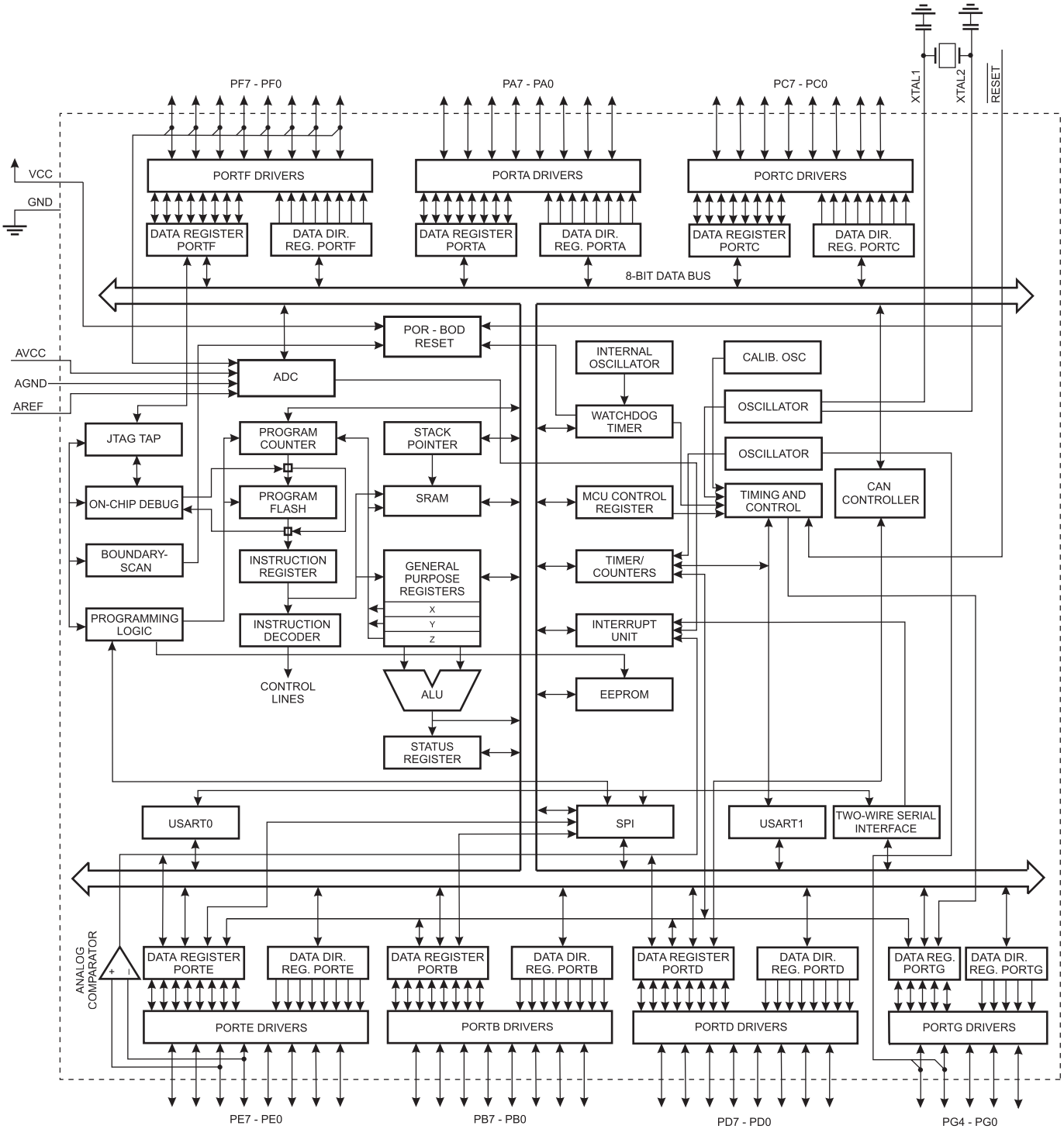
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded"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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	8-Bit
Speed	16MHz
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	53
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at90can32-15az

1.5 Block Diagram

Figure 1-1. Block Diagram



4. Memories

This section describes the different memories in the AT90CAN32/64/128. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the AT90CAN32/64/128 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

Table 4-1. Memory Mapping.

Memory		Mnemonic	AT90CAN32	AT90CAN64	AT90CAN128
Flash	*Size*	Flash size	32 K bytes	64 K bytes	128 K bytes
	Start Address	-	0x00000		
	End Address	Flash end	0x07FFF(1) 0x3FFF(2)	0x0FFFF(1) 0x7FFF(2)	0x1FFFF(1) 0xFFFF(2)
32 Registers	*Size*	-	32 bytes		
	Start Address	-	0x0000		
	End Address	-	0x001F		
I/O Registers	*Size*	-	64 bytes		
	Start Address	-	0x0020		
	End Address	-	0x005F		
Ext I/O Registers	*Size*	-	160 bytes		
	Start Address	-	0x0060		
	End Address	-	0x00FF		
Internal SRAM	*Size*	ISRAM size	2 K bytes	4 K bytes	4 K bytes
	Start Address	ISRAM start	0x0100		
	End Address	ISRAM end	0x08FF	0x10FF	0x10FF
External Memory	*Size*	XMem size	0-64 K bytes		
	Start Address	XMem start	0x0900	0x1100	0x1100
	End Address	XMem end	0xFFFF		
EEPROM	*Size*	E2 size	1 K bytes	2 K bytes	4 K bytes
	Start Address	-	0x0000		
	End Address	E2 end	0x03FF	0x07FF	0x0FFF

Notes: 1. Byte address.
2. Word (16-bit) address.

