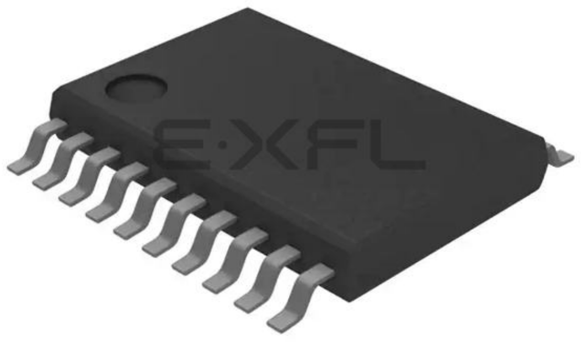
Microchip Technology - ATTINY461-15XD Datasheet



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	USI
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	16
Program Memory Size	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 150°C (TA)
Mounting Type	Surface Mount
Package / Case	20-TSSOP (0.173", 4.40mm Width)
Supplier Device Package	20-TSSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny461-15xd

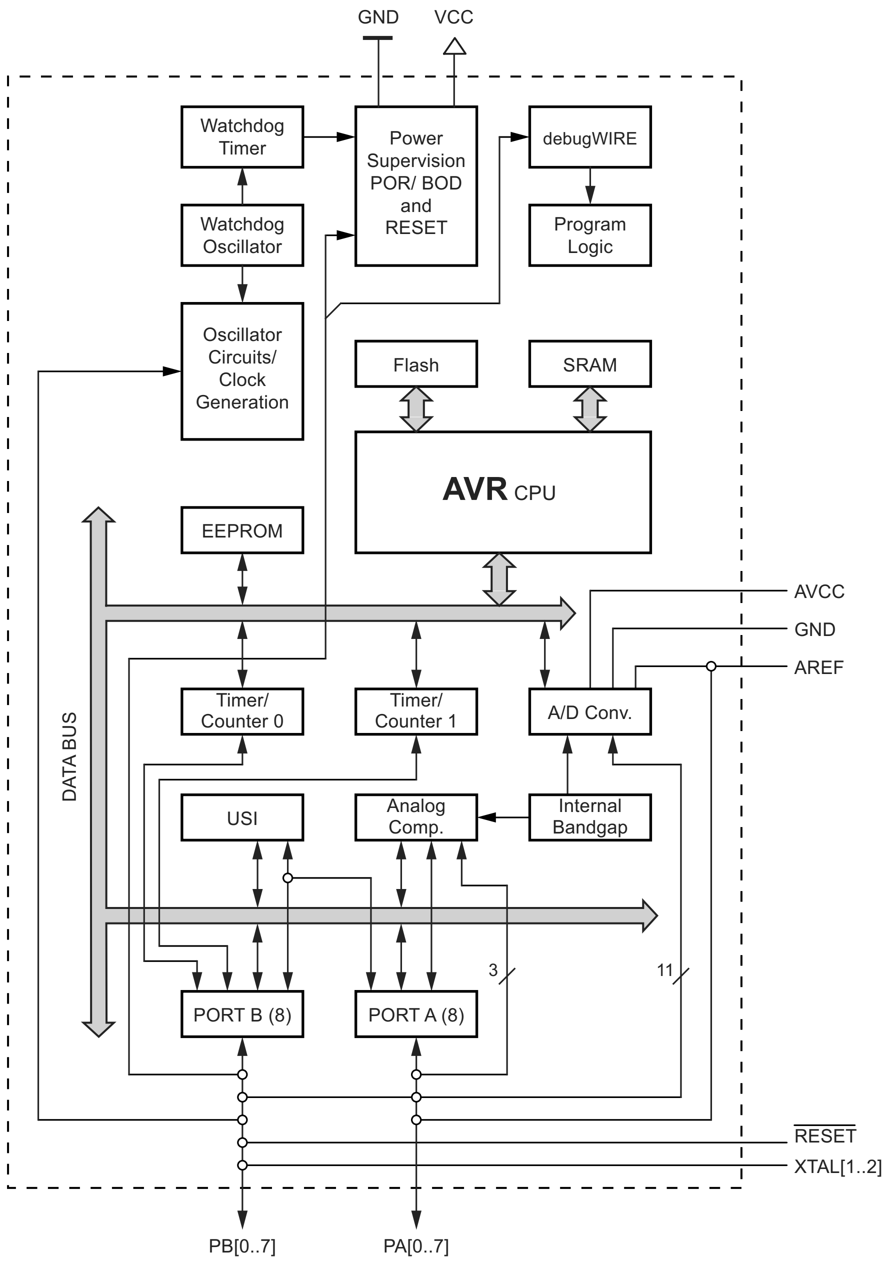
- I/O and packages
  - 16 programmable I/O lines
  - 20-pin SOIC, 32-pad MLF and 20-lead TSSOP
- Operating voltage:
  - 2.7 - 5.5V for Atmel ATtiny261/461/861
- Speed grade:
  - Atmel® ATtiny261/461/861: 0 - 8MHz at 2.7 - 5.5V, 0 - 16MHz at 4.5 - 5.5V
  - Operating temperature: Automotive (–40°C to +125°C)
- Low power consumption
  - Active mode ATD On: 1MHz, 2.7V, 25°C: 300µA
  - Power-down mode no watchdog: 2.7V, 25°C: 0.12µA

