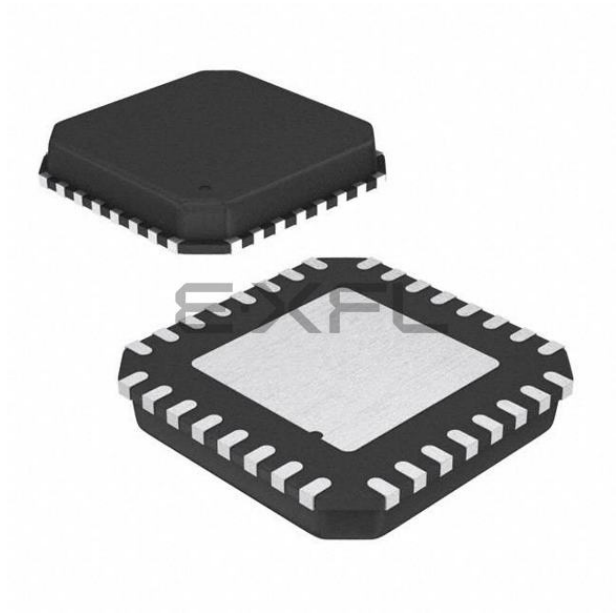


Welcome to E-XFL.COM

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	28
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.7V ~ 5.5V
Data Converters	A/D 28x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN Exposed Pad
Supplier Device Package	32-VQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny828-mur

When using the CLI instruction to disable interrupts, the interrupts will be immediately disabled. No interrupt will be executed after the CLI instruction, even if it occurs simultaneously with the CLI instruction. The following example shows how this can be used to avoid interrupts during the timed EEPROM write sequence.

Assembly Code Example

```
in    r16,  SREG              ; store SREG value
cli                           ; disable interrupts during timed sequence
sbi   EECR, EEMPE             ; start EEPROM write
sbi   EECR, EEPE
out   SREG, r16               ; restore SREG value (I-bit)
```

C Code Example

```
char  cSREG;

cSREG = SREG;           /* store SREG value */
_CLI();                 /* disable interrupts during timed sequence */
EECR |= (1<<EEMPE);     /* start EEPROM write */
EECR |= (1<<EEPE);
SREG = cSREG;           /* restore SREG value (I-bit) */
```

Note: See “Code Examples” on page 7.

When using the SEI instruction to enable interrupts, the instruction following SEI will be executed before any pending interrupts, as shown in the following example.

Assembly Code Example

```
sei               ; set Global Interrupt Enable
sleep             ; enter sleep, waiting for interrupt
                  ; note: will enter sleep before any pending interrupt(s)
```

C Code Example

```
_SEI();           /* set Global Interrupt Enable */
_SLEEP();         /* enter sleep, waiting for interrupt */
                  /* note: will enter sleep before any pending interrupt */
```

Note: See “Code Examples” on page 7.

4.7.1 Interrupt Response Time

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles the Program Vector address for the actual interrupt handling routine is executed. During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.

To ensure stable operation of the MCU it is required to avoid sudden changes in the external clock frequency . A variation in frequency of more than 2% from one clock cycle to the next can lead to unpredictable behavior. It is required to ensure that the MCU is kept in Reset during such changes in the clock frequency.

Stable operation for large step changes in system clock frequency is guaranteed when using the system clock prescaler. See “System Clock Prescaler” on page 30.

6.2.2 Calibrated Internal 8MHz Oscillator

The internal 8MHz oscillator operates with no external components and, by default, provides a clock source with an approximate frequency of 8MHz. Though voltage and temperature dependent, this clock can be very accurately calibrated by the user. See Table 104 on page 249 and “Internal Oscillator Speed” on page 293 for more details.

During reset, hardware loads the pre-programmed calibration value into the OSCCAL0 register and thereby automatically calibrates the oscillator. The accuracy of this calibration is referred to as “Factory Calibration” in Table 104 on page 249. For more information on automatic loading of pre-programmed calibration value, see section “Calibration Bytes” on page 229.

It is possible to reach higher accuracies than factory defaults, especially when the application allows temperature and voltage ranges to be narrowed. The firmware can reprogram the calibration data in OSCCAL0 either at start-up or during run-time. The continuous, run-time calibration method allows firmware to monitor voltage and temperature and compensate for any detected variations. See “OSCCAL0 – Oscillator Calibration Register” on page 32, “Temperature Measurement” on page 148, and Table 52 on page 150. The accuracy of this calibration is referred to as “User Calibration” in Table 104 on page 249.

The oscillator temperature calibration registers, OSCTCAL0A and OSCTCAL0B, can be used for one-time temperature calibration of oscillator frequency. See “OSCTCAL0A – Oscillator Temperature Calibration Register A” on page 33 and “OSCTCAL0B – Oscillator Temperature Calibration Register B” on page 33.

When this oscillator is used as the chip clock, it will still be used for the Watchdog Timer and for the Reset Time-out.

Start-up time for this clock source is determined by the SUT fuse bit, as shown in Table 7 on page 30.

6.2.3 Internal 32kHz Ultra Low Power (ULP) Oscillator

The internal 32kHz oscillator is a low power oscillator that operates with no external components. It provides a clock source with an approximate frequency of 32kHz. The frequency depends on supply voltage, temperature and batch variations. See Table 105 on page 250 for accuracy details.

During reset, hardware loads the pre-programmed calibration value into the OSCCAL1 register and thereby automatically calibrates the oscillator. The accuracy of this calibration is referred to as “Factory Calibration” in Table 105 on page 250. For more information on automatic loading of pre-programmed calibration value, see section “Calibration Bytes” on page 229.

Start-up time for this clock source is determined by the SUT fuse bit, as shown in Table 7 on page 30.

6.2.4 Default Clock Settings

The device is shipped with following fuse settings:

- Calibrated Internal 8MHz Oscillator (see CKSEL fuse bits in Table 6 on page 28)
- Longest possible start-up time (see SUT fuse bits in Table 7 on page 30)
- System clock prescaler set to 8 (see CKDIV8 fuse bit on page 32)

The default setting gives a 1MHz system clock and ensures all users can make their desired clock source setting using an in-system or high-voltage programmer.

Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 3 – PRTIM1: Power Reduction Timer/Counter1**

Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 2 – PRSPI: Power Reduction SPI**

Writing a logic one to this bit shuts down the SPI by stopping the clock to the module. When waking up the SPI again, the SPI should be re-initialized to ensure proper operation.

