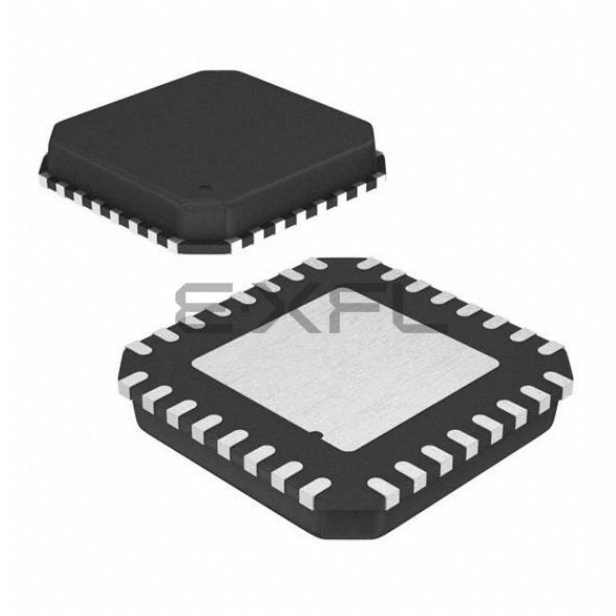


Welcome to E-XFL.COM

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	10MHz
Connectivity	USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	8KB (4K x 16)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 11x10b
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/attiny861v-10mu

The next code examples show assembly and C functions for reading the EEPROM. The examples assume that interrupts are controlled so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_read:
  ; Wait for completion of previous write
  sbic EECR,EEPE
  rjmp EEPROM_read
  ; Set up address (r18:r17) in address register
  out  EEARH, r18
  out  EEARL, r17
  ; Start eeprom read by writing EERE
  sbi  EECR,EERE
  ; Read data from data register
  in   r16,EEDR
  ret
```

C Code Example

```
unsigned char EEPROM_read(unsigned char ucAddress)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE))
    ;
  /* Set up address register */
  EEAR = ucAddress;
  /* Start eeprom read by writing EERE */
  EECR |= (1<<EERE);
  /* Return data from data register */
  return EEDR;
}
```

Note: See “Code Examples” on page 6.

5.3.7 Preventing EEPROM Corruption

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

An EEPROM data corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the EEPROM requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage is too low.

EEPROM data corruption can easily be avoided by following this design recommendation:

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 detection level of the internal BOD does not match the needed detection level, an external low V_{CC} reset protection circuit can

5.5.6 GPIOR1 – General Purpose I/O Register 1

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	MSB							LSB	GPIOR1
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	

5.5.7 GPIOR0 – General Purpose I/O Register 0

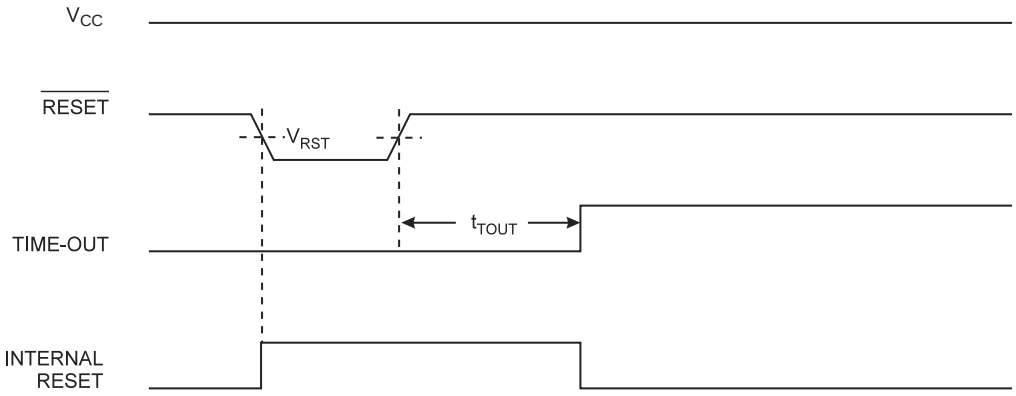
Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	MSB							LSB	GPIOR0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

cleared by hardware four cycles after it is written or when the CLKPS bits are written. Rewriting the CLKPCE bit within this time-out period does neither extend the time-out period, nor clear the CLKPCE bit.

8.1.2 External Reset

An External Reset is generated by a low level on the $\overline{\text{RESET}}$ pin if enabled. Reset pulses longer than the minimum pulse width (see “System and Reset Characteristics” on page 190) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} – on its positive edge, the delay counter starts the MCU after the Time-out period – t_{TOUT} – has expired.

Figure 8-4. External Reset During Operation



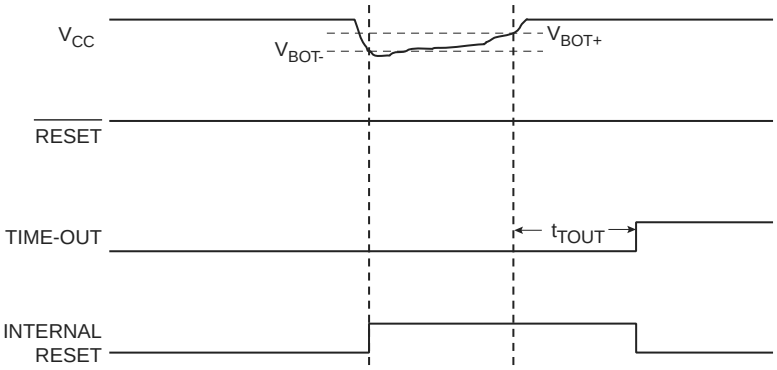
8.1.3 Brown-out Detection

ATtiny261/461/861 has an On-chip Brown-out Detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the BODLEVEL Fuses. The trigger level has a hysteresis to ensure spike free Brown-out Detection. The hysteresis on the detection level should be interpreted as $V_{BOT+} = V_{BOT} + V_{HYST}/2$ and $V_{BOT-} = V_{BOT} - V_{HYST}/2$.

When the BOD is enabled, and V_{CC} decreases to a value below the trigger level (V_{BOT-} in Figure 8-5), the Brown-out Reset is immediately activated. When V_{CC} increases above the trigger level (V_{BOT+} in Figure 8-5), the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in “System and Reset Characteristics” on page 190.