4.1 In-System Reprogrammable Flash Program Memory

The AT90CAN32/64/128 contains On-chip In-System Reprogrammable Flash memory for program storage (see “Flash size”). Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 16 bits wide. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The AT90CAN32/64/128 Program Counter (PC) address the program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in “Boot Loader Support – Read-While-Write Self-Programming” on page 320. “Memory Programming” on page 335 contains a detailed description on Flash data serial downloading using the SPI pins or the JTAG interface.

4.4 I/O Memory

The I/O space definition of the AT90CAN32/64/128 is shown in "Register Summary" on page 384.

All AT90CAN32/64/128 I/Os and peripherals are placed in the I/O space. All I/O locations may be accessed by the LD/LDS/LDD and ST/STS/STD instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O registers as data space using LD and ST instructions, 0x20 must be added to these addresses. The AT90CAN32/64/128 is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the status flags are cleared by writing a logical one to them. Note that, unlike most other AVR's, the CBI and SBI instructions will only operate on the specified bit, and can therefore be used on registers containing such status flags. The CBI and SBI instructions work with registers 0x00 to 0x1F only.

The I/O and peripherals control registers are explained in later sections.

4.5 External Memory Interface

With all the features the External Memory Interface provides, it is well suited to operate as an interface to memory devices such as External SRAM and Flash, and peripherals such as LCD-display, A/D, and D/A. The main features are:

- Four different wait-state settings (including no wait-state).
- Independent wait-state setting for different extErnal Memory sectors (configurable sector size).
- The number of bits dedicated to address high byte is selectable.
- Bus keepers on data lines to minimize current consumption (optional).

4.5.1 Overview

When the e**X**ternal **MEM**ory (XMEM) is enabled, address space outside the internal SRAM becomes available using the dedicated External Memory pins (see Figure 1-2 on page 5, Table 9-3 on page 74, Table 9-9 on page 78, and Table 9-21 on page 88). The memory configuration is shown in Figure 4-4.

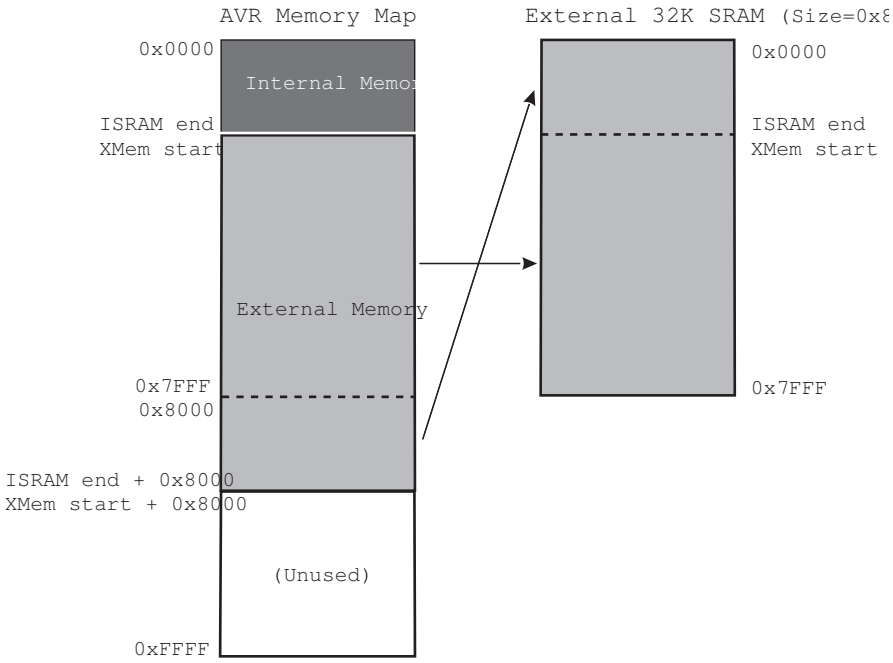
Table 4-5. Port C Pins Released as Normal Port Pins when the External Memory is Enabled

XMM2	XMM1	XMM0	# Bits for External Memory Address	Released Port Pins
0	0	0	8 (Full External Memory Space)	None
0	0	1	7	PC7
0	1	0	6	PC7 .. PC6
0	1	1	5	PC7 .. PC5
1	0	0	4	PC7 .. PC4
1	0	1	3	PC7 .. PC3
1	1	0	2	PC7 .. PC2
1	1	1	No Address high bits	Full Port C

4.5.8 Using all Locations of External Memory Smaller than 64 KB

Since the external memory is mapped after the internal memory as shown in Figure 4-4, the external memory is not addressed when addressing the first “ISRAM size” bytes of data space. It may appear that the first “ISRAM size” bytes of the external memory are inaccessible (external memory addresses 0x0000 to “ISRAM end”). However, when connecting an external memory smaller than 64 KB, for example 32 KB, these locations are easily accessed simply by addressing from address 0x8000 to “ISRAM end + 0x8000”. Since the External Memory Address bit A15 is not connected to the external memory, addresses 0x8000 to “ISRAM end + 0x8000” will appear as addresses 0x0000 to “ISRAM end” for the external memory. Addressing above address “ISRAM end + 0x8000” is not recommended, since this will address an external memory location that is already accessed by another (lower) address. To the Application software, the external 32 KB memory will appear as one linear 32 KB address space from “XMem start” to “XMem start + 0x8000”. This is illustrated in Figure 4-10.