2. Overview

The Atmel® ATtiny261/461/861 is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the Atmel ATtiny261/461/861 achieves throughputs approaching 1MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The fast-access register file contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle arithmetic logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the register file, the operation is executed, and the result is stored back in the register file – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for data space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the status register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR® instructions have a single 16-bit word format. Most AVR instructions are 16-bit wide. There are also 32-bit instructions.

During interrupts and subroutine calls, the return address program counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM, and consequently the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The stack pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All interrupts have a separate interrupt vector in the interrupt vector table. The interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as control registers, SPI, and other I/O functions. The I/O memory can be accessed directly, or as the data space locations following those of the register file, 0x20 - 0x5F.

5.2 ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

5.5.1 SPH and SPL – Stack Pointer Register

Bit	15	14	13	12	11	10	9	8	
0x3E (0x5E)	**SP15**	**SP14**	**SP13**	**SP12**	**SP11**	**SP10**	**SP9**	**SP8**	**SPH**
0x3D (0x5D)	**SP7**	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**	**SPL**
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	
	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	

5.6 Instruction Execution Timing

This section describes the general access timing concepts for instruction execution. The AVR® CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

Figure 5-4 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast access register file concept. This is the basic pipelining concept to obtain up to 1MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 5-4. The Parallel Instruction Fetches and Instruction Executions

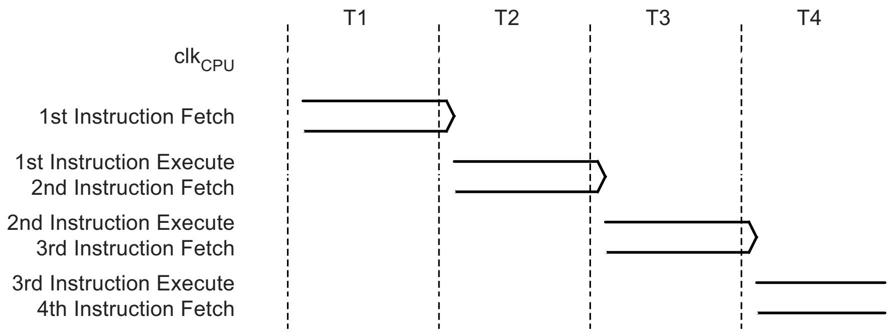
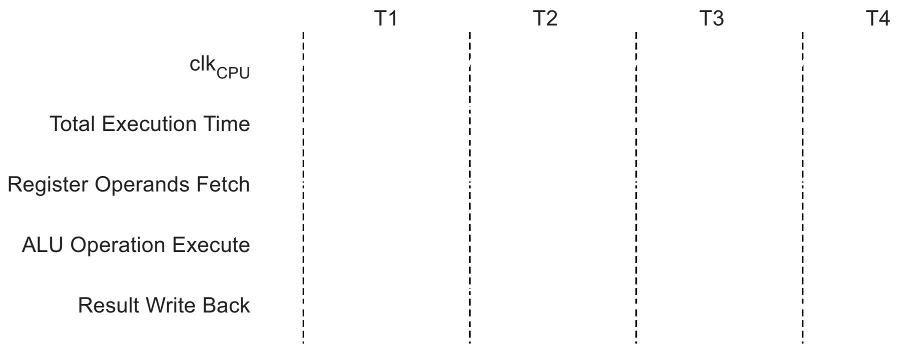


Figure 5-5 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

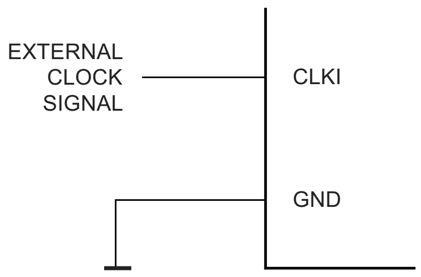
Figure 5-5. Single Cycle ALU Operation



7.4 External Clock

To drive the device from an external clock source, CLKI should be driven as shown in Figure 7-3. To run the device on an external clock, the CKSEL fuses must be programmed to “0000”.