Figure 8-5. Brown-out Reset During Operation



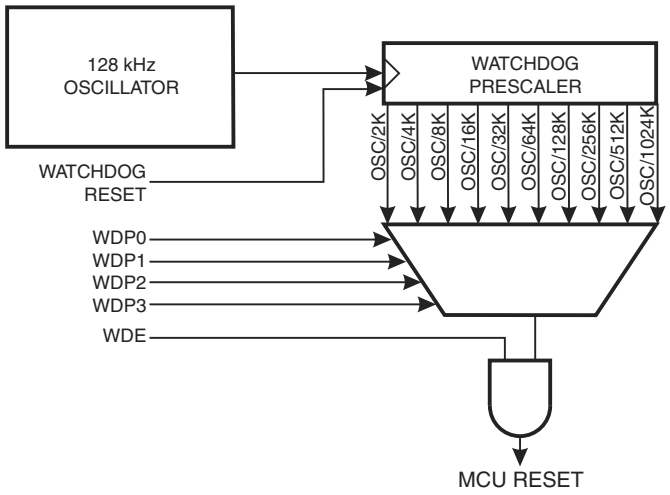
The Wathdog Timer can also be configured to generate an interrupt instead of a reset. This can be very helpful when using the Watchdog to wake-up from Power-down.

To prevent unintentional disabling of the Watchdog or unintentional change of time-out period, two different safety levels are selected by the fuse WDTON as shown in Table 8-1 Refer to “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 45 for details.

Table 8-1. WDT Configuration as a Function of the Fuse Settings of WDTON

WDTON	Safety Level	WDT Initial State	How to Disable the WDT	How to Change Time-out
Unprogrammed	1	Disabled	Timed sequence	No limitations
Programmed	2	Enabled	Always enabled	Timed sequence

Figure 8-7. Watchdog Timer



8.3.1 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.

8.3.1.1 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. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE regardless of the previous value of the WDE bit.
2. Within the next four clock cycles, in the same operation, write the WDE and WDP bits as desired, but with the WDCE bit cleared.

8.3.1.2 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:

the next time-out will generate a reset. To avoid the Watchdog Reset, WDIE must be set after each interrupt.

Table 8-2. Watchdog Timer Configuration

WDE	WDIE	Watchdog Timer State	Action on Time-out
0	0	Stopped	None
0	1	Running	Interrupt
1	0	Running	Reset
1	1	Running	Interrupt

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

This bit must be set when the WDE bit is written to logic zero. Otherwise, the Watchdog will not be disabled. Once written to one, hardware will clear this bit after four clock cycles. Refer to the description of the WDE bit for a Watchdog disable procedure. This bit must also be set when changing the prescaler bits. See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 45.

- **Bit 3 – WDE: Watchdog Enable**

When the WDE is written to logic one, the Watchdog Timer is enabled, and if the WDE is written to logic zero, the Watchdog Timer function is disabled. WDE can only be cleared if the WDCE bit has logic level one. To disable an enabled Watchdog Timer, the following procedure must be followed:

1. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE even though it is set to one before the disable operation starts.
2. Within the next four clock cycles, write a logic 0 to WDE. This disables the Watchdog.

In safety level 2, it is not possible to disable the Watchdog Timer, even with the algorithm described above. See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 45.

In safety level 1, WDE is overridden by WDRF in MCUSR. See “MCUSR – MCU Status Register” on page 47 for description of WDRF. This means that WDE is always set when WDRF is set. To clear WDE, WDRF must be cleared before disabling the Watchdog with the procedure described above. This feature ensures multiple resets during conditions causing failure, and a safe start-up after the failure.

Note: If the watchdog timer is not going to be used in the application, it is important to go through a watchdog disable procedure in the initialization of the device. If the Watchdog is accidentally enabled, for example by a runaway pointer or brown-out condition, the device will be reset, which in turn will lead to a new watchdog reset. To avoid this situation, the application software should always clear the WDRF flag and the WDE control bit in the initialization routine.

11.10 Register Description

11.10.1 TCCR0A – Timer/Counter0 Control Register A

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	TCW0	ICEN0	ICNC0	ICES0	ACIC0	–	–	CTC0	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – TCW0: Timer/Counter0 Width**

When this bit is written to one 16-bit mode is selected as described Figure 11-7 on page 79. Timer/Counter0 width is set to 16-bits and the Output Compare Registers OCR0A and OCR0B are combined to form one 16-bit Output Compare Register. Because the 16-bit registers TCNT0H/L and OCR0B/A are accessed by the AVR CPU via the 8-bit data bus, special procedures must be followed. These procedures are described in section “Accessing Registers in 16-bit Mode” on page 80.

- **Bit 6 – ICEN0: Input Capture Mode Enable**

When this bit is written to onem, the Input Capture Mode is enabled.

- **Bit 5 – ICNC0: Input Capture Noise Canceler**

Setting this bit activates the Input Capture Noise Canceler. When the noise canceler is activated, the input from the Input Capture Pin (ICP0) is filtered. The filter function requires four successive equal valued samples of the ICP0 pin for changing its output. The Input Capture is therefore delayed by four System Clock cycles when the noise canceler is enabled.

- **Bit 4 – ICES0: Input Capture Edge Select**

This bit selects which edge on the Input Capture Pin (ICP0) that is used to trigger a capture event. When the ICES0 bit is written to zero, a falling (negative) edge is used as trigger, and when the ICES0 bit is written to one, a rising (positive) edge will trigger the capture. When a capture is triggered according to the ICES0 setting, the counter value is copied into the Input Capture Register. The event will also set the Input Capture Flag (ICF0), and this can be used to cause an Input Capture Interrupt, if this interrupt is enabled.

- **Bit 3 - ACIC0: Analog Comparator Input Capture Enable**

When written logic one, this bit enables the input capture function in Timer/Counter0 to be triggered by the Analog Comparator. The comparator output is in this case directly connected to the input capture front-end logic, making the comparator utilize the noise canceler and edge select features of the Timer/Counter0 Input Capture interrupt. When written logic zero, no connection between the Analog Comparator and the input capture function exists. To make the comparator trigger the Timer/Counter0 Input Capture interrupt, the TICIE0 bit in the Timer Interrupt Mask Register (TIMSK) must be set.

- **Bits 2:1 – Res: Reserved Bits**

These bits are reserved and will always read zero.

- **Bit 0 – CTC0: Waveform Generation Mode**

This bit controls the counting sequence of the counter, the source for maximum (TOP) counter value, see Figure 11-7 on page 79. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter) and Clear Timer on Compare Match (CTC) mode (see “Modes of Operation” on page 77).