Figure 4-10. Address Map with 32 KB External Memory



“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 5-12. 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

Note: The frequency of the asynchronous clock must be lower than 1/4th of the frequency of the scaled down Source clock. Otherwise, interrupts may be lost, and accessing the Timer/Counter2 registers may fail.

Table 9-2. Generic Description of Overriding Signals for Alternate Functions

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
PTOE	Port Toggle Override Enable	If PTOE is set, the PORTxn Register bit is inverted.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU state (Normal mode, sleep mode).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, sleep mode).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/Output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

9.3.1 MCU Control Register – MCUCR

Bit	7	6	5	4	3	2	1	0	
	JTD	–	–	PUD	–	–	IVSEL	IVCE	MCUCR
Read/Write	R/W	R	R	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

MOSI, SPI Master Data output, Slave Data input for SPI channel. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB2. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB2. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB2 bit.

- **SCK – Port B, Bit 1**

SCK, Master Clock output, Slave Clock input pin for SPI channel. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB1. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB1. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB1 bit.

- **$\overline{SS}$ – Port B, Bit 0**

$\overline{SS}$, Slave Port Select input. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB0. As a slave, the SPI is activated when this pin is driven low. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB0. When the pin is forced to be an input, the pull-up can still be controlled by the PORTB0 bit.

Table 9-7 and Table 9-8 relate the alternate functions of Port B to the overriding signals shown in Figure 9-5 on page 72. SPI MSTR INPUT and SPI SLAVE OUTPUT constitute the MISO signal, while MOSI is divided into SPI MSTR OUTPUT and SPI SLAVE INPUT.

Table 9-7 and Table 9-8 relates the alternate functions of Port B to the overriding signals shown in Figure 9-5 on page 72.

Table 9-7. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/OC0A/OC1C	PB6/OC1B	PB5/OC1A	PB4/OC2A
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	OC0A/OC1C ENABLE(1)	OC1B ENABLE	OC1A ENABLE	OC2A ENABLE
PVOV	OC0A/OC1C(1)	OC1B	OC1A	OC2A
PTOE	0	0	0	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	–	–	–	–

Note: 1. See "Output Compare Modulator - OCM" on page 164 for details.

- **AIN0/XCK0 – Port E, Bit 2**

AIN0 – Analog Comparator Positive input. This pin is directly connected to the positive input of the Analog Comparator.

XCK0, USART0 External clock. The Data Direction Register (DDE2) controls whether the clock is output (DDE2 set) or input (DDE2 cleared). The XCK0 pin is active only when the USART0 operates in Synchronous mode.

- **PDO/TXD0 – Port E, Bit 1**

PDO, SPI Serial Programming Data Output. During Serial Program Downloading, this pin is used as data output line for the AT90CAN32/64/128.

TXD0, UART0 Transmit pin.

- **PDI/RXD0 – Port E, Bit 0**

PDI, SPI Serial Programming Data Input. During Serial Program Downloading, this pin is used as data input line for the AT90CAN32/64/128.

RXD0, USART0 Receive Pin. Receive Data (Data input pin for the USART0). When the USART0 receiver is enabled this pin is configured as an input regardless of the value of DDRE0. When the USART0 forces this pin to be an input, a logical one in PORTE0 will turn on the internal pull-up.

Table 9-16 and Table 9-17 relates the alternate functions of Port E to the overriding signals shown in Figure 9-5 on page 72.

Table 9-16. Overriding Signals for Alternate Functions PE7..PE4

Signal Name	PE7/INT7/ICP3	PE6/INT6/T3	PE5/INT5/OC3C	PE4/INT4/OC3B
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	OC3C ENABLE	OC3B ENABLE
PVOV	0	0	OC3C	OC3B
PTOE	0	0	0	0
DIEOE	INT7 ENABLE	INT6 ENABLE	INT5 ENABLE	INT4 ENABLE
DIEOV	INT7 ENABLE	INT6 ENABLE	INT5 ENABLE	INT4 ENABLE
DI	INT7 INPUT /ICP3 INPUT	INT6 INPUT /T3 INPUT	INT5 INPUT	INT4 INPUT
AIO	–	–	–	–

uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{Tn}).

The double buffered Output Compare Registers (OCRnx) are compared with the Timer/Counter value at all time. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pin (OCnx). See “Output Compare Units” on page 123.. The compare match event will also set the Compare Match Flag (OCFnx) which can be used to generate an Output Compare interrupt request.