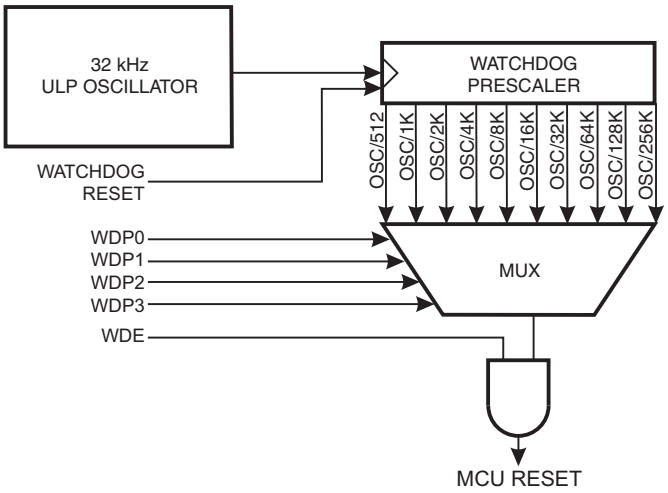
- **Bit 1 – PRUSART0: Power Reduction USART0**

Writing a logic one to this bit shuts down the USART0 module. When the USART0 is enabled, operation will continue like before the shutdown.

- **Bit 0 – PRADC: Power Reduction ADC**

Writing a logic one to this bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot be used when the ADC is shut down.

Figure 19. Watchdog Timer



8.4.1 Timed Sequences for Changing the Configuration of the Watchdog Timer

The sequence for changing configuration differs slightly between the two safety levels. Separate procedures are described for each level.

- Safety Level 1

  In this mode, the Watchdog Timer is initially disabled, but can be enabled by writing the WDE bit to one without any restriction. A timed sequence is needed when disabling an enabled Watchdog Timer. To disable an enabled Watchdog Timer, the following procedure must be followed:

  1. Write the signature for change enable of protected I/O registers to register CCP
  2. Within four instruction cycles, in the same operation, write WDE and WDP bits

- Safety Level 2

  In this mode, the Watchdog Timer is always enabled, and the WDE bit will always read as one. A timed sequence is needed when changing the Watchdog Time-out period. To change the Watchdog Time-out, the following procedure must be followed:

  1. Write the signature for change enable of protected I/O registers to register CCP
  2. Within four instruction cycles, write the WDP bit. The value written to WDE is irrelevant

8.4.2 Code Examples

The following code example shows how to turn off the WDT. The example assumes that interrupts are controlled (e.g., by disabling interrupts globally) so that no interrupts will occur during execution of these functions.

Note: 1. When the IVSEL bit in MCUCR is set, interrupt vectors are moved to the start of the Flash boot section. In this case, the address of each interrupt vector will be the address in this table added to the start address of the Flash boot section.

2. When the BOOTRST fuse is programmed, the device will jump to the boot loader address at reset. See “Entering the Boot Loader Program” on page 216.

In case the program never enables an interrupt source, the interrupt vectors will not be used and, consequently, regular program code can be placed at these locations. This is also the case if the reset vector is in the application section while the interrupt vectors are in the boot section, or vice versa.

A typical and general setup for interrupt vector addresses in ATtiny828 is shown in the program example below.

Assembly Code Example

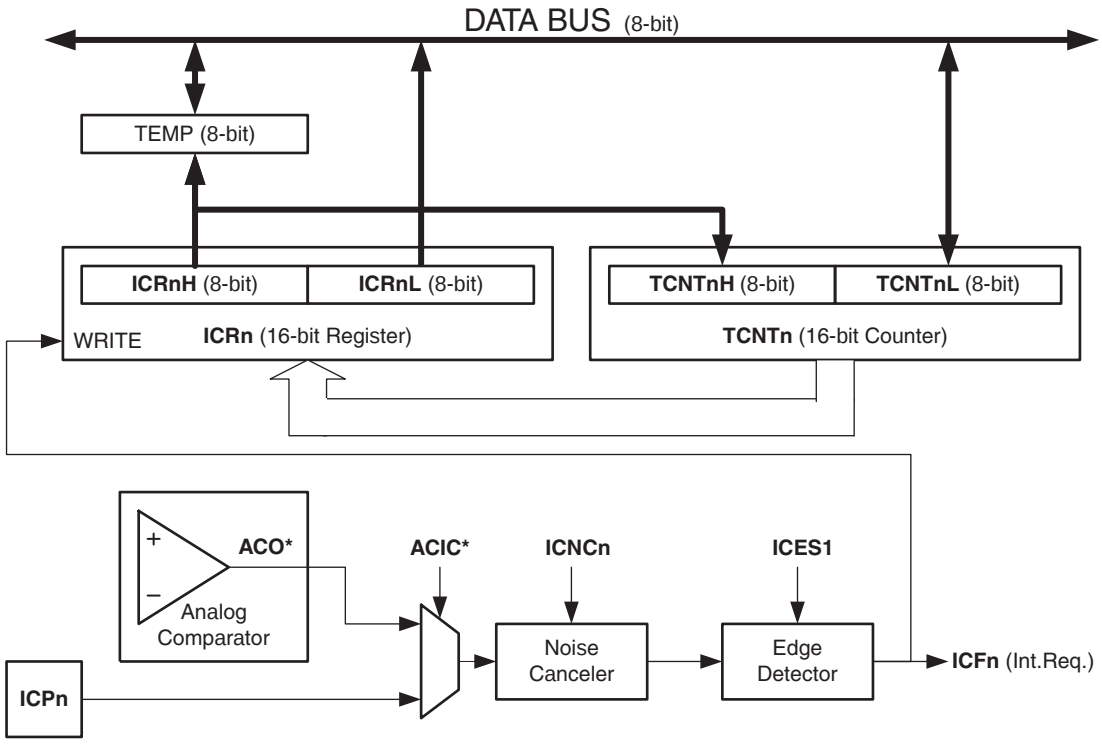
```
      .org 0x0000                                ; Set address of next
      statement

            rjmp              RESET                ; Address 0x0000
            rjmp              INT0_ISR             ; Address 0x0001
            rjmp              INT1_ISR             ; Address 0x0002
            rjmp              PCINT0_ISR           ; Address 0x0003
            rjmp              PCINT1_ISR           ; Address 0x0004
            rjmp              PCINT2_ISR           ; Address 0x0005
            rjmp              PCINT3_ISR           ; Address 0x0006
            rjmp              WDT_ISR              ; Address 0x0007
            rjmp              TIM1_CAPT_ISR        ; Address 0x0008
            rjmp              TIM1_COMPA_ISR       ; Address 0x0009
            rjmp              TIM1_COMPB_ISR       ; Address 0x000A
            rjmp              TIM1_OVF_ISR         ; Address 0x000B
            rjmp              TIM0_COMPA_ISR       ; Address 0x000C
            rjmp              TIM0_COMPB_ISR       ; Address 0x000D
            rjmp              TIM0_OVF_ISR         ; Address 0x000E
            rjmp              SPI_ISR              ; Address 0x000F
            rjmp              USART0_RXS_ISR       ; Address 0x0010
            rjmp              USART0_RXC_ISR       ; Address 0x0011
            rjmp              USART0_DRE_ISR       ; Address 0x0012
            rjmp              USART0_TXC_ISR       ; Address 0x0013
            rjmp              ADC_ISR              ; Address 0x0014
            rjmp              EE_RDY_ISR           ; Address 0x0015
            rjmp              ANA_COMP_ISR         ; Address 0x0016
            rjmp              TWI_ISR              ; Address 0x0017
            rjmp              SPM_RDY_ISR          ; Address 0x0018
            rjmp              RESERVED             ; Address 0x0019

      RESET:                                     ; Main program start
            <instr>                              ; Address 0x001A
            ...
```

Note: See “Code Examples” on page 7.