12.12.3 TCCR1C – Timer/Counter1 Control Register C

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	COM1A1S	COM1A0S	COM1B1S	COM1B0S	COM1D1	COM1D0	FOC1D	PWM1D	TCCR1C
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 7,6 – COM1A1S, COM1A0S: Comparator A Output Mode, Shadow Bits 1 and 0**

These are shadow bits of COM1A1 and COM1A0 in TCCR1A. Writing to bits COM1A1S and COM1A0S will also change bits COM1A1 and COM1A0 in TCCR1A. Similary, changes written to bits COM1A1 and COM1A0 in TCCR1A will show here.

See “TCCR1A – Timer/Counter1 Control Register A” on page 112 for information on bit usage.

- **Bits 5,4 – COM1B1S, COM1B0S: Comparator B Output Mode, Shadow Bits 1 and 0**

These are shadow bits of COM1B1 and COM1B0 in TCCR1A. Writing to bits COM1B1S and COM1B0S will also change bits COM1B1 and COM1B0 in TCCR1A. Similary, changes written to bits COM1B1 and COM1B0 in TCCR1A will show here.

See “TCCR1A – Timer/Counter1 Control Register A” on page 112 for information on bit usage.

- **Bits 3,2 – COM1D1, COM1D0: Comparator D Output Mode, Bits 1 and 0**

These bits control the behaviour of the Waveform Output (OCW1D) and the connection of the Output Compare pin (OC1D). If one or both of the COM1D1:0 bits are set, the OC1D output overrides the normal port functionality of the I/O pin it is connected to. The complementary $\overline{\text{OC1D}}$ output is connected only in PWM modes when the COM1D1:0 bits are set to “01”. Note that the Data Direction Register (DDR) bit corresponding to the OC1D pin must be set in order to enable the output driver.

The function of the COM1D1:0 bits depends on the PWM1D and WGM11:10 bit settings. Table 12-18 shows the COM1D1:0 bit functionality when the PWM1D bit is set to a Normal Mode (non-PWM).

Table 12-18. Compare Output Mode, Normal Mode (non-PWM)

COM1D1:0	OCW1D Behaviour	OC1D Pin	$\overline{\text{OC1D}}$ Pin
00	Normal port operation.	Disconnected	Disconnected
01	Toggle on Compare Match.	Connected	Disconnected
10	Clear on Compare Match.	Connected	Disconnected
11	Set on Compare Match.	Connected	Disconnected

Table 12-19 shows the COM1D1:0 bit functionality when the PWM1D and WGM11:10 bits are set to Fast PWM Mode.

Table 12-19. Compare Output Mode, Fast PWM Mode

COM1D1:0	OCW1D Behaviour	OC1D Pin	$\overline{\text{OC1D}}$ Pin
00	Normal port operation.	Disconnected	Disconnected
01	Cleared on Compare Match. Set when TCNT1=0x000.	Connected	Connected
10	Cleared on Compare Match. Set when TCNT1=0x000.	Connected	Disconnected
11	Set on Compare Match. Cleared when TCNT1=0x000.	Connected	Disconnected

13.3.3 SPI Slave Operation Example

The following code demonstrates how to use the USI module as a SPI Slave:

```
init:
    ldi    r16,(1<<USIWM0)|(1<<USICS1)
    sts    USICR,r16
...
SlaveSPITransfer:
    sts    USIDR,r16
    ldi    r16,(1<<USIOIF)
    sts    USISR,r16
SlaveSPITransfer_loop:
    lds    r16, USISR
    sbrs   r16, USIOIF
    rjmp   SlaveSPITransfer_loop
    lds    r16,USIDR
    ret
```

The code is size optimized using only eight instructions (+ ret). The code example assumes that the DO is configured as output and USCK pin is configured as input in the DDR Register. The value stored in register r16 prior to the function is called is transferred to the master device, and when the transfer is completed the data received from the Master is stored back into the r16 Register.

Note that the first two instructions are for initialization, only, and need only be executed once. These instructions set three-wire mode and positive edge clock. The loop is repeated until the USI Counter Overflow Flag is set.

13.3.4 Two-wire Mode

The USI Two-wire mode is compliant to the Inter IC (TWI) bus protocol, but without slew rate limiting on outputs and input noise filtering. Pin names used by this mode are SCL and SDA.

Figure 13-4 on page 130 shows two USI units operating in two-wire mode, one as master and one as slave. It is only the physical layer that is shown since the system operation is highly dependent of the communication scheme used. The main differences between the master and slave operation at this level is the serial clock generation which is always done by the master. Only the slave uses the clock control unit.

Clock generation must be implemented in software, but the shift operation is done automatically in both devices. Note that clocking only on negative edges for shifting data is of practical use in this mode. The slave can insert wait states at start or end of transfer by forcing the SCL clock low. This means that the master must always check if the SCL line was actually released after it has generated a positive edge.

Since the clock also increments the counter, a counter overflow can be used to indicate that the transfer is completed. The clock is generated by the master by toggling the USCK pin via the PORTA register.

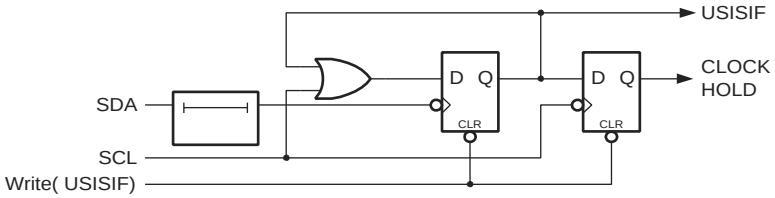
3. The master set the first bit to be transferred and releases the SCL line (C). The slave samples the data and shifts it into the USI Data Register at the positive edge of the SCL clock.
4. After eight bits containing slave address and data direction (read or write) have been transferred, the slave counter overflows and the SCL line is forced low (D). If the slave is not the one the master has addressed, it releases the SCL line and waits for a new start condition.
5. When the slave is addressed, it holds the SDA line low during the acknowledgment cycle before holding the SCL line low again (i.e., the USI Counter Register must be set to 14 before releasing SCL at (D)). Depending on the R/W bit the master or slave enables its output. If the bit is set, a master read operation is in progress (i.e., the slave drives the SDA line) The slave can hold the SCL line low after the acknowledge (E).
6. Multiple bytes can now be transmitted, all in same direction, until a stop condition is given by the master (F), or a new start condition is given.