The Input Capture Register can capture the Timer/Counter value at a given external (edge triggered) event on either the Input Capture pin (ICPn) or on the Analog Comparator pins (See “Analog Comparator” on page 268.) The Input Capture unit includes a digital filtering unit (Noise Canceler) for reducing the chance of capturing noise spikes.

The TOP value, or maximum Timer/Counter value, can in some modes of operation be defined by either the OCRnA Register, the ICRn Register, or by a set of fixed values. When using OCRnA as TOP value in a PWM mode, the OCRnA Register can not be used for generating a PWM output. However, the TOP value will in this case be double buffered allowing the TOP value to be changed in run time. If a fixed TOP value is required, the ICRn Register can be used as an alternative, freeing the OCRnA to be used as PWM output.

13.2.2 Definitions

The following definitions are used extensively throughout the section:

BOTTOM	The counter reaches the BOTTOM when it becomes 0x0000.
MAX	The counter reaches its MAXimum when it becomes 0xFFFF (decimal 65,535).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values: 0x00FF, 0x01FF, or 0x03FF, or to the value stored in the OCRnA or ICRn Register. The assignment is dependent of the mode of operation.

13.2.3 Compatibility

The 16-bit Timer/Counter has been updated and improved from previous versions of the 16-bit AVR Timer/Counter. This 16-bit Timer/Counter is fully compatible with the earlier version regarding:

- All 16-bit Timer/Counter related I/O Register address locations, including Timer Interrupt Registers.
- Bit locations inside all 16-bit Timer/Counter Registers, including Timer Interrupt Registers.
- Interrupt Vectors.

The following control bits have changed name, but have same functionality and register location:

- PWMn0 is changed to WGMn0.
- PWMn1 is changed to WGMn1.
- CTCn is changed to WGMn2.

The following registers are added to the 16-bit Timer/Counter:

- Timer/Counter Control Register C (TCCRnC).
- Output Compare Register C, OCRnCH and OCRnCL, combined OCRnC.

13.3.1 Code Examples

The following code examples show how to access the 16-bit timer registers assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCRnx and ICRn Registers. Note that when using “C”, the compiler handles the 16-bit access.

Assembly Code Examples[1]

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

C Code Examples[1]

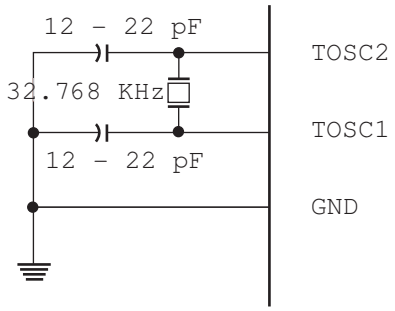
```
unsigned int i;
...
/* Set TCNTn to 0x01FF */
TCNTn = 0x1FF;
/* Read TCNTn into i */
i = TCNTn;
...
```

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

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

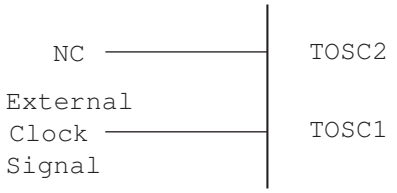
It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit timer registers, then the result of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

Figure 14-14. Timer/Counter2 Crystal Oscillator Connections



A external clock can also be used using TOSC1 as input. Setting AS2 and EXCLK enables this configuration.

Figure 14-15. Timer/Counter2 External Clock Connections



For Timer/Counter2, the possible prescaled selections are: $clk_{T2S}/8$, $clk_{T2S}/32$, $clk_{T2S}/64$, $clk_{T2S}/128$, $clk_{T2S}/256$, and $clk_{T2S}/1024$. Additionally, clk_{T2S} as well as 0 (stop) may be selected. Setting the PSR2 bit in GTCCR resets the prescaler. This allows the user to operate with a predictable prescaler.

14.11.1 General Timer/Counter Control Register – GTCCR

Bit	7	6	5	4	3	2	1	0	
	TSM	–	–	–	–	–	PSR2	PSR310	GTCCR
Read/Write	R/W	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 1 – PSR2: Prescaler Reset Timer/Counter2**

When this bit is one, the Timer/Counter2 prescaler will be reset. This bit is normally cleared immediately by hardware. If the bit is written when Timer/Counter2 is operating in asynchronous mode, the bit will remain one until the prescaler has been reset. The bit will not be cleared by hardware if the TSM bit is set. Refer to the description of the “Bit 7 – TSM: Timer/Counter Synchronization Mode” on page 98 for a description of the Timer/Counter Synchronization mode.

15.2.2 Resolution of the PWM Signal

The resolution of the PWM signal (OC1C) is reduced by the modulation. The reduction factor is equal to the number of system clock cycles of one period of the carrier (OC0A). In this example the resolution is reduced by a factor of two. The reason for the reduction is illustrated in Figure 15-3 at the second and third period of the PB7 output when PORTB7 equals zero. The period 2 high time is one cycle longer than the period 3 high time, but the result on the PB7 output is equal in both periods.

