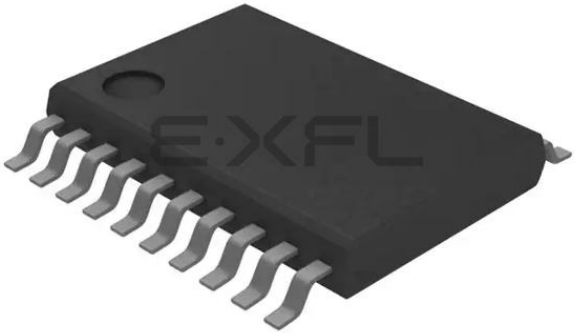


Welcome to E-XFL.COM

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	2KB (1K x 16)
Program Memory Type	FLASH
EEPROM Size	128 x 8
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/attiny261-15xz

The AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the arithmetic logic unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The Atmel® ATtiny261/461/861 provides the following features: 2/4/8Kbyte of in-system programmable flash, 128/256/512 bytes EEPROM, 128/256/512 bytes SRAM, 6 general purpose I/O lines, 32 general purpose working registers, one 8-bit Timer/Counter with compare modes, one 8-bit high speed Timer/Counter, universal serial interface, internal and external interrupts, a 4-channel, 10-bit ADC, a programmable watchdog timer with internal oscillator, and three software selectable power saving modes. The idle mode stops the CPU while allowing the SRAM, Timer/Counter, ADC, analog comparator, and interrupt system to continue functioning. The power-down mode saves the register contents, disabling all chip functions until the next interrupt or hardware reset. The ADC noise reduction mode stops the CPU and all I/O modules except ADC, to minimize switching noise during ADC conversions.

The device is manufactured using Atmel high density non-volatile memory technology. The on-chip ISP flash allows the program memory to be re-programmed in-system through an SPI serial interface, by a conventional non-volatile memory programmer or by an on-chip boot code running on the AVR core.

The Atmel ATtiny261/461/861 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Pin Descriptions

2.2.1 VCC

Supply voltage.

2.2.2 GND

Ground.

2.2.3 AVCC

Analog supply voltage.

2.2.4 AGND

Analog ground.

2.2.5 Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, port A pins that are externally pulled low will source current if the pull-up resistors are activated. The port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the Atmel ATtiny261/461/861 as listed on Section 12.3.2 “Alternate Functions of Port A” on page 62.

2.2.6 Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the Atmel ATtiny261/461/861 as listed on

Section 12.3.1 “Alternate Functions of Port B” on page 59.

2.2.7 $\overline{\text{RESET}}$

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 23-3 on page 174. Shorter pulses are not guaranteed to generate a reset.

6.3 EEPROM Data Memory

The Atmel® ATtiny261/461/861 contains 128/256/512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address registers, the EEPROM data register, and the EEPROM control register. For a detailed description of serial data downloading to the EEPROM, see Section 22.9 “Serial Downloading” on page 167.

6.3.1 EEPROM Read/Write Access

The EEPROM access registers are accessible in the I/O space.

The write access times for the EEPROM are given in Table 6-1 on page 22. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write the EEPROM, some precautions must be taken. In heavily filtered power supplies, V_{CC} is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. See Section 6.3.6 “Preventing EEPROM Corruption” on page 20 for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to Section 6.3.2 “Atomic Byte Programming” on page 18 and Section 6.3.3 “Split Byte Programming” on page 18 for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

6.3.2 Atomic Byte Programming

Using atomic byte programming is the simplest mode. When writing a byte to the EEPROM, the user must write the address into the EEARL register and data into EEDR register. If the EEPMn bits are zero, writing EEPE (within four cycles after EEMPE is written) will trigger the erase/write operation. Both the erase and write cycle are done in one operation and the total programming time is given in Table 1. The EEPE bit remains set until the erase and write operations are completed. While the device is busy with programming, it is not possible to do any other EEPROM operations.

6.3.3 Split Byte Programming

It is possible to split the erase and write cycle in two different operations. This may be useful if the system requires short access time for some limited period of time (typically if the power supply voltage falls). In order to take advantage of this method, it is required that the locations to be written have been erased before the write operation. But since the erase and write operations are split, it is possible to do the erase operations when the system allows doing time-critical operations (typically after power-up).

6.3.4 Erase

To erase a byte, the address must be written to EEAR. If the EEPMn bits are 0b01, writing the EEPE (within four cycles after EEMPE is written) will trigger the erase operation only (programming time is given in Table 1). The EEPE bit remains set until the erase operation completes. While the device is busy programming, it is not possible to do any other EEPROM operations.

6.3.5 Write

To write a location, the user must write the address into EEAR and the data into EEDR. If the EEPMn bits are 0b10, writing the EEPE (within four cycles after EEMPE is written) will trigger the write operation only (programming time is given in Table 1-1 on page 4). The EEPE bit remains set until the write operation completes. If the location to be written has not been erased before write, the data that is stored must be considered as lost. While the device is busy with programming, it is not possible to do any other EEPROM operations.

The calibrated oscillator is used to time the EEPROM accesses. Make sure the oscillator frequency is within the requirements described in Section 7.12.1 “OSCCAL – Oscillator Calibration Register” on page 31.

8.7.4 Internal Voltage Reference

The internal voltage reference will be enabled when needed by the brown-out detection, the analog comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to
Section 9.7 “Internal Voltage Reference” on page 42 for details on the start-up time.