If the slave is not able to receive more data it does not acknowledge the data byte it has last received. When the master does a read operation it must terminate the operation by forcing the acknowledge bit low after the last byte transmitted.

13.3.5 Start Condition Detector

The start condition detector is shown in Figure 13-6. The SDA line is delayed (in the range of 50 to 300 ns) to ensure valid sampling of the SCL line. The start condition detector is only enabled in Two-wire mode.

Figure 13-6. Start Condition Detector, Logic Diagram



The start condition detector works asynchronously and can therefore wake up the processor from power-down sleep mode. However, the protocol used might have restrictions on the SCL hold time. Therefore, when using this feature in this case the Oscillator start-up time set by the CKSEL Fuses (see “Clock System” on page 24) must also be taken into the consideration. Refer to the USISIF bit description on page 133 for further details.

13.3.6 Clock speed considerations

Maximum frequency for SCL and SCK is f_{CK} / 2. This is also the maximum data transmit and receive rate in both two- and three-wire mode. In two-wire slave mode the Two-wire Clock Control Unit will hold the SCL low until the slave is ready to receive more data. This may reduce the actual data rate in two-wire mode.

Table 14-1. Analog Comparator Multiplexed Input (Continued)

ACME	ADEN	MUX5:0	ACM2:0	Positive Input	Negative Input
0	x	xxxxxx	100	AIN2	AIN0
0	x	xxxxxx	101,110,111	AIN2	AIN1
1	1	xxxxxx	000	AIN0	AIN1
1	0	000000	000	AIN0	ADC0
1	0	000000	01x	AIN1	ADC0
1	0	000000	1xx	AIN2	ADC0
1	0	000001	000	AIN0	ADC1
1	0	000001	01x	AIN1	ADC1
1	0	000001	1xx	AIN2	ADC1
1	0	000010	000	AIN0	ADC2
1	0	000010	01x	AIN1	ADC2
1	0	000010	1xx	AIN2	ADC2
1	0	000011	000	AIN0	ADC3
1	0	000011	01x	AIN1	ADC3
1	0	000011	1xx	AIN2	ADC3
1	0	000100	000	AIN0	ADC4
1	0	000100	01x	AIN1	ADC4
1	0	000100	1xx	AIN2	ADC4
1	0	000101	000	AIN0	ADC5
1	0	000101	01x	AIN1	ADC5
1	0	000101	1xx	AIN2	ADC5
1	0	000110	000	AIN0	ADC6
1	0	000110	01x	AIN1	ADC6
1	0	000110	1xx	AIN2	ADC6
1	0	000111	000	AIN0	ADC7
1	0	000111	01x	AIN1	ADC7
1	0	000111	1xx	AIN2	ADC7
1	0	001000	000	AIN0	ADC8
1	0	001000	01x	AIN1	ADC8
1	0	001000	1xx	AIN2	ADC8
1	0	001001	000	AIN0	ADC9
1	0	001001	01x	AIN1	ADC9
1	0	001001	1xx	AIN2	ADC9
1	0	001010	000	AIN0	ADC10
1	0	001010	01x	AIN1	ADC10
1	0	001010	1xx	AIN2	ADC10

15. ADC – Analog to Digital Converter

15.1 Features

- **10-bit Resolution**
- **1.0 LSB Integral Non-linearity**
- **± 2 LSB Absolute Accuracy**
- **13µs Conversion Time**
- **15 kSPS at Maximum Resolution**
- **11 Multiplexed Single Ended Input Channels**
- **16 Differential input pairs**
- **15 Differential input pairs with selectable gain**
- **Temperature Sensor Input Channel**
- **Optional Left Adjustment for ADC Result Readout**
- **0 - V_{CC} ADC Input Voltage Range**
- **Selectable 1.1V / 2.56V ADC Voltage Reference**
- **Free Running or Single Conversion Mode**
- **ADC Start Conversion by Auto Triggering on Interrupt Sources**
- **Interrupt on ADC Conversion Complete**
- **Sleep Mode Noise Cancele**
- **Unipolar / Bipolar Input Mode**
- **Input Polarity Reversal Mode**

15.2 Overview

The ATtiny261/461/861 features a 10-bit successive approximation ADC. The ADC is connected to a 11-channel Analog Multiplexer which allows 16 differential voltage input combinations and 11 single-ended voltage inputs constructed from the pins PA7:PA0 or PB7:PB4. The differential input is equipped with a programmable gain stage, providing amplification steps of 1x, 8x, 20x or 32x on the differential input voltage before the A/D conversion. The single-ended voltage inputs refer to 0V (GND).

The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 15-1 on page 143.

Internal reference voltages of nominally 1.1V or 2.56V are provided On-chip. The Internal reference voltage of 2.56V, can optionally be externally decoupled at the AREF (PA3) pin by a capacitor, for better noise performance. Alternatively, V_{CC} can be used as reference voltage for single ended channels. There is also an option to use an external voltage reference and turn-off the internal voltage reference. These options are selected using the REFS2:0 bits of the ADCSRB and ADMUX registers.

reference may be decoupled by an external capacitor at the AREF pin to improve noise immunity.

The analog input channel and differential gain are selected by writing to the MUX5:0 bits in ADMUX. Any of the 11 ADC input pins ADC10:0 can be selected as single ended inputs to the ADC. The positive and negative inputs to the differential gain amplifier are described in Table 15-4.

If differential channels are selected, the differential gain stage amplifies the voltage difference between the selected input pair by the selected gain factor, 1x, 8x, 20x or 32x, according to the setting of the MUX5:0 bits in ADMUX and the GSEL bit in ADCSRB. This amplified value then becomes the analog input to the ADC. If single ended channels are used, the gain amplifier is bypassed altogether.

If the same ADC input pin is selected as both the positive and negative input to the differential gain amplifier, the remaining offset in the gain stage and conversion circuitry can be measured directly as the result of the conversion. This figure can be subtracted from subsequent conversions with the same gain setting to reduce offset error to below 1 LSW.

The on-chip temperature sensor is selected by writing the code “111111” to the MUX5:0 bits in ADMUX register when the ADC11 channel is used as an ADC input.

The ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. Voltage reference and input channel selections will not go into effect until ADEN is set. The ADC does not consume power when ADEN is cleared, so it is recommended to switch off the ADC before entering power saving sleep modes.

The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted, but can optionally be presented left adjusted by setting the ADLAR bit in ADMUX.