22. JTAG Interface and On-chip Debug System

22.1 Features

- **JTAG (IEEE std. 1149.1 Compliant) Interface**
- **Boundary-scan Capabilities According to the IEEE std. 1149.1 (JTAG) Standard**
- **Debugger Access to:**
  - **All Internal Peripheral Units**
  - **Internal and External RAM**
  - **The Internal Register File**
  - **Program Counter**
  - **EEPROM and Flash Memories**
- **Extensive On-chip Debug Support for Break Conditions, Including**
  - **AVR Break Instruction**
  - **Break on Change of Program Memory Flow**
  - **Single Step Break**
  - **Program Memory Break Points on Single Address or Address Range**
  - **Data Memory Break Points on Single Address or Address Range**
- **Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface**
- **On-chip Debugging Supported by AVR Studio®**

22.2 Overview

The AVR IEEE std. 1149.1 compliant JTAG interface can be used for:

- Testing PCBs by using the JTAG Boundary-scan capability
- Programming the non-volatile memories, Fuses and Lock bits
- On-chip debugging

A brief description is given in the following sections. Detailed descriptions for Programming via the JTAG interface, and using the Boundary-scan Chain can be found in the sections “JTAG Programming Overview” on page 351 and “Boundary-scan IEEE 1149.1 (JTAG)” on page 299, respectively. The On-chip Debug support is considered being private JTAG instructions, and distributed within ATMEL and to selected third party vendors only.

Figure 22-1 shows a block diagram of the JTAG interface and the On-chip Debug system. The TAP Controller is a state machine controlled by the TCK and TMS signals. The TAP Controller selects either the JTAG Instruction Register or one of several Data Registers as the scan chain (Shift Register) between the TDI – input and TDO – output. The Instruction Register holds JTAG instructions controlling the behavior of a Data Register.

The ID-Register (IDentifier Register), Bypass Register, and the Boundary-scan Chain are the Data Registers used for board-level testing. The JTAG Programming Interface (actually consisting of several physical and virtual Data Registers) is used for serial programming via the JTAG interface. The Internal Scan Chain and Break Point Scan Chain are used for On-chip debugging only.

22.3 Test Access Port – TAP

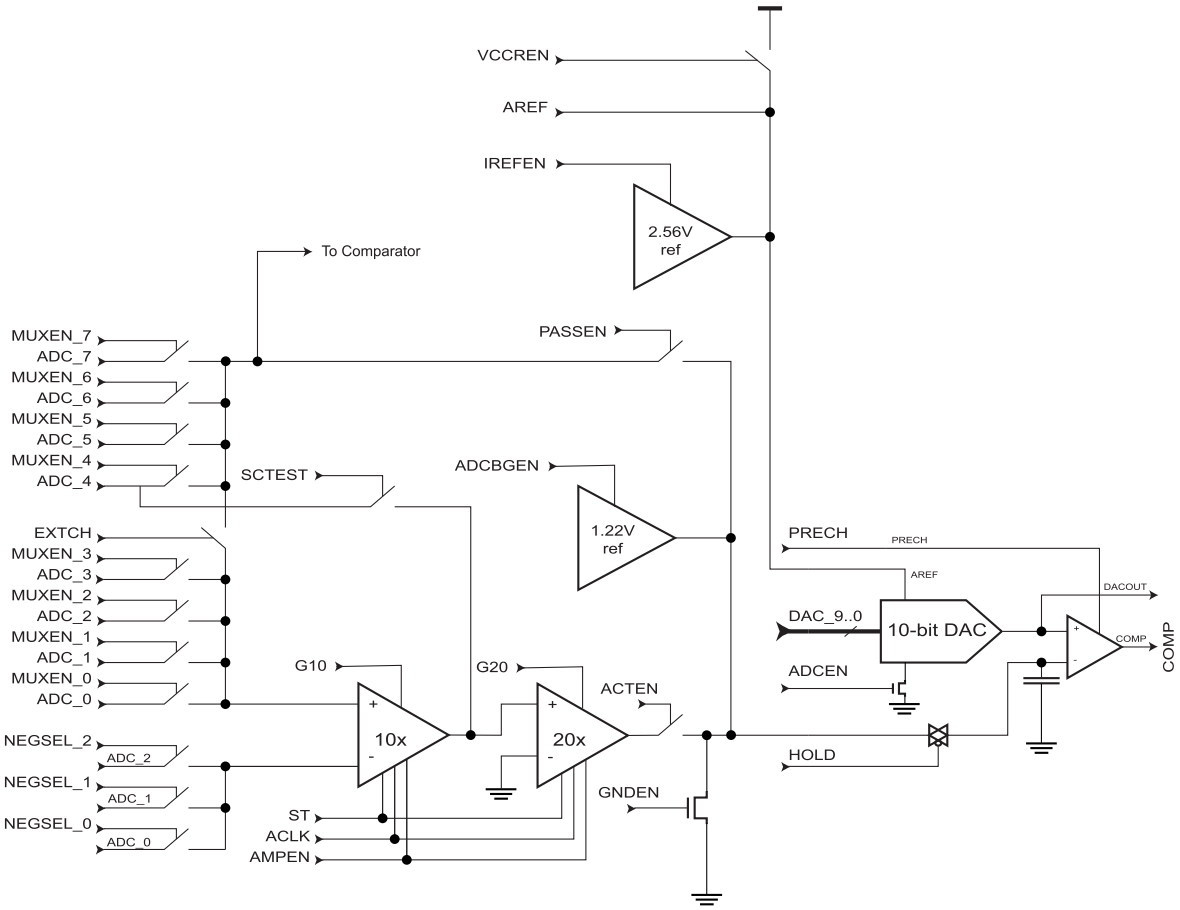
The JTAG interface is accessed through four of the AVR’s pins. In JTAG terminology, these pins constitute the Test Access Port – TAP. These pins are:

- **TMS**: Test mode select. This pin is used for navigating through the TAP-controller state machine.