Figure 7-3. External Clock Drive Configuration



When this clock source is selected, start-up times are determined by the SUT fuses as shown in Table 7-3.

Table 7-3. Start-up Times for the External Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset	Recommended Usage
00	6CK	14CK	BOD enabled
01	6CK	14CK + 4ms	Fast rising power
10	6CK	14CK + 64ms	Slowly rising power
11	Reserved		

Note that the system clock prescaler can be used to implement run-time changes of the internal clock frequency while still ensuring stable operation. Refer to Section 7.11 “System Clock Prescaler” on page 31 for details.

7.5 High Frequency PLL Clock - PLL_{CLK}

There is an internal PLL that provides nominally 64MHz clock rate locked to the RC oscillator for the use of the peripheral Timer/Counter1 and for the system clock source. When selected as a system clock source, by programming the CKSEL fuses to ‘0001’, it is divided by four like shown in Table 7-4. When this clock source is selected, start-up times are determined by the SUT fuses as shown in Table 7-5. See also Section 7-2 “PCK Clocking System” on page 25.

Table 7-4. PLLCK Operating Modes

CKSEL3..0	Nominal Frequency
0001	16MHz

Table 7-5. Start-up Times for the PLLCK

SUT1..0	Start-up Time from Power Down	Additional Delay from Power-On-Reset (V_{CC} = 5.0V)	Recommended Usage
00	14CK + 1K (1024) + 4ms	4ms	BOD enabled
01	14CK + 16K (16384) + 4ms	4ms	Fast rising power
10	14CK + 1K (1024) + 64ms	4ms	Slowly rising power
11	14CK + 16K (16384) + 64ms	4ms	Slowly rising power

9.3 Power-on Reset

A power-on reset (POR) pulse is generated by an on-chip detection circuit. The detection level is defined in Section 23.5 “System and Reset Characteristics” on page 174. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the start-up reset, as well as to detect a failure in supply voltage.

A power-on reset (POR) circuit ensures that the device is reset from power-on. Reaching the power-on reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise. The RESET signal is activated again, without any delay, when V_{CC} decreases below the detection level.

Figure 9-2. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}

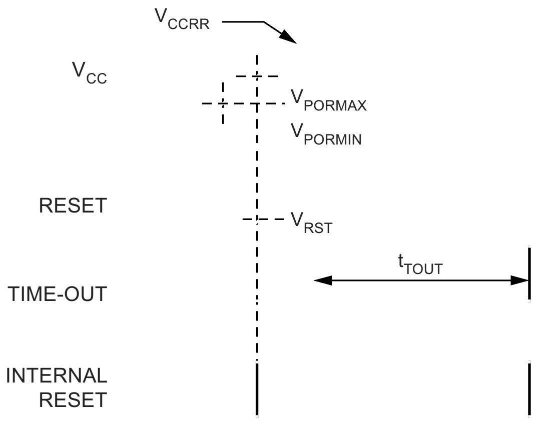


Figure 9-3. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally

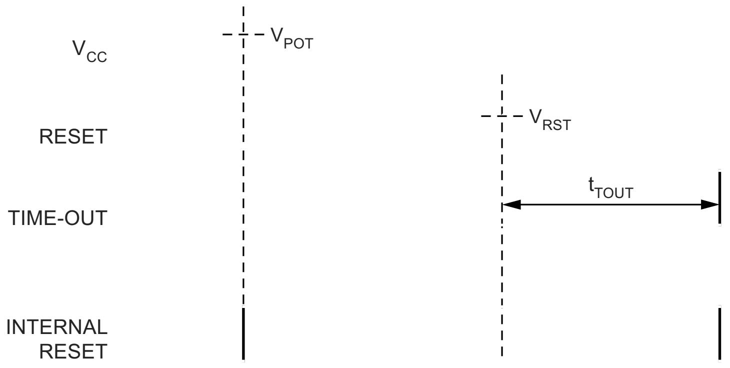


Table 9-1. Power On Reset Specifications

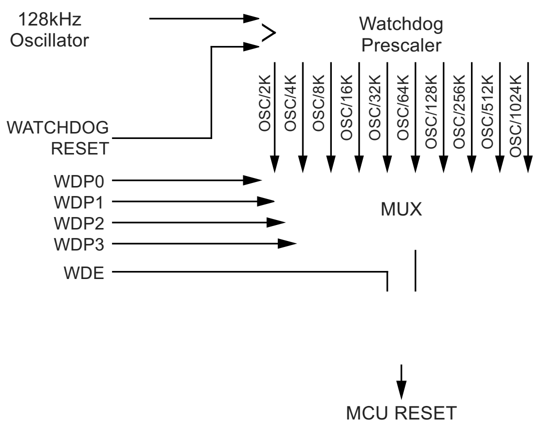
Parameter	Symbol	Min	Typ	Max	Unit
Power-on reset threshold voltage (rising)	V_{POT}	1.1	1.4	1.7	V
Power-on reset threshold voltage (falling)(1)		0.8	1.3	1.6	V
VCC max. start voltage to ensure internal power-on reset signal	V_{PORMAX}			0.4	V
VCC min. start voltage to ensure internal power-on reset signal	V_{PORMIN}	–0.1			V
VCC rise rate to ensure power-on reset	V_{CCRR}	0.01			V/ms
$\overline{\text{RESET}}$ pin threshold voltage	V_{RST}	0.1 V_{CC}		0.9V_{CC}	V