If the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH, to ensure that the content of the data registers belongs to the same conversion. Once ADCL is read, ADC access to data registers is blocked. This means that if ADCL has been read, and a conversion completes before ADCH is read, neither register is updated and the result from the conversion is lost. When ADCH is read, ADC access to the ADCH and ADCL Registers is re-enabled.

The ADC has its own interrupt which can be triggered when a conversion completes. When ADC access to the data registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result is lost.

15.4 Starting a Conversion

A single conversion is started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. If a different data channel is selected while a conversion is in progress, the ADC will finish the current conversion before performing the channel change.

Alternatively, a conversion can be triggered automatically by various sources. Auto Triggering is enabled by setting the ADC Auto Trigger Enable bit, ADATE in ADCSRA. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB (see description of the ADTS bits for a list of the trigger sources). When a positive edge occurs on the selected trigger signal, the ADC prescaler is reset and a conversion is started. This provides a method of starting conversions at fixed intervals. If the trigger signal still is set when the conversion completes, a new

described in Section 15.12 on page 154. A good system design with properly placed, external bypass capacitors does reduce the need for using ADC Noise Reduction Mode

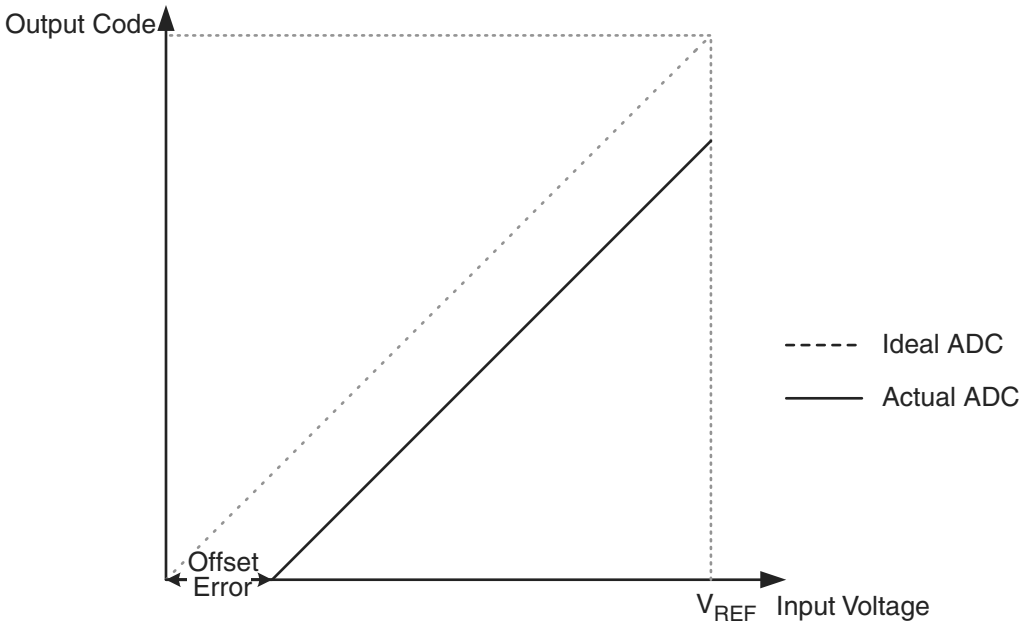
15.10 ADC Accuracy Definitions

An n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSBs). The lowest code is read as 0, and the highest code is read as 2^n-1.

Several parameters describe the deviation from the ideal behavior:

- Offset: The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSB). Ideal value: 0 LSB.

Figure 15-9. Offset Error



18. Memory Programming

This section describes the different methods for programming ATtiny261/461/861 memories.

18.1 Program And Data Memory Lock Bits

The ATtiny261/461/861 provides two lock bits which can be left unprogrammed (“1”) or can be programmed (“0”) to obtain the additional security listed in Table 18-2. The lock bits can only be erased to “1” with the Chip Erase command.

The device has no separate boot loader section. The SPM instruction is enabled for the whole Flash, if the SELFPROGEN fuse is programmed (“0”), otherwise it is disabled.

Program memory can be read out via the debugWIRE interface when the DWEN fuse is programmed, even if lock bits are set. Thus, when lock bit security is required, debugWIRE should always be disabled by clearing the DWEN fuse.

Table 18-1. Lock Bit Byte

Lock Bit Byte	Bit No	Description	Default Value
	7	–	1 (unprogrammed)
	6	–	1 (unprogrammed)
	5	–	1 (unprogrammed)
	4	–	1 (unprogrammed)
	3	–	1 (unprogrammed)
	2	–	1 (unprogrammed)
LB2	1	Lock bit	1 (unprogrammed)
LB1	0	Lock bit	1 (unprogrammed)

Note: “1” means unprogrammed, “0” means programmed.

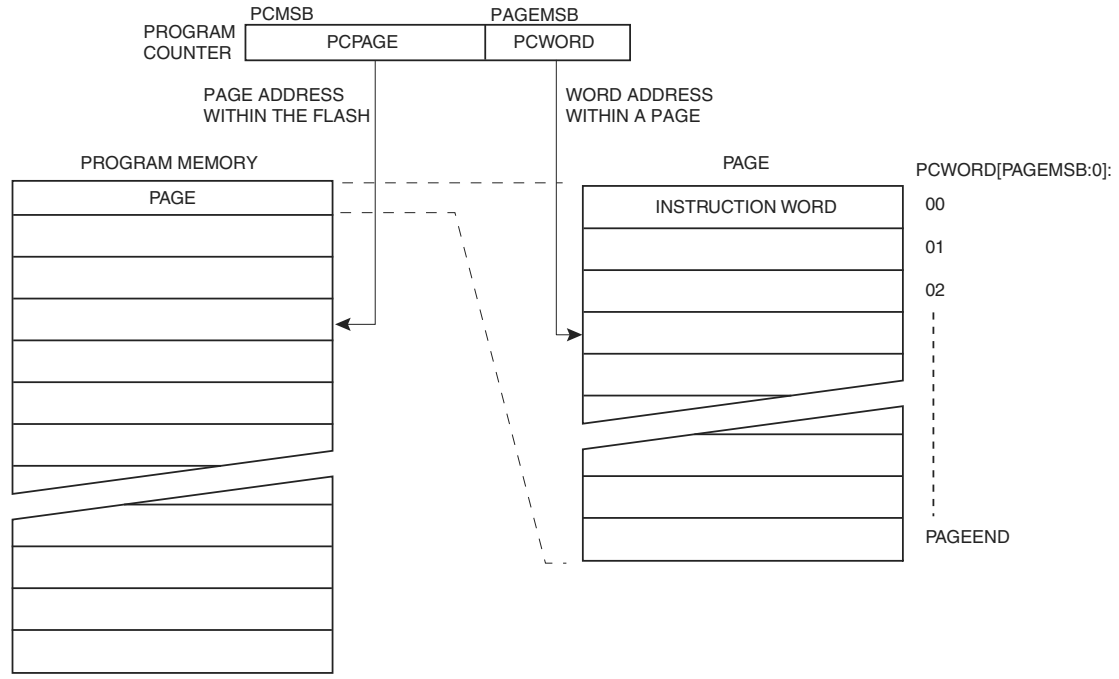
Table 18-2. Lock Bit Protection Modes.