Table 17 shows reset and interrupt vector placement for combinations of BOOTRST and IVSEL settings.

or alternatively, via the analog-comparator unit. The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied. Alternatively the time-stamps can be used for creating a log of the events.

Figure 39. Input Capture Unit Block Diagram



The Input Capture unit is illustrated by the block diagram shown in Figure 39. The elements of the block diagram that are not directly a part of the Input Capture unit are gray shaded. The small “n” in register and bit names indicates the Timer/Counter number.

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

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

The ICR1 Register can only be written when using a Waveform Generation mode that utilizes the ICR1 Register for defining the counter’s TOP value. In these cases the Waveform Generation mode (WGM1[3:0]) bits must be set before the TOP value can be written to the ICR1 Register. When writing the ICR1 Register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

For more information on how to access the 16-bit registers refer to “Accessing 16-bit Registers” on page 120.

12.5.1 Input Capture Trigger Source

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

The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O port control registers (DDR and PORT) that are affected by the COM1x[1:0] bits are shown. When referring to the OC1x state, the reference is for the internal OC1x Register, not the OC1x pin. If a system reset occur, the OC1x Register is reset to “0”.

The general I/O port function is overridden by the Output Compare (OC1x) from the Waveform Generator if either of the COM1x[1:0] bits are set. However, the OC1x pin direction (input or output) is still controlled by the *Data Direction Register* (DDR) for the port pin. The Data Direction Register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the Waveform Generation mode, but there are some exceptions. See Table 39 on page 124, Table 40 on page 124 and Table 41 on page 124 for details.

The design of the Output Compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x[1:0] bit settings are reserved for certain modes of operation. See “Register Description” on page 123

The COM1x[1:0] bits have no effect on the Input Capture unit.

12.7.1 Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM1x[1:0] bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x[1:0] = 0 tells the Waveform Generator that no action on the OC1x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 39 on page 124. For fast PWM mode refer to Table 40 on page 124, and for phase correct and phase and frequency correct PWM refer to Table 41 on page 124.

A change of the COM1x[1:0] bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC1x strobe bits.

12.8 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the Waveform Generation mode (WGM1[3:0]) and Compare Output mode (COM1x[1:0]) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM1x[1:0] bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM1x[1:0] bits control whether the output should be set, cleared or toggle at a compare match (“Compare Match Output Unit” on page 110)

For detailed timing information refer to “Timer/Counter Timing Diagrams” on page 118.

12.8.1 Normal Mode

The simplest mode of operation is the Normal mode (WGM1[3:0] = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 16-bit value (MAX = 0xFFFF) and then restarts from the BOTTOM (0x0000). In normal operation the Timer/Counter Overflow Flag (TOV1) will be set in the same timer clock cycle as the TCNT1 becomes zero. The TOV1 flag in this case behaves like a 17th bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV1 flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

The Input Capture unit is easy to use in Normal mode. However, observe that the maximum interval between the external events must not exceed the resolution of the counter. If the interval between events are too long, the timer overflow interrupt or the prescaler must be used to extend the resolution for the capture unit.

The Output Compare units can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

12.8.2 Clear Timer on Compare Match (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM1[3:0] = 4 or 12), the OCR1A or ICR1 Register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT1) matches either the OCR1A (WGM1[3:0] = 4) or the ICR1 (WGM1[3:0] = 12). The OCR1A or ICR1 define the top value for the counter, hence

12.11.5 TOCPMCOE – Timer/Counter Output Compare Pin Mux Channel Output Enable

Bit	7	6	5	4	3	2	1	0	
(0xE2)	TOCC7OE	TOCC6OE	TOCC5OE	TOCC4OE	TOCC3OE	TOCC2OE	TOCC1OE	TOCC0OE	TOCPMCOE
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:0 – TOCCnOE: Timer/Counter Output Compare Channel Output Enable**

These bits enable the selected output compare channel on the corresponding TOCCn pin, regardless if the output compare mode is selected, or not.

12.11.6 TCNT1H and TCNT1L – Timer/Counter1

Bit	7	6	5	4	3	2	1	0	
(0x85)	TCNT1[15:8]								TCNT1H
(0x84)	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The two Timer/Counter I/O locations (TCNT1H and TCNT1L, combined TCNT1) give direct access, both for read and for write operations, to the Timer/Counter unit 16-bit counter. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary high byte register (TEMP). This temporary register is shared by all the other 16-bit registers. See “Accessing 16-bit Registers” on page 120.

Modifying the counter (TCNT1) while the counter is running introduces a risk of missing a compare match between TCNT1 and one of the OCR1x Registers.

Writing to the TCNT1 Register blocks (removes) the compare match on the following timer clock for all compare units.

12.11.7 OCR1AH and OCR1AL – Output Compare Register 1 A

Bit	7	6	5	4	3	2	1	0	
(0x89)	OCR1A[15:8]								OCR1AH
(0x88)	OCR1A[7:0]								OCR1AL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

12.11.8 OCR1BH and OCR1BL – Output Compare Register 1 B

Bit	7	6	5	4	3	2	1	0	
(0x8B)	OCR1B[15:8]								OCR1BH
(0x8A)	OCR1B[7:0]								OCR1BL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Registers contain a 16-bit value that is continuously compared with the counter value (TCNT1). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC1x pin.