23.6.6 Scanning the ADC

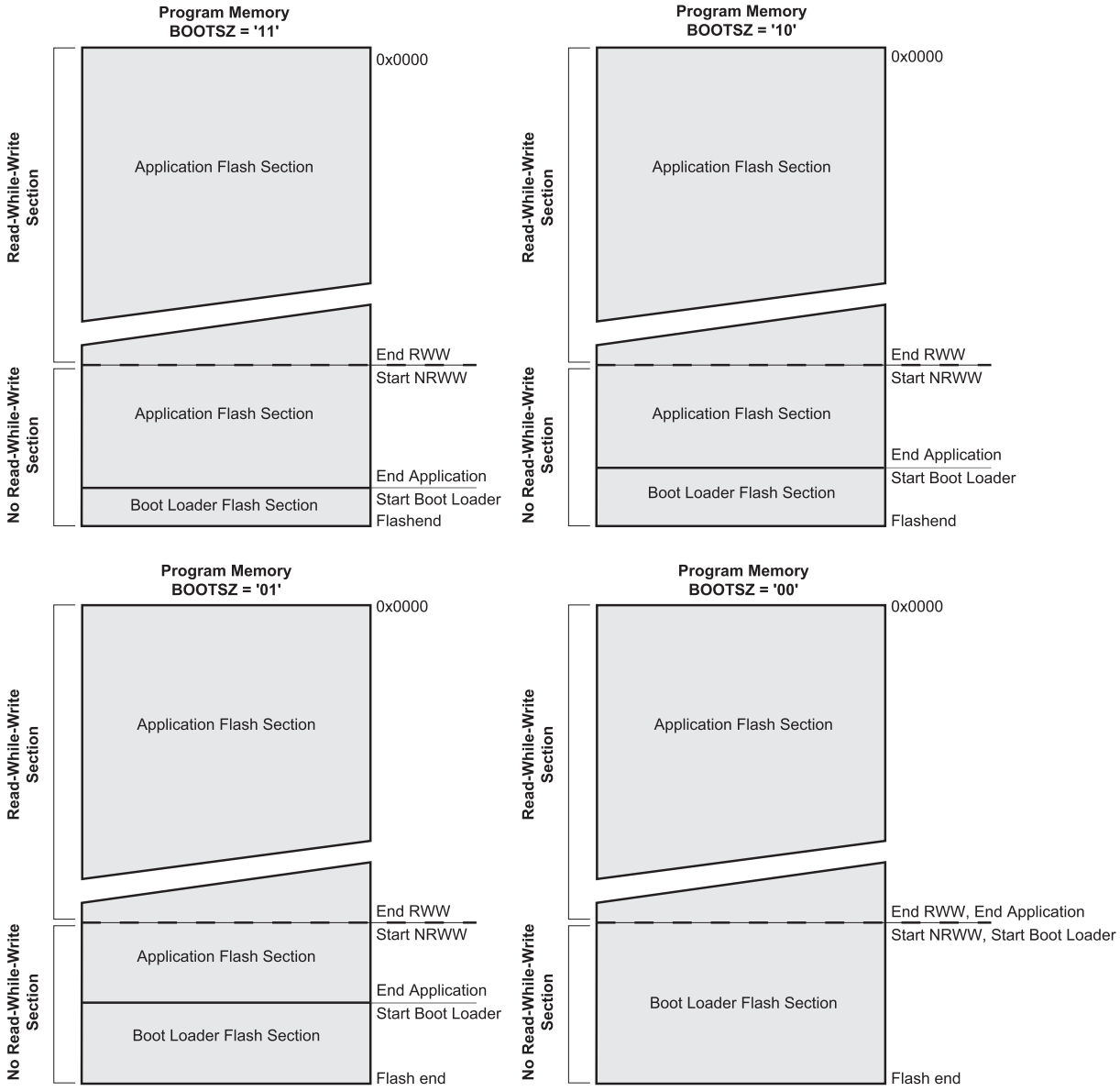
Figure 23-10 shows a block diagram of the ADC with all relevant control and observe signals. The Boundary-scan cell from Figure 23-9 is attached to each of these signals. The ADC need not be used for pure connectivity testing, since all analog inputs are shared with a digital port pin as well.

Figure 23-10. Analog to Digital Converter



The signals are described briefly in Table 23-7.

Figure 24-2. Memory Sections



Note: The parameters in the figure above are given in Table 24-6 on page 333.

24.4 Boot Loader Lock Bits

If no Boot Loader capability is needed, the entire Flash is available for application code. The Boot Loader has two separate sets of Boot Lock bits which can be set independently. This gives the user a unique flexibility to select different levels of protection.