Memory Lock Bits (1) (2)			Protection Type
LB Mode	LB2	LB1	
1	1	1	No memory lock features enabled.
2	1	0	Further programming of the Flash and EEPROM is disabled in High-voltage and Serial Programming mode. The Fuse bits are locked in both Serial and High-voltage Programming mode.(1)
3	0	0	Further programming and verification of the Flash and EEPROM is disabled in High-voltage and Serial Programming mode. The Fuse bits are locked in both Serial and High-voltage Programming mode.(1)

Notes: 1. Program fuse bits before programming LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed.

Lock bits can also be read by device firmware. See section “Reading Fuse and Lock Bits from Software” on page 167.

Figure 18-4. Addressing the Flash Which is Organized in Pages

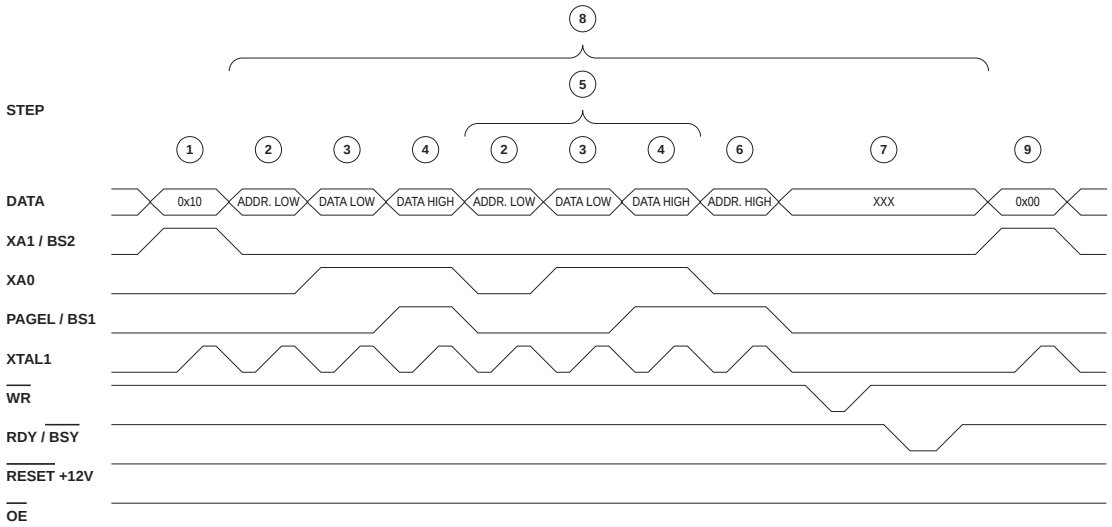


Note: PCPAGE and PCWORD are listed in Table 18-7 on page 173.

In the figure below, “XX” means don’t care. The numbers in the figure refer to the programming description above.

WR

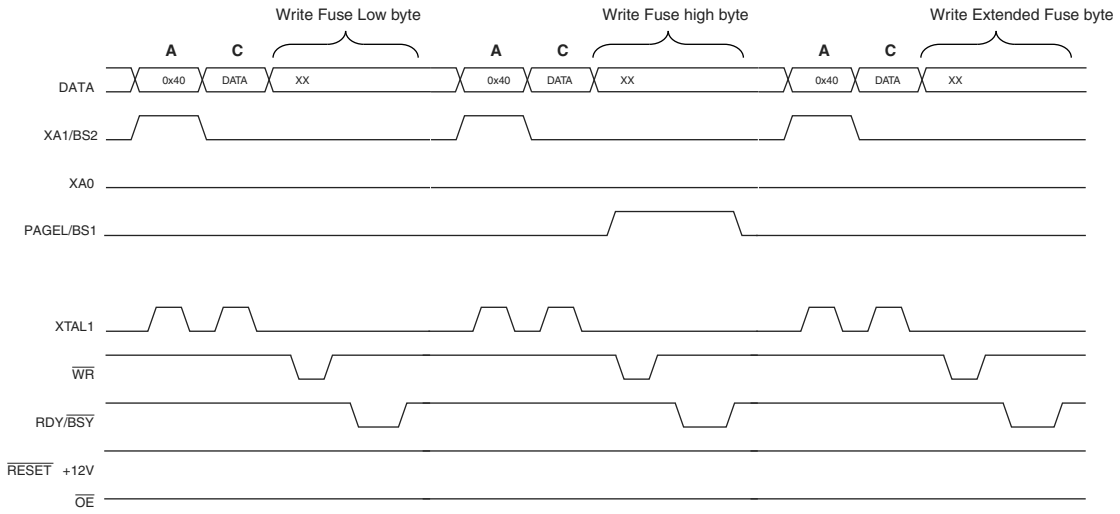
Figure 18-5. Flash Programming Waveforms



18.7.6 Programming the EEPROM

The EEPROM is organized in pages, see Table 18-8 on page 173. When programming the EEPROM, the program data is latched into a page buffer. This allows one page of data to be

Figure 18-7. Programming the FUSES Waveforms



18.7.12 Programming the Lock Bits

The algorithm for programming the Lock bits is as follows (refer to “Programming the Flash” on page 180 for details on Command and Data loading):

1. A: Load Command “0010 0000”.
2. C: Load Data Low Byte. Bit n = “0” programs the Lock bit. If LB mode 3 is programmed (LB1 and LB2 is programmed), it is not possible to program the Boot Lock bits by any External Programming mode.
3. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

The Lock bits can only be cleared by executing Chip Erase.