16. SPI – Serial Peripheral Interface

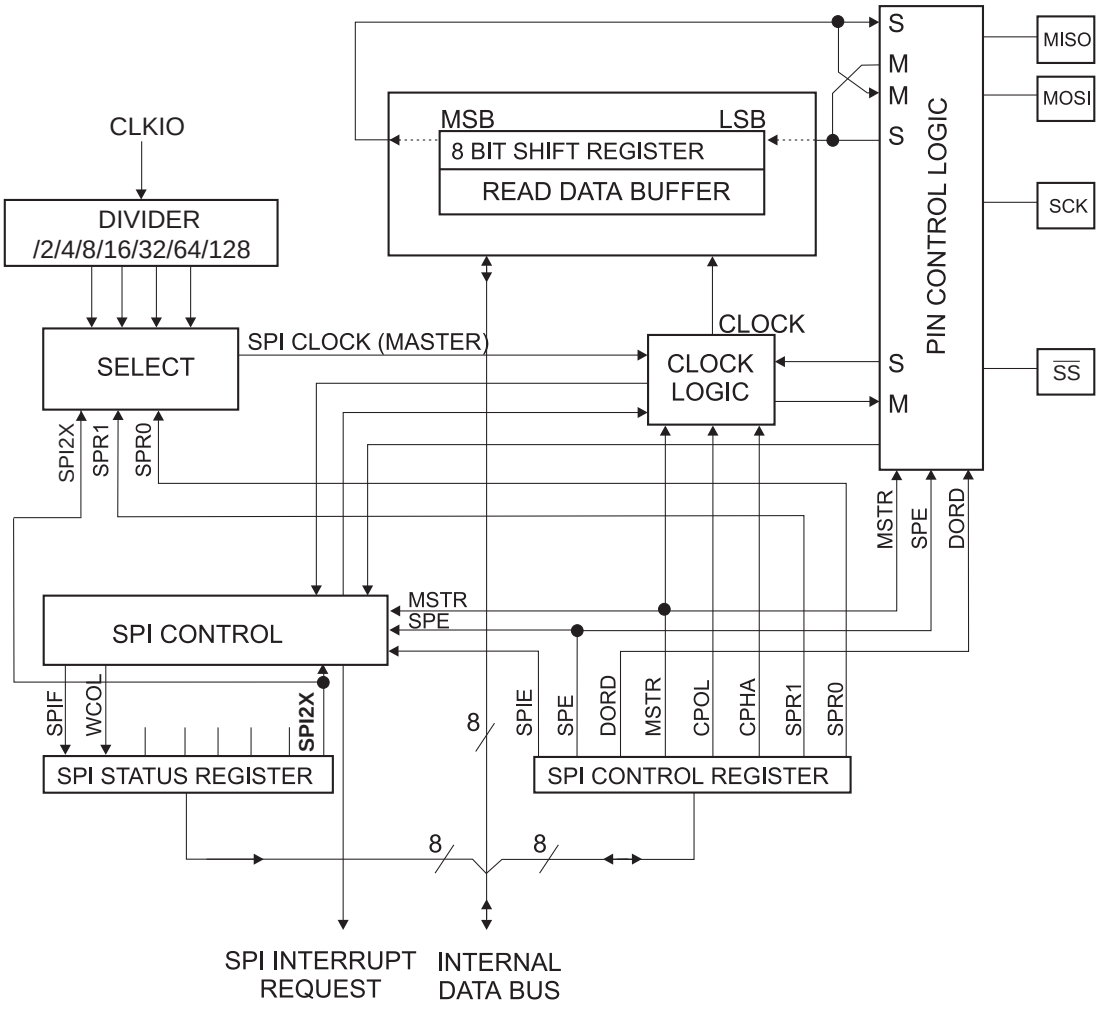
16.1 Features

- Full-duplex, Three-wire Synchronous Data Transfer
- Master or Slave Operation
- LSB First or MSB First Data Transfer
- Seven Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wake-up from Idle Mode
- Double Speed (CK/2) Master SPI Mode

16.2 Overview

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between ATtiny828 and peripheral devices, or between several AVR devices. The SPI module is illustrated in Figure 66.

Figure 66. SPI Block Diagram



Note: For SPI pin placement, see Figure 1 on page 2, and Table 55 on page 157.

17.7.2 Receiving Frames with 9 Data Bits

If 9-bit characters are used (UCSZ = 7) the ninth bit must be read from the RXB8 bit before reading the low bits from UDR. This rule applies to the FE, DOR and UPE status flags, as well. Status bits must be read before data from UDR, since reading UDR will change the state of the receive buffer FIFO and, consequently, state of TXB8, FE, DOR and UPE bits.

The following code example shows a simple USART receive function that handles both nine bit characters and the status bits.

Assembly Code Example[1]

```
USART_Receive:
      ; Wait for data to be received
      sbis              UCSRA, RXC
      rjmp              USART_Receive

      ; Get status and 9th bit, then data from buffer
      in                r18, UCSRA
      in                r17, UCSRB
      in                r16, UDR

      ; If error, return -1
      andi              r18,(1<<FE)|(1<<DOR)|(1<<UPE)
      breq              USART_ReceiveNoError
      ldi               r17, HIGH(-1)
      ldi               r16, LOW(-1)

USART_ReceiveNoError:
      ; Filter the 9th bit, then return
      lsr               r17
      andi              r17, 0x01
      ret
```

- **Bit 1 – UCPHA: Clock Phase**

The UCPHA bit setting determine if data is sampled on the leasing edge (first) or tailing (last) edge of XCK. See “SPI Data Modes and Timing” on page 191 for details.

- **Bit 0 – UCPOL: Clock Polarity**

The UCPOL bit sets the polarity of the XCK clock. The combination of the UCPOL and UCPHA bit settings determine the timing of the data transfer. See “SPI Data Modes and Timing” on page 191 for details.

18.8.5 UBRRL and UBRRH – USART MSPIM Baud Rate Registers

The function and bit description of the baud rate registers in MSPI mode is identical to normal USART operation. See “UBRRL and UBRRH – USART Baud Rate Registers” on page 189.

21.10 Programming Time for Flash when Using SPM

Flash access is timed using the internal, calibrated 8MHz oscillator. Typical Flash programming times for the CPU are shown in Table 84.

Table 84. SPM Programming Time

Operation	Min (1)	Max (1)
SPM: Flash Page Erase, Flash Page Write, and lock bit write	3.7 ms	4.5 ms

Note: 1. Min and max programming times are per individual operation.

21.11 Register Description

21.11.1 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	RSIG	RWWSRE	RWFLB	PGWRT	PGERS	SPMEN	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 SPMEN bit is cleared.

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

When this bit is set, the RWW section cannot be accessed.

This bit is set when a self-programming operation (Page Erase or Page Write) to the RWW section is initiated.

This bit is cleared if the RWWSRE bit is written to one after a self-programming operation is completed. This bit is automatically cleared when a page load operation is initiated.

- **Bit 5 – RSIG: Read Device Signature Imprint Table**

Issuing an LPM instruction within three cycles after RSIG and SPMEN bits have been set in SPMCSR will return the selected data (depending on Z-pointer value) from the device signature imprint table into the destination register. See “Device Signature Imprint Table” on page 228 for details.

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

The RWW section is blocked for reading (see RWWSB bit) when the section is being programmed. To re-enable the section, the software must first wait until the programming is completed (see SPMEN bit). The RWW section is then re-enabled by simultaneously writing bits RWWSRE and SPMEN and, within four clock cycles, issuing an SPM instruction.

The RWW section cannot be re-enabled while the Flash is busy with a Page Erase or a Page Write operation (see SPMEN). If the RWWSRE bit is written while the Flash is being loaded, the operation will abort and the data will be lost.

5. Wait until V_{CC} actually reaches 4.5 – 5.5V before giving any parallel programming commands.
6. Exit programming mode by powering the device down or by bringing $\overline{\text{RESET}}$ pin to 0V.

23.2.2 Considerations for Efficient Programming

Loaded commands and addresses are retained in the device during programming. For efficient programming, the following should be considered.

- When writing or reading multiple memory locations, the command needs only be loaded once
- Do not write the data value 0xFF, since this already is the contents of the entire Flash and EEPROM (unless the EESAVE Fuse is programmed) after a Chip Erase
- Address high byte needs only be loaded before programming or reading a new 256 word window in Flash or 256 byte EEPROM. This also applies to reading signature bytes

23.2.3 Chip Erase

A Chip Erase must be performed before the Flash and/or EEPROM are reprogrammed. The Chip Erase command will erase all Flash and EEPROM plus lock bits. If the EESAVE fuse is programmed, the EEPROM is not erased.