The user can select:

- To protect the entire Flash from a software update by the MCU.
- To protect only the Boot Loader Flash section from a software update by the MCU.
- To protect only the Application Flash section from a software update by the MCU.

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 SPMEN bits in SPMCSR. When an LPM instruction is executed within three cycles after the BLBSET and SPMEN 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 25-5 on page 337 for a detailed description and mapping of the Fuse Low byte.

Bit	7	6	5	4	3	2	1	0
Rd (Z=0x0000)	FLB7	FLB6	FLB5	FLB4	FLB3	FLB2	FLB1	FLB0

Similarly, when reading the Fuse High byte, load 0x0003 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SPMEN bits are set in the SPMCSR, the value of the Fuse High byte (FHB) will be loaded in the destination register as shown below. Refer to Table 25-4 on page 336 for detailed description and mapping of the Fuse High byte.

Bit	7	6	5	4	3	2	1	0
Rd (Z=0x0003)	FHB7	FHB6	FHB5	FHB4	FHB3	FHB2	FHB1	FHB0

When reading the Extended Fuse byte, load 0x0002 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SPMEN bits are set in the SPMCSR, the value of the Extended Fuse byte (EFB) will be loaded in the destination register as shown below. Refer to Table 25-3 on page 336 for detailed description and mapping of the Extended Fuse byte.

Bit	7	6	5	4	3	2	1	0
Rd (Z=0x0002)	–	–	–	–	EFB3	EFB2	EFB1	EFB0

Fuse and Lock bits that are programmed, will be read as zero. Fuse and Lock bits that are unprogrammed, will be read as one.

24.7.10 Preventing Flash Corruption

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

A Flash program corruption can be caused by two situations when the voltage is too low.

- First, a regular write sequence to the Flash requires a minimum voltage to operate correctly.
- Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage for executing instructions is too low.

Flash corruption can easily be avoided by following these design recommendations (one is sufficient):

1. If there is no need for a Boot Loader update in the system, program the Boot Loader Lock bits to prevent any Boot Loader software updates.
2. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.

- Capture-DR: The content of the selected Flash byte is captured into the Flash Data Byte Register. The AVR automatically alternates between reading the low and the high byte for each new Capture-DR state, starting with the low byte for the first Capture-DR encountered after entering the PROG_PAGEREAD command. The Program Counter is post-incremented after reading each high byte, including the first read byte. This ensures that the first data is captured from the first address set up by PROG_COMMANDS, and reading the last location in the page makes the program counter increment into the next page.
- Shift-DR: The Flash Data Byte Register is shifted by the TCK input.

25.9.2 Data Registers

The data registers are selected by the JTAG instruction registers described in section "Programming Specific JTAG Instructions" on page 352. The data registers relevant for programming operations are:

- Reset Register
- Programming Enable Register
- Programming Command Register
- Flash Data Byte Register

25.9.2.1 Reset Register

The Reset Register is a Test Data Register used to reset the part during programming. It is required to reset the part before entering Programming mode.

A high value in the Reset Register corresponds to pulling the external reset low. The part is reset as long as there is a high value present in the Reset Register. Depending on the Fuse settings for the clock options, the part will remain reset for a Reset Time-out period (refer to "Clock Sources" on page 38) after releasing the Reset Register. The output from this data register is not latched, so the reset will take place immediately, as shown in Figure 23-2 on page 301.

25.9.2.2 Programming Enable Register

The Programming Enable Register is a 16-bit register. The contents of this register is compared to the programming enable signature, binary code 0b1010_0011_0111_0000. When the contents of the register is equal to the programming enable signature, programming via the JTAG port is enabled. The register is reset to 0 on Power-on Reset, and should always be reset when leaving Programming mode.

Figure 25-10. Programming Enable Register

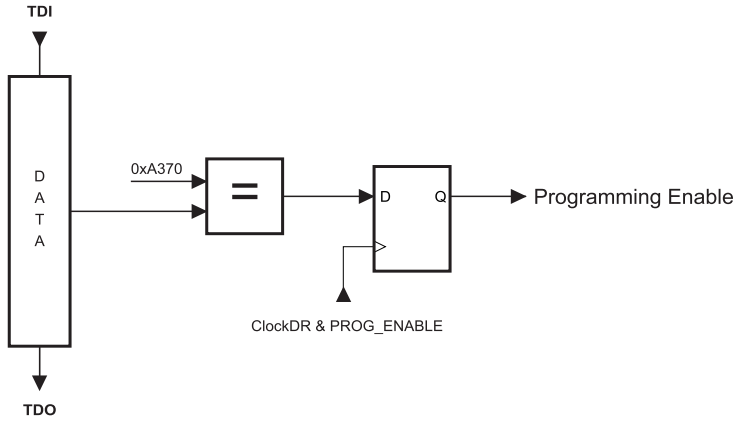


Table 27-1. External Clock Drive

Symbol	Parameter	V_{CC} = 2.7 - 5.5V		V_{CC} = 4.5 - 5.5V		Units
		Min.	Max.	Min.	Max.	
$1/t_{CLCL}$	Oscillator Frequency	0	8	0	16	MHz
t_{CLCL}	Clock Period	125		62.5		ns
t_{CHCX}	High Time	50		25		ns
t_{CLCX}	Low Time	50		25		ns
t_{CLCH}	Rise Time		1.6		0.5	µs
t_{CHCL}	Fall Time		1.6		0.5	µs
Δt_{CLCL}	Change in period from one clock cycle to the next		2		2	%