Note: 1. Before rising, the supply has to be between V_{PORMIN} and V_{PORMAX} to ensure a reset.

Table 9-2. 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 9-7. Watchdog Timer



9.9 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.

9.9.1 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.

9.9.2 Safety Level 2

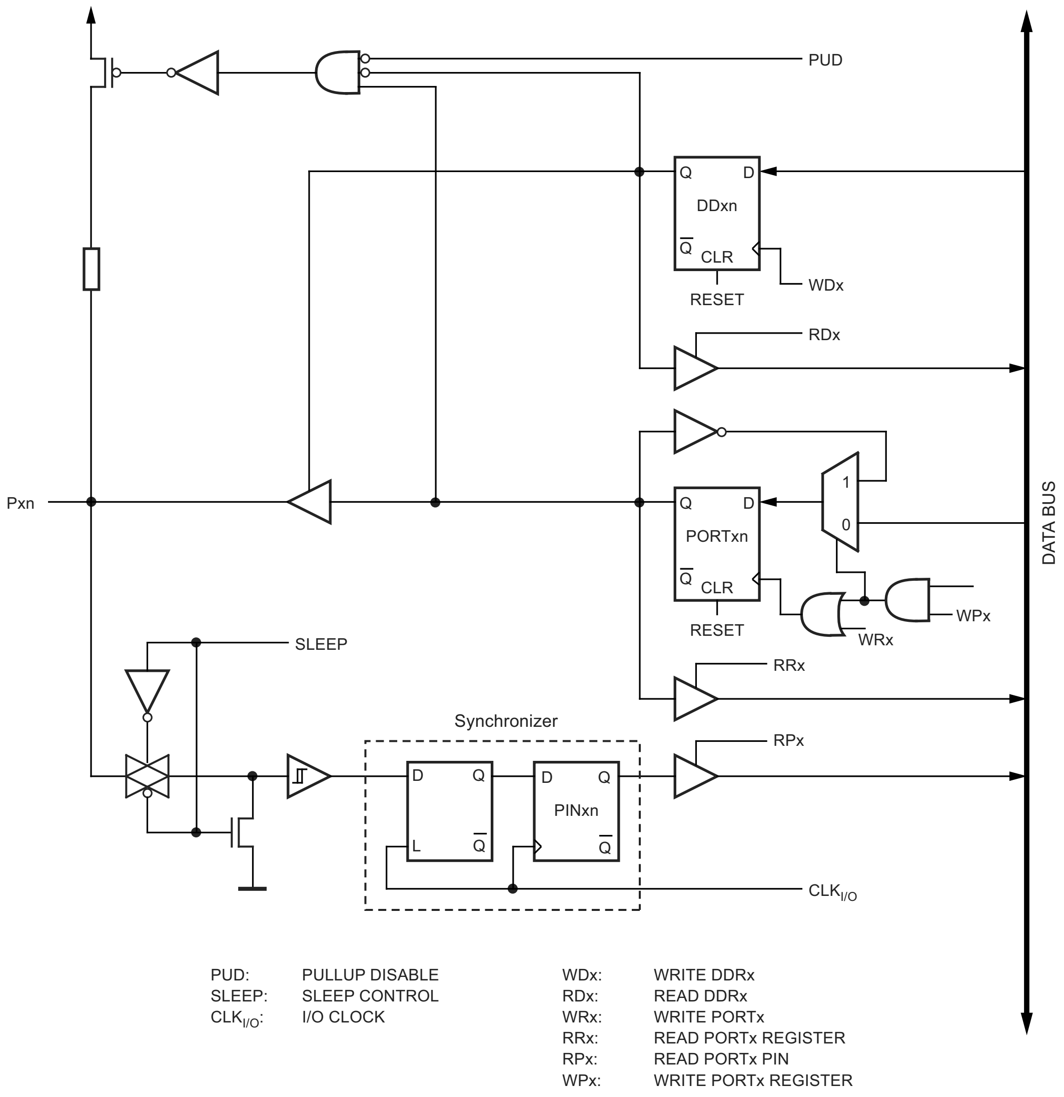
In this mode, the watchdog timer is always enabled, and the WDE bit will always read as one. A timed sequence is needed when changing the watchdog time-out period. To change the watchdog time-out, the following procedure must be followed:

1. In the same operation, write a logical one to WDCE and WDE. Even though the WDE always is set, the WDE must be written to one to start the timed sequence.
2. Within the next four clock cycles, in the same operation, write the WDP bits as desired, but with the WDCE bit cleared. The value written to the WDE bit is irrelevant.