Lock bits are not reset until the program memory has been completely erased. Fuse bits are not changed.

The Chip Erase command is loaded as follows:

1. Set XA1, XA0 to “10”. This enables command loading
2. Set BS1 to “0”
3. Set DATA to “1000 0000”. This is the command for Chip Erase
4. Give CLKI a positive pulse. This loads the command
5. Give $\overline{\text{WR}}$ a negative pulse. This starts the Chip Erase. RDY/$\overline{\text{BSY}}$ goes low
6. Wait until RDY/$\overline{\text{BSY}}$ goes high before loading a new command

23.2.4 Programming the Flash

Flash is organized in pages, as shown in Table 94 on page 232. When programming the Flash, the program data is first latched into a page buffer. This allows one page of program data to be programmed simultaneously. The following procedure describes how to program the entire Flash memory:

A. Load Command “Write Flash”

1. Set XA1, XA0 to “10”. This enables command loading.
2. Set BS1 to “0”.
3. Set DATA to “0001 0000”. This is the command for Write Flash.
4. Give CLKI a positive pulse. This loads the command.

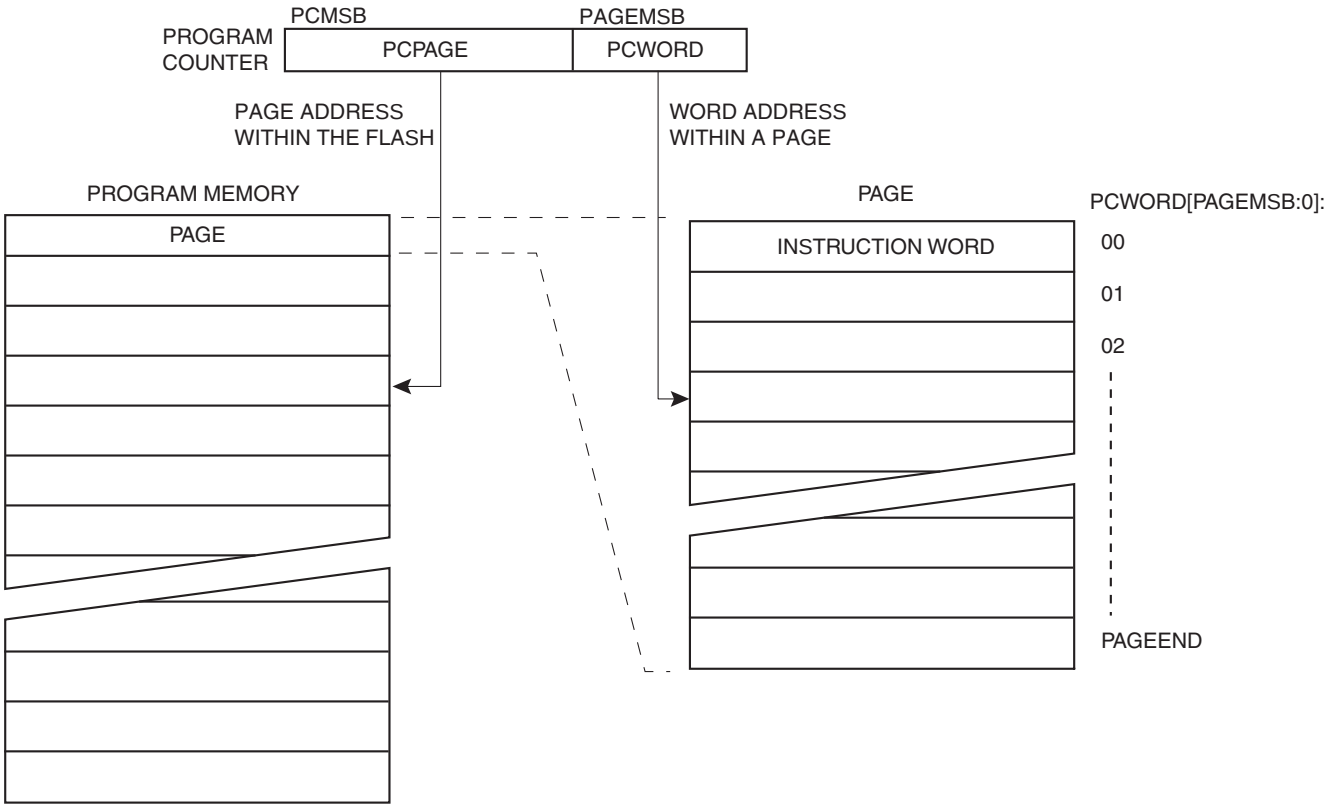
B. Load Address Low byte

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

C. Load Data Low Byte

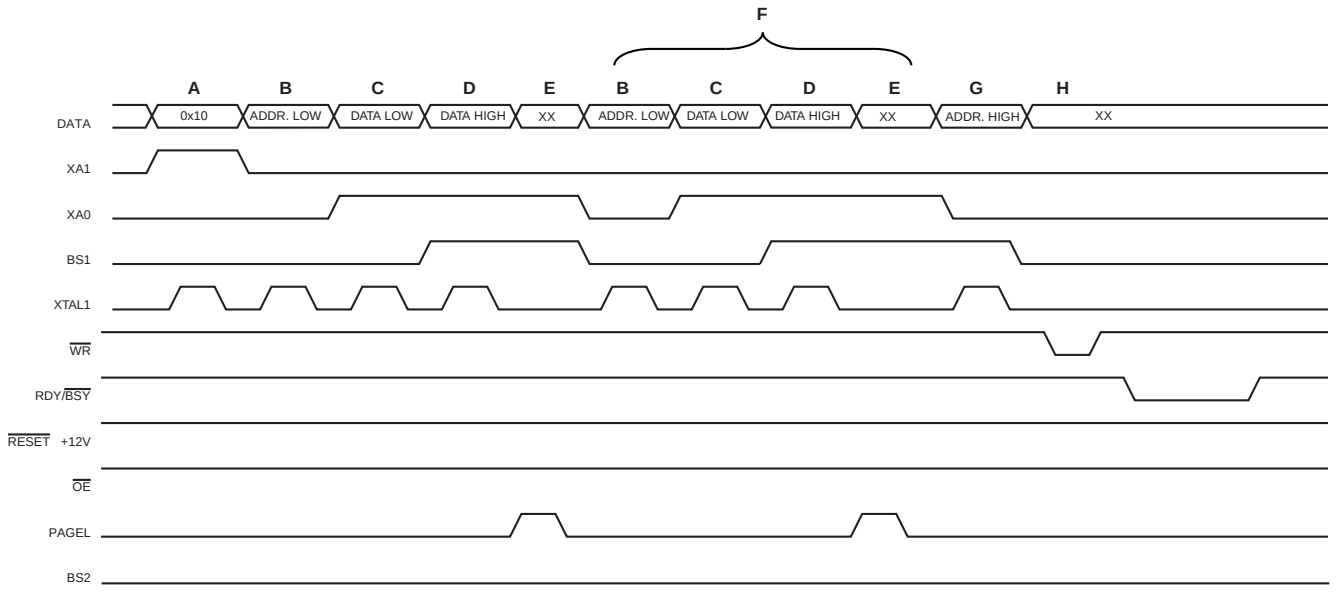
1. Set XA1, XA0 to “01”. This enables data loading.
2. Set DATA = Data low byte (0x00 – 0xFF).