18.7.13 Reading the Fuse and Lock Bits

The algorithm for reading the Fuse and Lock bits is as follows (refer to “Programming the Flash” on page 180 for details on Command loading):

1. A: Load Command “0000 0100”.
2. Set $\overline{OE}$ to “0”, BS2 to “0” and BS1 to “0”. The status of the Fuse Low bits can now be read at DATA (“0” means programmed).
3. Set $\overline{OE}$ to “0”, BS2 to “1” and BS1 to “1”. The status of the Fuse High bits can now be read at DATA (“0” means programmed).
4. Set OE to “0”, BS2 to “1”, and BS1 to “0”. The status of the Extended Fuse bits can now be read at DATA (“0” means programmed).
5. Set $\overline{OE}$ to “0”, BS2 to “0” and BS1 to “1”. The status of the Lock bits can now be read at DATA (“0” means programmed).
6. Set $\overline{OE}$ to “1”.

Figure 20-2. Active Supply Current vs. Frequency (1 - 20 MHz)

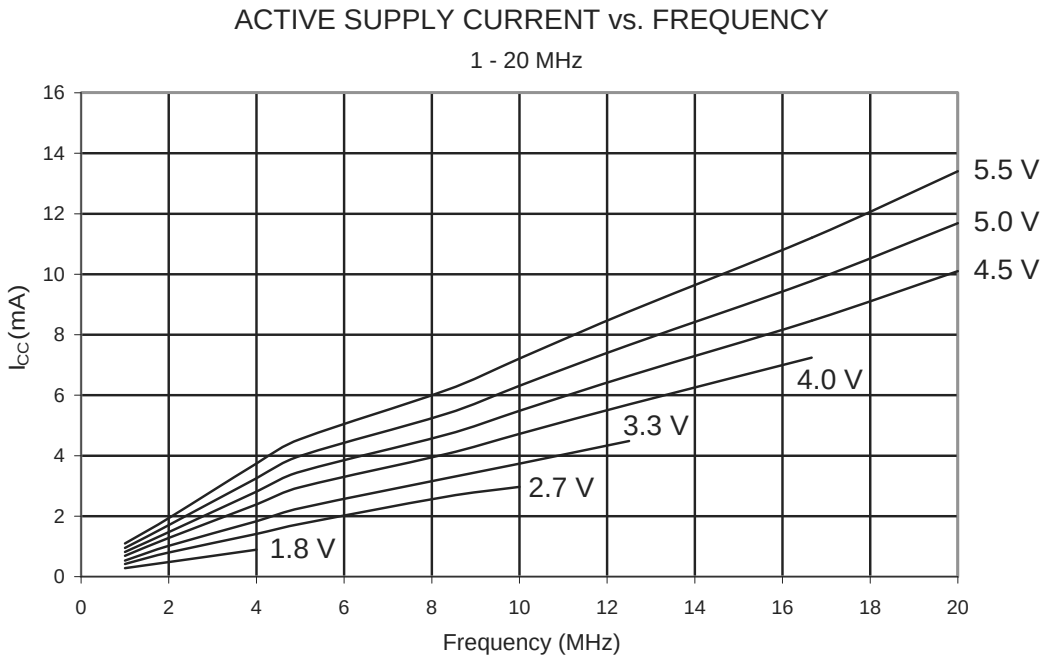


Figure 20-3. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz)

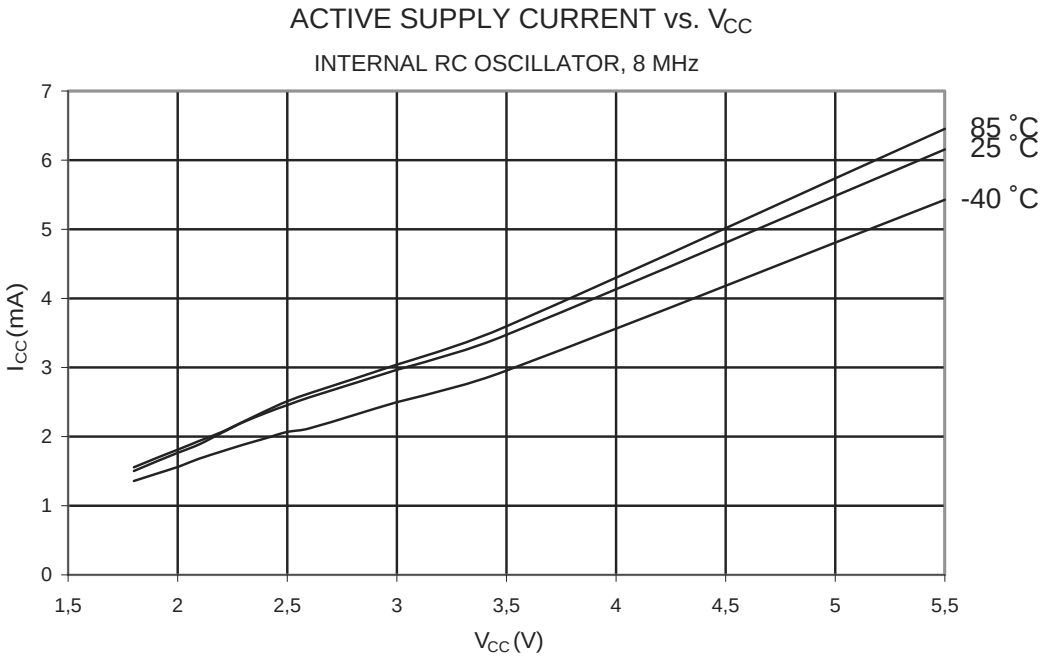


Figure 20-14. I/O Pin pull-up Resistor Current vs. Input Voltage (V_{CC} = 2.7V)

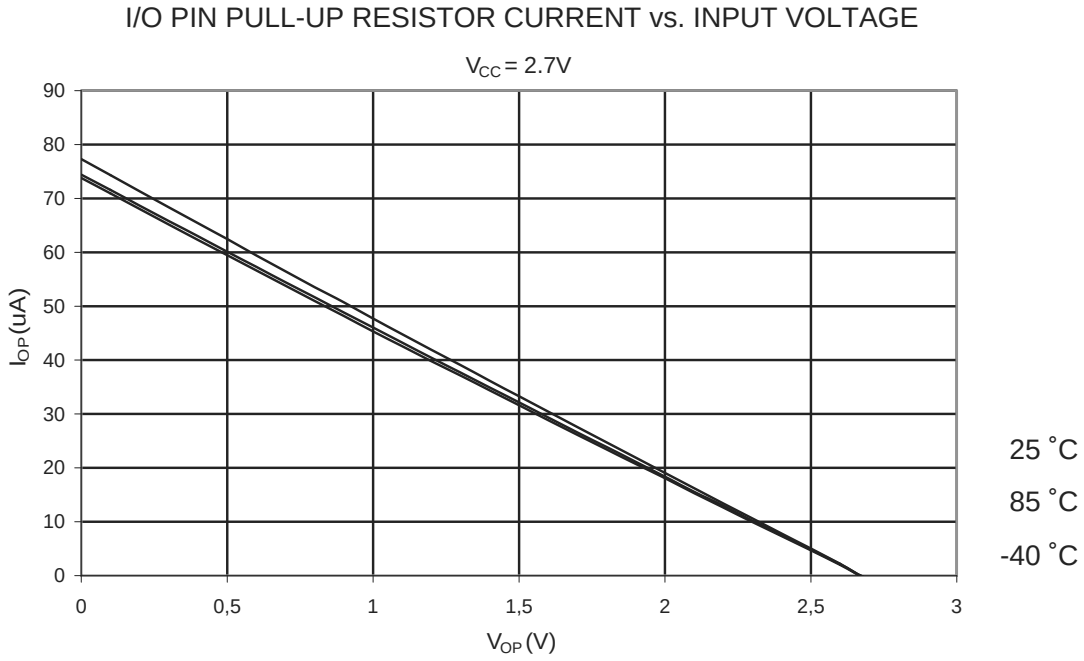


Figure 20-15. I/O Pin pull-up Resistor Current vs. Input Voltage (V_{CC} = 5V)

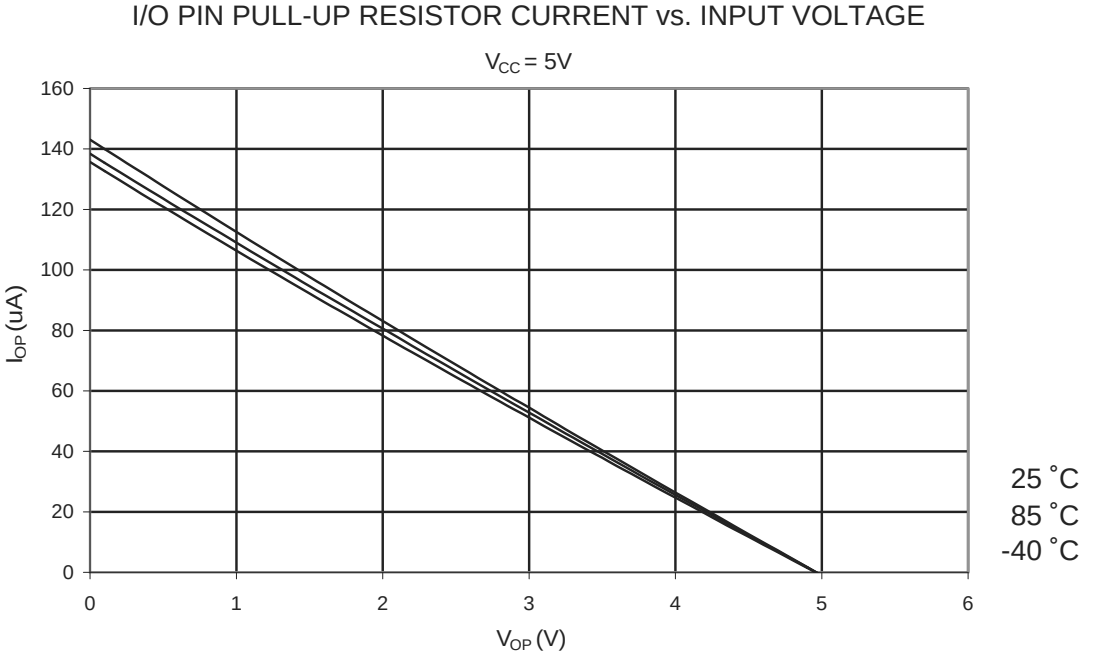


Figure 20-42. AREF External Reference Current vs. V_{CC}

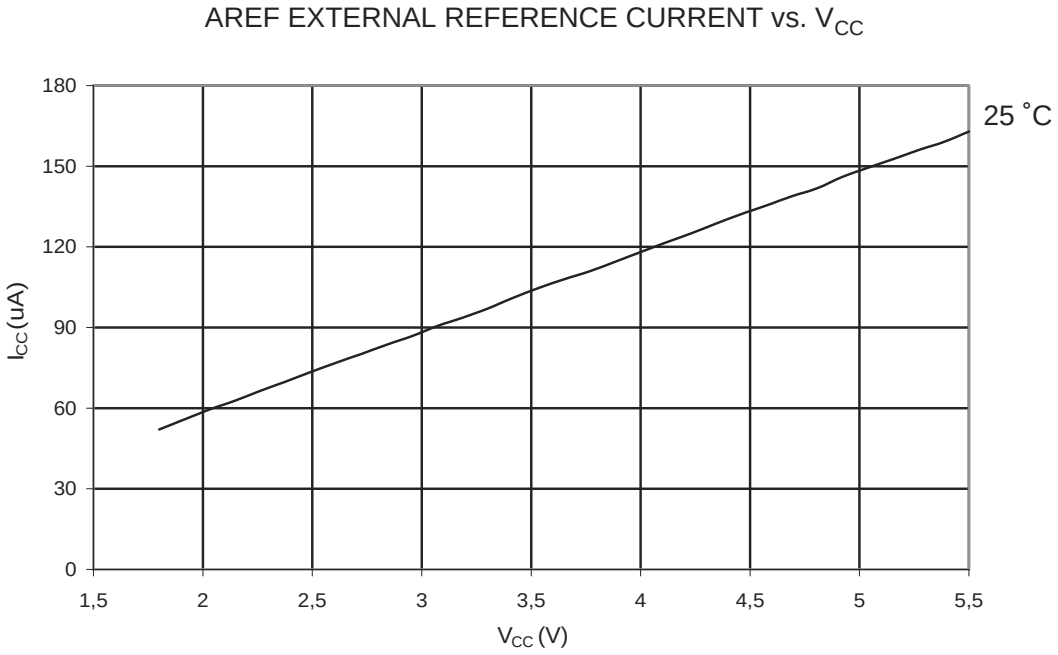


Figure 20-43. Analog Comparator vs. V_{CC}