12.2 Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 12-2 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 12-2. General Digital I/O(1)



Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. clk$_{I/O}$, SLEEP, and PUD are common to all ports.

12.2.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in Section 12.4 "Register Description" on page 64, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

The following code example shows how to set port B pins 0 and 1 high, 2 and 3 low, and define the port pins from 4 to 5 as input with a pull-up assigned to port pin 4. The resulting pin values are read back again, but as previously discussed, a nop instruction is included to be able to read back the value recently assigned to some of the pins.

Assembly Code Example[(1)]

```
        ...
        ; Define pull-ups and set outputs high
        ; Define directions for port pins
        ldi    r16,(1<<PB4)|(1<<PB1)|(1<<PB0)
        ldi    r17,(1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
        out    PORTB,r16
        out    DDRB,r17
        ; Insert nop for synchronization
        nop
        ; Read port pins
        in     r16,PINB
        ...
```

C Code Example

```
unsigned char i;
        ...
        /* Define pull-ups and set outputs high */
        /* Define directions for port pins */
        PORTB = (1<<PB4)|(1<<PB1)|(1<<PB0);
        DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
        /* Insert nop for synchronization*/
        _NOP();
        /* Read port pins */
        i = PINB;
        ...
```

Note: 1. For the assembly program, two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1 and 4, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

12.2.5 Digital Input Enable and Sleep Modes

As shown in Figure 12-2 on page 53, the digital input signal can be clamped to ground at the input of the schmitt-trigger. The signal denoted SLEEP in the figure, is set by the MCU sleep controller in power-down mode, power-save mode, and standby mode to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

SLEEP is overridden for port pins enabled as external interrupt pins. If the external interrupt request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in Section 12.3 “Alternate Port Functions” on page 57.

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

12.2.6 Unconnected Pins

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

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

12.4.4 PINA – Port A Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x19 (0x39)	**PINA7**	**PINA6**	**PINA5**	**PINA4**	**PINA3**	**PINA2**	**PINA1**	**PINA0**	**PINA**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

12.4.5 PORTB – Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x18 (0x38)	**PORTB7**	**PORTB6**	**PORTB5**	**PORTB4**	**PORTB3**	**PORTB2**	**PORTB1**	**PORTB0**	**PORTB**
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	

12.4.6 DDRB – Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x17 (0x37)	**DDB7**	**DDB6**	**DDB5**	**DDB4**	**DDB3**	**DDB2**	**DDB1**	**DDB0**	**DDRB**
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	

12.4.7 PINB – Port B Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	**PINB7**	**PINB6**	**PINB5**	**PINB4**	**PINB3**	**PINB2**	**PINB1**	**PINB0**	**PINB**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

14.10.7 TIFR – Timer/Counter0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x38 (0x58)	**OCF1D**	**OCF1A**	**OCF1B**	**OCF0A**	**OCF0B**	**TOV1**	**TOV0**	**ICF0**	**TIFR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 4– OCF0A: Output Compare Flag 0 A**

The OCF0A bit is set when a compare match occurs between the Timer/Counter0 and the data in OCR0A – output compare register0. OCF0A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0A (Timer/Counter0 compare match Interrupt Enable), and OCF0A are set, the Timer/Counter0 compare match interrupt is executed.

The OCF0A is also set in 16-bit mode when a compare match occurs between the Timer/Counter and 16-bit data in OCR0B/A. The OCF0A is not set in input capture mode when the output compare register OCR0A is used as an Input capture register.

- **Bit 3 – OCF0B: Output Compare Flag 0 B**

The OCF0B bit is set when a compare match occurs between the Timer/Counter and the data in OCR0B – output compare register0 B. OCF0B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0B (Timer/Counter compare B match interrupt enable), and OCF0B are set, the Timer/Counter Compare match interrupt is executed.

The OCF0B is not set in 16-bit output compare mode when the output compare register OCR0B is used as the high byte of the 16-bit output compare register or in 16-bit input capture mode when the output compare register OCR0B is used as the high byte of the input capture register.

- **Bit 1 – TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 overflow interrupt enable), and TOV0 are set, the Timer/Counter0 Overflow interrupt is executed.