Figure 94. Addressing the Flash Which is Organized in Pages



Flash programming waveforms are illustrated in Figure 95, where XX means “don’t care” and letters refer to the programming steps described earlier.

Figure 95. Flash Programming Waveforms



Symbol	Parameter	Condition	Min	Typ [1]	Max	Units
V_{OH}	Output High-voltage[4] Except RESET pin[6]	V_{CC} = 5V, I_{OH} = -10 mA [5]	4.3			V
		V_{CC} = 3V, I_{OH} = -5 mA [5]	2.5			
		V_{CC} = 1.8V, I_{OH} = -2 mA [5]	1.4			
I_{LIL}	Input Leakage Current, I/O Pin (absolute value)	V_{CC} = 5.5V, pin low		<0.05	1	µA
I_{LIH}	Input Leakage Current, I/O Pin (absolute value)	V_{CC} = 5.5V, pin high		<0.05	1	µA
I_{LIAC}	Input Leakage Current, Analog Comparator	V_{CC} = 5V V_{IN} = V_{CC}/2	-50		50	nA
R_{RST}	Reset Pull-up Resistor	V_{CC} = 5.5V, input low	30		60	kΩ
R_{PU}	I/O Pin Pull-up Resistor	V_{CC} = 5.5V, input low	20		50	kΩ
I_{CC}	Power Supply Current[10]	Active 1 MHz, V_{CC} = 2V		0.2	0.4	mA
		Active 4 MHz, V_{CC} = 3V		1.2	2	mA
		Active 8 MHz, V_{CC} = 5V		3.9	5	mA
		Idle 1 MHz, V_{CC} = 2V		0.03	0.1	mA
		Idle 4 MHz, V_{CC} = 3V		0.2	0.4	mA
		Idle 8 MHz, V_{CC} = 5V		0.9	1.5	mA
	Power-down mode[11]	WDT enabled, V_{CC} = 3V		1.8	4	µA
		WDT disabled, V_{CC} = 3V		0.1	2	µA

Notes:
1. Typical values at 25°C.
2. “Min” means the lowest value where the pin is guaranteed to be read as high.
3. “Max” means the highest value where the pin is guaranteed to be read as low.
4. Under steady-state (non-transient) conditions I/O ports can sink/source more current than the test conditions, however, the sum current of PORTA and PORTB mustn’t exceed 100mA. Also, the sum current of PORTC and PORTD mustn’t exceed 120mA. V_{OL}/V_{OH} is not guaranteed to meet specifications if pin or port currents exceed the limits given.
5. Pins are not guaranteed to sink/source currents greater than those listed at the given supply voltage.
6. The RESET pin must tolerate high voltages when entering and operating in programming modes and, as a consequence, has a weak drive strength as compared to regular I/O pins. See “Reset Pin as I/O” on page 279, and “Reset Pin as I/O” on page 285.
7. Ports with standard sink strength: PORTD0, PORTD3.
8. Ports with high sink strength: PORTA[7:0], PORTB[7:0], PORTC[7:0], PORTD1.
9. Ports with extra high strength: PORTC[7:0]. See “PHDE – Port High Drive Enable Register” on page 81.
10. Results obtained using external clock and methods described in “Minimizing Power Consumption” on page 35. Power reduction fully enabled (PRR = 0xFF) and with no I/O drive.
11. BOD Disabled.

Table 105. Calibration Accuracy of Internal 32kHz Oscillator

Calibration Method	Target Frequency	V_{CC}	Temperature	Accuracy
Factory Calibration	32kHz	1.7 – 5.5V	-40°C to +85°C	±30%

24.4.3 External Clock Drive

Figure 102. External Clock Drive Waveform

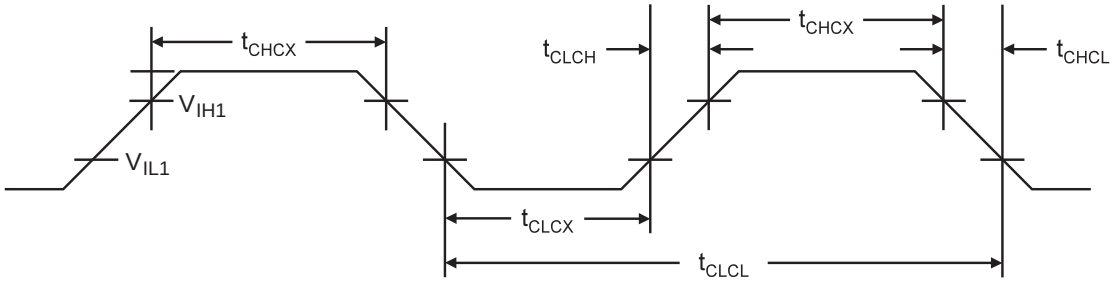


Table 106. External Clock Drive Characteristics

Symbol	Parameter	V_{CC} = 1.7 – 5.5V		V_{CC} = 2.7–5.5V		V_{CC} = 4.5–5.5V		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
$1/t_{CLCL}$	Clock Frequency	0	4	0	8	0	12	MHz
t_{CLCL}	Clock Period	250		125		83		ns
t_{CHCX}	High Time	100		40		20		ns
t_{CLCX}	Low Time	100		40		20		ns
t_{CLCH}	Rise Time		2.0		1.6		0.5	µs
t_{CHCL}	Fall Time		2.0		1.6		0.5	µs
Δt_{CLCL}	Period change from one clock cycle to next		2		2		2	%

24.5 System and Reset Characteristics

Table 107. Reset and Internal Voltage Characteristics

Symbol	Parameter	Condition	Min(1)	Typ(1)	Max(1)	Units
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		0.2 V_{CC}		0.9V_{CC}	V
V_{BG}	Internal bandgap voltage	V_{CC} = 2.7V T_A = 25°C	1.0	1.1	1.2	V
t_{RST}	Minimum pulse width on $\overline{\text{RESET}}$ Pin	V_{CC} = 1.8V V_{CC} = 3V V_{CC} = 5V		2000 700 400		ns

Note: 1. Values are guidelines, only

24.9 Analog Comparator Characteristics

Table 113. Analog Comparator Characteristics, T = -40°C to +85°C

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{AIO}	Input Offset Voltage	V_{CC} = 5V, VIN = V_{CC} / 2		< 10	40	mV
I_{LAC}	Input Leakage Current	V_{CC} = 5V, VIN = V_{CC} / 2	-50		50	nA
t_{APD}	Analog Propagation Delay (from saturation to slight overdrive)	V_{CC} = 2.7V		750		ns
		V_{CC} = 4.0V		500		
	Analog Propagation Delay (large step change)	V_{CC} = 2.7V		100		
		V_{CC} = 4.0V		75		
t_{DPD}	Digital Propagation Delay	V_{CC} = 1.7V – 5.5		1	2	CLK