27.4 Maximum Speed vs. V_{CC}

Maximum frequency is depending on V_{CC}. As shown in Figure 27-2., the Maximum Frequency vs. V_{CC} curve is linear between 1.8V < V_{CC} < 4.5V. To calculate the maximum frequency at a given voltage in this interval, use this equation:

$$Frequency = a \bullet (V - Vx) + Fy$$

To calculate required voltage for a given frequency, use this equation:

$$Voltage = b \bullet (F - Fy) + Vx$$

Table 27-2. Constants used to calculate maximum speed vs. V_{CC}

Voltage and Frequency range	a	b	Vx	Fy
2.7 < VCC < 4.5 or 8 < Frequency < 16	8/1.8	1.8/8	2.7	8

At 3 Volt, this gives: $Frequency = \frac{8}{1.8} \bullet (3 - 2.7) + 8 = 9.33$

Thus, when V_{CC} = 3V, maximum frequency will be 9.33 MHz.

At 8 MHz this gives: $Voltage = \frac{1.8}{8} \bullet (8 - 8) + 2.7 = 2.7$

Thus, a maximum frequency of 8 MHz requires V_{CC} = 2.7V.

Figure 27-2. Maximum Frequency vs. V_{CC}, AT90CAN32/64/128

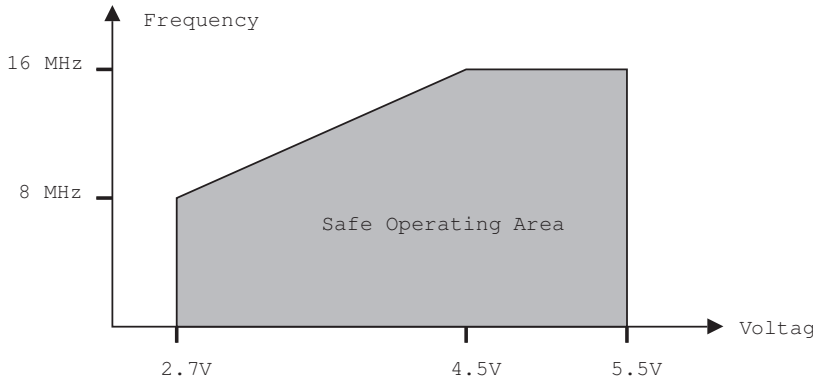


Figure 29-27. BOD Thresholds vs. Temperature (BOD level is 2.7V)

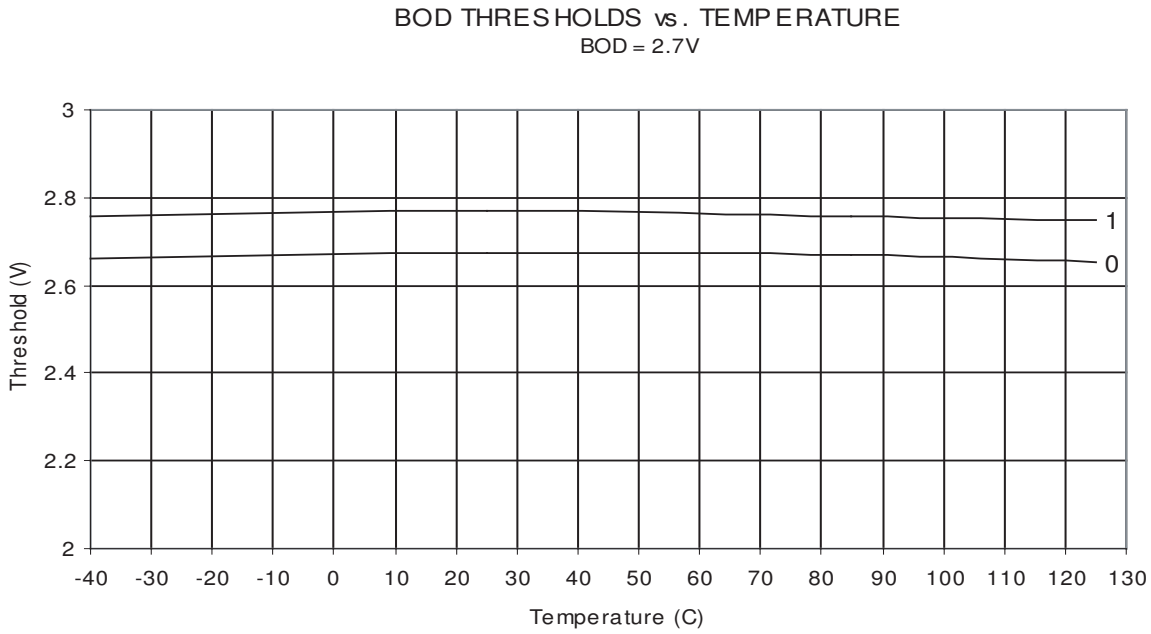


Figure 29-28. Bandgap Voltage vs. Operating Voltage