- **Bits 0 – ICF0: Timer/Counter0, Input Capture Flag**

This flag is set when a capture event occurs on the ICP0 pin. When the input capture register (ICR0) is set to be used as the TOP value, the ICF0 flag is set when the counter reaches the TOP value.

ICF0 is automatically cleared when the input capture interrupt vector is executed. Alternatively, ICF0 can be cleared by writing a logic one to its bit location.

19. ADC – Analog to Digital Converter

19.1 Features

- 10-bit resolution
- 1.0 LSB integral non-linearity
- ±2 LSB absolute accuracy
- 65 - 260µs conversion time
- Up to 15kSPS 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 cancel
- Unipolar/bipolar input mode
- Input polarity reversal mode

19.2 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 19-1 on page 133.

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 ADMUX control register.

19.6 Changing Channel or Reference Selection

The MUX5:0 and REFS2:0 bits in the ADMUX register are single buffered through a temporary register to which the CPU has random access. This ensures that the channels and reference selection only takes place at a safe point during the conversion. The channel and reference selection is continuously updated until a conversion is started. Once the conversion starts, the channel and reference selection is locked to ensure a sufficient sampling time for the ADC. Continuous updating resumes in the last ADC clock cycle before the conversion completes (ADIF in ADCSRA is set).

Note that the conversion starts on the following rising ADC clock edge after ADSC is written. The user is thus advised not to write new channel or reference selection values to ADMUX until one ADC clock cycle after ADSC is written.

If auto triggering is used, the exact time of the triggering event can be indeterministic. Special care must be taken when updating the ADMUX register, in order to control which conversion will be affected by the new settings.

If both ADATE and ADEN is written to one, an interrupt event can occur at any time. If the ADMUX register is changed in this period, the user cannot tell if the next conversion is based on the old or the new settings. ADMUX can be safely updated in the following ways:

a. When ADATE or ADEN is cleared.
b. During conversion, minimum one ADC clock cycle after the trigger event.
c. After a conversion, before the Interrupt Flag used as trigger source is cleared.

When updating ADMUX in one of these conditions, the new settings will affect the next ADC conversion.

19.6.1 ADC Input Channels

When changing channel selections, the user should observe the following guidelines to ensure that the correct channel is selected:

In single conversion mode, always select the channel before starting the conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the conversion to complete before changing the channel selection.

In free running mode, always select the channel before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the first conversion to complete, and then change the channel selection. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. Subsequent conversions will reflect the new channel selection.

19.6.2 ADC Voltage Reference

The voltage reference for the ADC (V_{REF}) indicates the conversion range for the ADC. Single ended channels that exceed V_{REF} will result in codes close to 0x3FF. V_{REF} can be selected as either V_{CC}, or internal 1.1V/2.56V voltage reference, or external AREF pin. The first ADC conversion result after switching voltage reference source may be inaccurate, and the user is advised to discard this result.

Table 19-5. Input Channel Selections

MUX5..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
000000	ADC0 (PA0)	NA	NA	NA
000001	ADC1 (PA1)			
000010	ADC2 (PA2)			
000011	ADC3 (PA4)			
000100	ADC4 (PA5)			
000101	ADC5 (PA6)			
000110	ADC6 (PA7)			
000111	ADC7 (PB4)			
001000	ADC8 (PB5)			
001001	ADC9 (PB6)			
001010	ADC10 (PB7)			
001011	NA	ADC0 (PA0)	ADC1 (PA1)	20x
001100		ADC0 (PA0)	ADC1 (PA1)	1x
001101		ADC1 (PA1)	ADC1 (PA1)	20x
001110		ADC2 (PA2)	ADC1 (PA1)	20x
001111		ADC2 (PA2)	ADC1 (PA1)	1x
010000	N/A	ADC2 (PA2)	ADC3 (PA4)	1x
010001		ADC3 (PA4)	ADC3 (PA4)	20x
010010		ADC4 (PA5)	ADC3 (PA4)	20x
010011		ADC4 (PA5)	ADC3 (PA4)	1x
010100	NA	ADC4 (PA5)	ADC5 (PA6)	20x
010101		ADC4 (PA5)	ADC5 (PA6)	1x
010110		ADC5 (PA6)	ADC5 (PA6)	20x
010111		ADC6 (PA7)	ADC5 (PA6)	20x
011000		ADC6 (PA7)	ADC5 (PA6)	1x
011001	NA	ADC8 (PB5)	ADC9 (PB6)	20x
011010		ADC8 (PB5)	ADC9 (PB6)	1x
011011		ADC9 (PB6)	ADC9 (PB6)	20x
011100		ADC10 (PB7)	ADC9 (PB6)	20x
011101		ADC10 (PB7)	ADC9 (PB6)	1x
011110	1.1V	N/A	N/A	N/A
011111	0V			
100000	N/A	ADC0(PA0)	ADC1(PA1)	20x/32x
100001		ADC0(PA0)	ADC1(PA1)	1x/8x
100010		ADC1(PA1	ADC0(PA0)	20x/32x
100011		ADC1(PA1)	ADC0(PA0)	1x/8x
100100	N/A	ADC1(PA1)	ADC2(PA2)	20x/32x
100101		ADC1(PA1)	ADC2(PA2)	1x/8x
100110		ADC2(PA2	ADC1(PA1)	20x/32x
100111		ADC2(PA2)	ADC1(PA1)	1x/8x