24.10 Parallel Programming Characteristics

Figure 104. Parallel Programming Timing, Including some General Timing Requirements

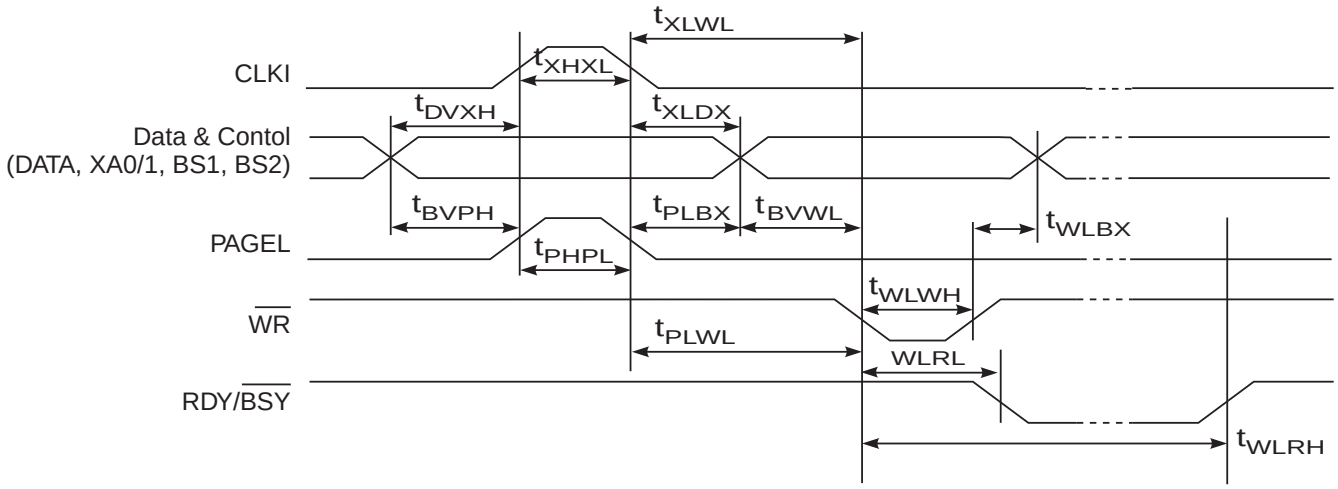


Figure 130. I/O Pin Pull-up Resistor Current vs. input Voltage (V_{CC} = 2.7V)

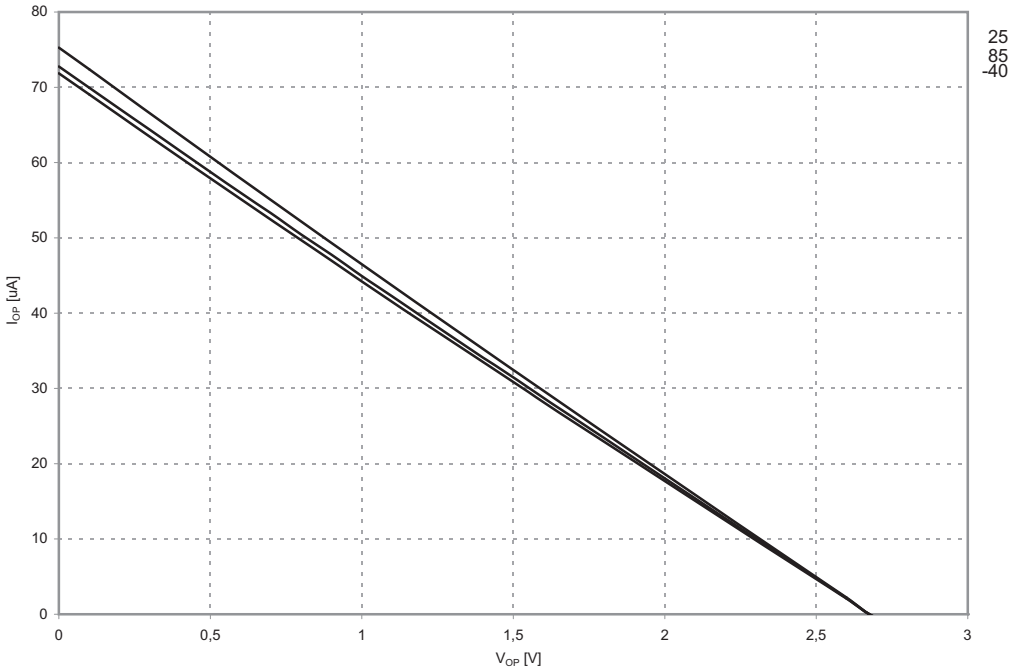


Figure 131. I/O pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5V)

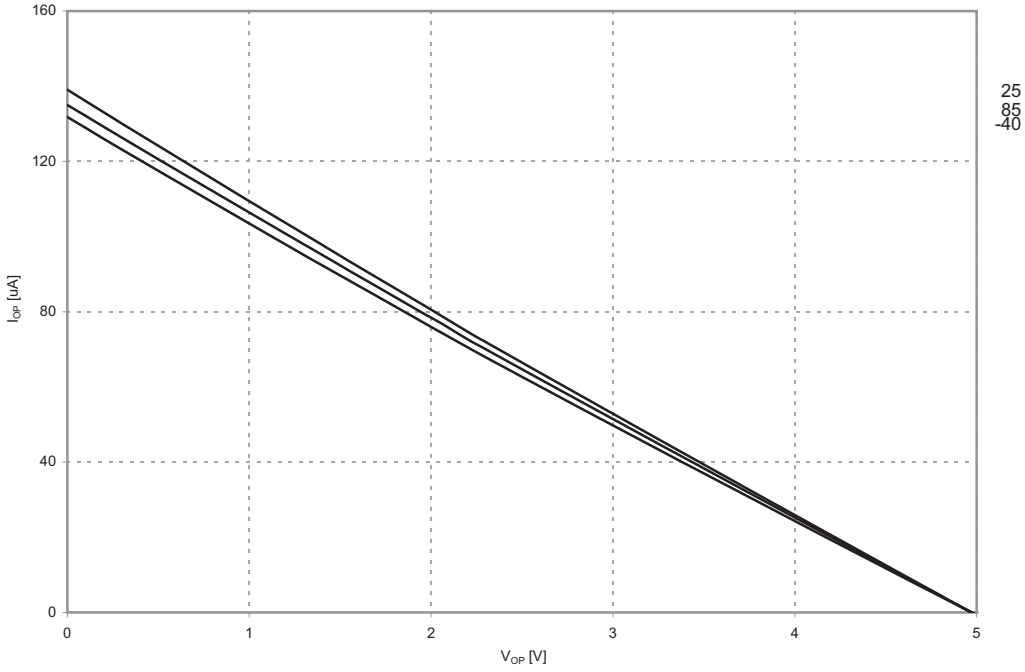
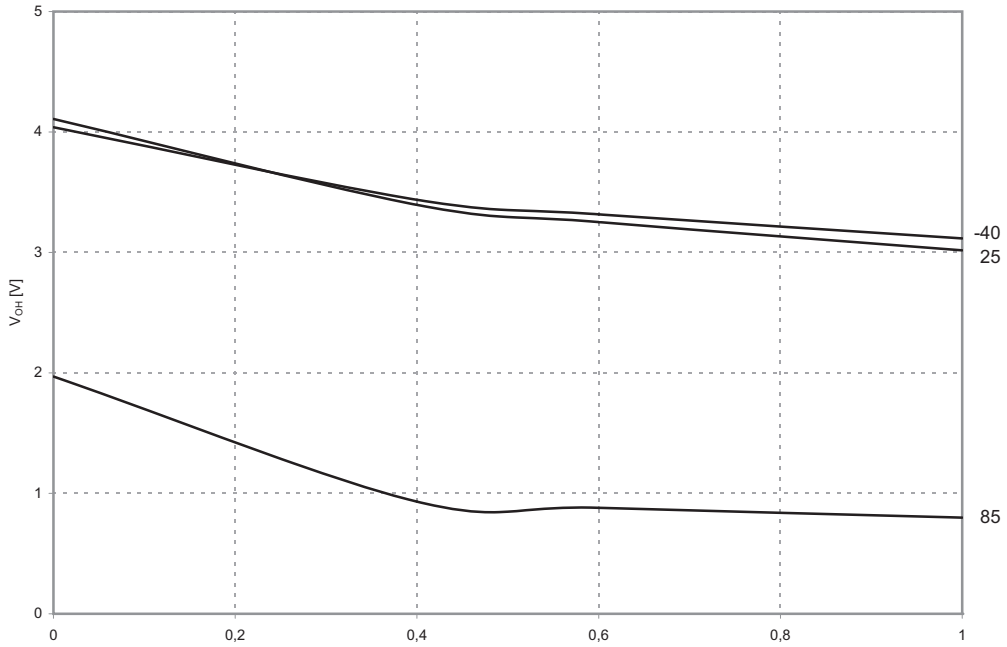