Note: 1. For temperature sensor

19.10.3.2ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
0x05 (0x25)	**ADC9**	**ADC8**	**ADC7**	**ADC6**	**ADC5**	**ADC4**	**ADC3**	**ADC2**	**ADCH**
0x04 (0x24)	**ADC1**	**ADC0**	**–**	**–**	**–**	**–**	**–**	**–**	**ADCL**
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers.

When ADCL is read, the ADC data register is not updated until ADCH is read. Consequently, 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.

The ADLAR bit in ADMUX, and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set, the result is left adjusted. If ADLAR is cleared (default), the result is right adjusted.

- **ADC9:0: ADC Conversion Result**

These bits represent the result from the conversion, as detailed in Section 19.8 “ADC Conversion Result” on page 142.

19.10.4 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	**BIN**	**GSEL**	**-**	**REFS2**	**MUX5**	**ADTS2**	**ADTS1**	**ADTS0**	**ADCSRB**
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7– BIN: Bipolar Input Mode**

The gain stage is working in the unipolar mode as default, but the bipolar mode can be selected by writing the BIN bit in the ADCSRB register. In the unipolar mode only one-sided conversions are supported and the voltage on the positive input must always be larger than the voltage on the negative input. Otherwise the result is saturated to the voltage reference. In the bipolar mode two-sided conversions are supported and the result is represented in the two’s complement form. In the unipolar mode the resolution is 10 bits and the bipolar mode the resolution is 9 bits + 1 sign bit.

- **Bits 6 – GSEL: Gain Select**

The gain select bit selects the 32x gain instead of the 20x gain and the 8x gain instead of the 1x gain when the gain select bit is written to one.

- **Bits 5 – Res: Reserved Bit**

This bit is a reserved bit in the ATtiny261/461/861 and will always read as zero.

- **Bits 4 – REFS2: Reference Selection Bit**

These bit selects either the voltage reference of 1.1 V or 2.56 V for the ADC, as shown in Table 19-4. If active channels are used, using AVCC or an external AREF higher than (AVCC - 1V) is not recommended, as this will affect ADC accuracy. The internal voltage reference options may not be used if an external voltage is being applied to the AREF pin.

- **Bits 3 – MUX5: Analog Channel and Gain Selection Bit 5**

The MUX5 bit is the MSB of the analog channel and gain selection bits. Refer to Table 19-5 for details. If this bit is changed during a conversion, the change will not go into effect until this conversion is complete (ADIF in ADCSRA is set).

20.4 Software Break Points

debugWIRE supports program memory break points by the AVR® break instruction. Setting a break point in AVR Studio® will insert a BREAK instruction in the program memory. The instruction replaced by the BREAK instruction will be stored. When program execution is continued, the stored instruction will be executed before continuing from the program memory. A break can be inserted manually by putting the BREAK instruction in the program.

The flash must be re-programmed each time a break point is changed. This is automatically handled by AVR Studio through the debugWIRE interface. The use of break points will therefore reduce the flash data retention. Devices used for debugging purposes should not be shipped to end customers.

20.5 Limitations of debugWIRE

The debugWIRE communication pin (dW) is physically located on the same pin as external reset (RESET). An external reset source is therefore not supported when the debugWIRE is enabled.

The debugWIRE system accurately emulates all I/O functions when running at full speed, i.e., when the program in the CPU is running. When the CPU is stopped, care must be taken while accessing some of the I/O registers via the debugger (AVR Studio).

A programmed DWEN fuse enables some parts of the clock system to be running in all sleep modes. This will increase the power consumption while in sleep. Thus, the DWEN fuse should be disabled when debugWire is not used.

20.6 Register Description

The following section describes the registers used with the debugWire.

20.6.1 DWDR – debugWire Data Register

Bit	7	6	5	4	3	2	1	0	
0x20 (0x40)	**DWDR[7:0]**								**DWDR**
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 DWDR register provides a communication channel from the running program in the MCU to the debugger. This register is only accessible by the debugWIRE and can therefore not be used as a general purpose register in the normal operations.

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 RFLB and SPMEN bits in SPMCSR. When an LPM instruction is executed within three cycles after the RFLB 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 22-5 on page 158 for a detailed description and mapping of the fuse low byte.

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

Similarly, when reading the fuse high byte, load 0x0003 in the Z-pointer. When an LPM instruction is executed within three cycles after the RFLB 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 22-4 on page 157 for detailed description and mapping of the fuse high byte.

Bit	7	6	5	4	3	2	1	0
Rd	**FHB7**	**FHB6**	**FHB5**	**FHB4**	**FHB3**	**FHB2**	**FHB1**	**FHB0**

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

21.4.3 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. 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.
2. Keep the AVR core in power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR register and thus the flash from unintentional writes.

21.4.4 Programming Time for Flash when Using SPM

The calibrated RC oscillator is used to time flash accesses. Table 21-1 shows the typical programming time for flash accesses from the CPU.

Table 21-1. SPM Programming Time[1]

Symbol	Min Programming Time	Max Programming Time
Flash write (page erase, page write, and write lock bits by SPM)	3.7ms	4.5ms

Note: 1. Minimum and maximum programming time is per individual operation.

22. Memory Programming

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

22.1 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 22-2. The lock bits can only be erased to “1” with the chip erase command. The ATtiny261/461/861 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.

Table 22-1. Lock Bit Byte(1)

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

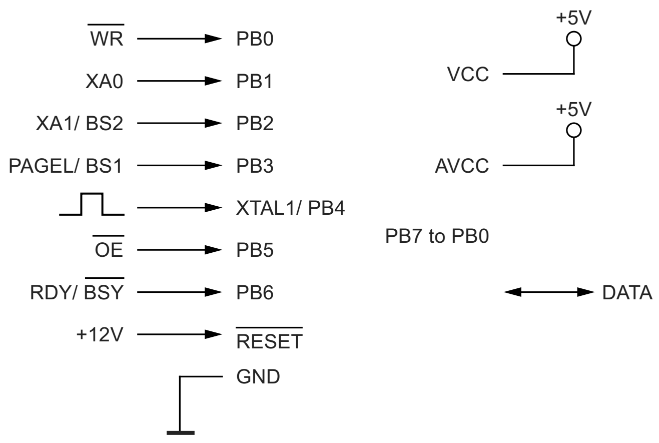
Table 22-2. Lock Bit Protection Modes(1)(2)

Memory Lock Bits			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 the fuse bits before programming the LB1 and LB2.

2. “1” means unprogrammed, “0” means programmed

Figure 22-1. Parallel Programming



$\overline{\text{WR}}$ → PB0
XA0 → PB1
XA1/ BS2 → PB2
PAGEL/ BS1 → PB3
XTAL1/ PB4
$\overline{\text{OE}}$ → PB5
RDY/ $\overline{\text{BSY}}$ → PB6
+12V → $\overline{\text{RESET}}$
GND
+5V
VCC
+5V
AVCC
PB7 to PB0
DATA

Table 22-9. Pin Name Mapping

Signal Name in Programming Mode	Pin Name	I/O	Function
$\overline{\text{WR}}$	PB0	I	Write pulse (active low).
XA0	PB1	I	XTAL action bit 0
XA1/BS2	PB2	I	XTAL action bit 1. Byte select 2 (“0” selects low byte, “1” selects 2’nd high byte).
PAGEL/BS1	PB3	I	Byte select 1 (“0” selects low byte, “1” selects high byte). Program memory and EEPROM data page load.
$\overline{\text{OE}}$	PB5	I	Output enable (active low).
RDY/$\overline{\text{BSY}}$	PB6	O	0: Device is busy programming, 1: Device is ready for new command.
DATA I/O	PA7-PA0	I/O	Bi-directional data bus (output when $\overline{\text{OE}}$ is low).

Table 22-10. Pin Values Used to Enter Programming Mode

Pin	Symbol	Value
PAGEL/BS1	Prog_enable[3]	0
XA1/BS2	Prog_enable[2]	0
XA0	Prog_enable[1]	0
WR	Prog_enable[0]	0

Table 22-11. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load flash or EEPROM address (high or low address byte determined by BS1).
0	1	Load data (high or low data byte for flash determined by BS1).
1	0	Load command
1	1	No action, idle

Figure 24-10. I/O Pin Output Voltage versus Source Current (V_{CC} = 3V)

Figure 24-11. I/O Pin Output Voltage versus Source Current (V_{CC} = 5V)

24.7 Pin Threshold and Hysteresis

Figure 24-12. I/O Pin Input Threshold Voltage versus V_{CC} (V_{IH}, I/O Pin Read as '1')