Figure 152. V_{OH}: Output Voltage vs. Source Current (Reset Pin as I/O, V_{CC} = 5V



25.9 Current Sink Capability

25.9.1 I/O Pins with Standard Sink Capability

Figure 153. V_{OL}: Output Voltage vs. Sink Current (Standard I/O Pin, V_{CC} = 1.8V)

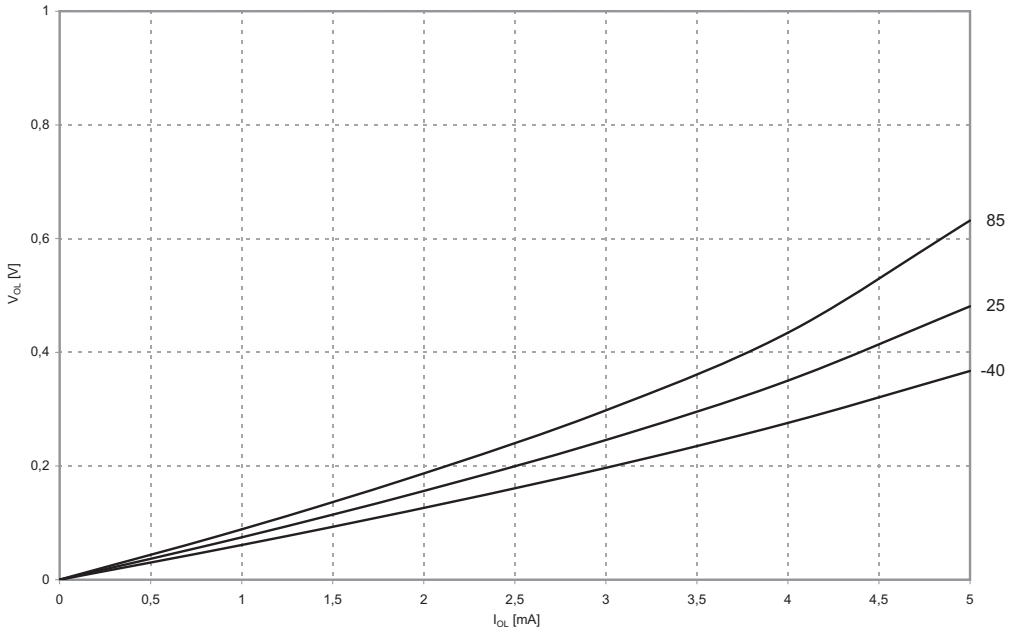
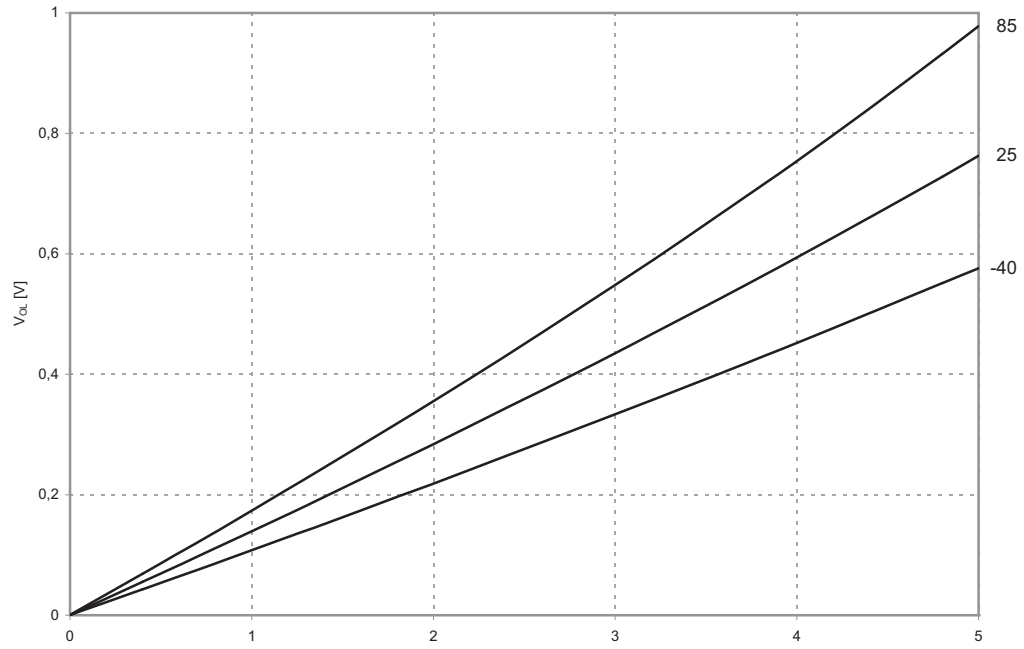


Figure 164. V_{OL}: Output Voltage vs. Sink Current (Reset Pin as I/O, V_{CC} = 5V)



25.10 BOD

Figure 165. BOD Threshold vs Temperature (BODLEVEL = 4.3V)