8.7.5 Watchdog Timer

If the watchdog timer is not needed in the application, this module should be turned off. If the watchdog timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to Section 9.8 “Watchdog Timer” on page 42 for details on how to configure the Watchdog Timer.

8.7.6 Port Pins

When entering a sleep mode, all port pins should be configured to use minimum power. The most important thing is then to ensure that no pins drive resistive loads. In sleep modes where both the I/O clock ($clk_{I/O}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to Section 12.2.5 “Digital Input Enable and Sleep Modes” on page 56 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or has an analog signal level close to $V_{CC}/2$, the input buffer will use excessive power.

For analog input pins, the digital input buffer should be disabled at all times. An analog signal level close to $V_{CC}/2$ on an input pin can cause significant current even in active mode. Digital input buffers can be disabled by writing to the digital input disable registers (DIDR0, DIDR1). Refer to Section 19.10.5 “DIDR0 – Digital Input Disable Register 0” on page 149 or Section 19.10.6 “DIDR1 – Digital Input Disable Register 1” on page 149 for details.

8.8 Register Description

8.8.1 MCUCR – MCU Control Register

The MCU control register contains control bits for power management.

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	**–**	**PUD**	**SE**	**SM1**	**SM0**	**—**	**ISC01**	**ISC00**	**MCUCR**
Read/Write	R	R/W	R/W	R/W	R/W	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 5 – SE: Sleep Enable**

The SE bit must be written to logic one to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode unless it is the programmer’s purpose, it is recommended to write the sleep enable (SE) bit to one just before the execution of the SLEEP instruction and to clear it immediately after waking up.

- **Bits 4, 3 – SM1:0: Sleep Mode Select Bits 2..0**

These bits select between the three available sleep modes as shown in Table 8-2 on page 37.

Table 12-4 and Table 12-5 relate the alternate functions of Port B to the overriding signals shown in Figure 12-5 on page 57.

Table 12-4. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/RESET/dW/ ADC10/PCINT15	PB6/ADC9/T0/INT0/ PCINT14	PB5/XTAL2/CLKO/ OC1D/ADC8/PCINT13(1)	PB4/XTAL1/OC1D/ADC7/PCINT12(1)
PUOE	$\overline{\text{RSTDISBL}}$(1) × DWEN(1)	0	$\overline{\text{INTRC}}$ × $\overline{\text{EXTCLK}}$	INTRC
PUOV	1	0	0	0
DDOE	RSTDISBL(1) × DWEN(1)	0	$\overline{\text{INTRC}}$ × $\overline{\text{EXTCLK}}$	INTRC
DDOV	debugWire transmit	0	0	0
PVOE	0	0	OC1D Enable	OC1D enable
PVOV	0	0	OC1D	OC1D
PTOE	0	0	0	0
DIEOE	0	$\overline{\text{RSTDISBL}}$ + (PCINT5 × PCIE + ADC9D)	INTRC × EXTCLK + PCINT4 × PCIE + ADC8D	$\overline{\text{INTRC}}$ + PCINT12 × PCIE + ADC7D
DIEOV	ADC10D	ADC9D	(INTRC × EXTCLK) + ADC8D	INTRC × ADC7D
DI	PCINT15	T0/INT0/PCINT14	PCINT13	PCINT12
AIO	RESET / ADC10	ADC9	XTAL2, ADC8	XTAL1, ADC7

Note: 1. 1 when the fuse is “0” (programmed).

Table 12-5. Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/OC1B/ PCINT11	PB2/SCK/USCK/SCL/ OC1B/PCINT10	PB1/MISO/DO/OC1A/ PCINT9	PB0/MOSI/DI/SDA/ OC1A/PCINT8
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	USI_TWO_WIRE × $\overline{\text{USIPOS}}$	0	USI_TWO_WIRE × $\overline{\text{USIPOS}}$
DDOV	0	(USI_SCL_HOLD + $\overline{\text{PORTB2}}$) × DDB2 × $\overline{\text{USIPOS}}$	0	($\overline{\text{SDA}}$ + $\overline{\text{PORTB0}}$) × DDB0 × $\overline{\text{USIPOS}}$
PVOE	OC1B enable	$\overline{\text{OC1B Enable}}$ + $\overline{\text{USIPOS}}$ × USI_TWO_WIRE × DDB2	OC1A Enable + $\overline{\text{USIPOS}}$ × USI_THREE_WIRE	$\overline{\text{OC1A Enable}}$ + (USI_TWO_WIRE × DDB0 × $\overline{\text{USIPOS}}$)
PVOV	OC1B	OC1B	OC1A + (DO × $\overline{\text{USIPOS}}$)	OC1A
PTOE	0	USITC × $\overline{\text{USIPOS}}$	0	0
DIEOE	PCINT11 × PCIE	PCINT10 × PCIE + USISIE × $\overline{\text{USIPOS}}$	PCINT9 × PCIE	PCINT8 × PCIE + (USISIE × $\overline{\text{USIPOS}}$)
DIEOV	0	0	0	0
DI	PCINT11	USCK/SCL/PCINT10	PCINT9	DI/SDA/PCINT8
AIO				

Note: INTRC means that one of the internal RC oscillators are selected (by the CKSEL fuses), EXTCK means that external clock is selected (by the CKSEL fuses).

15.3.2 TCCR1B – Timer/Counter1 Control Register B

Bit	7	6	5	4	3	2	1	0	
0x2F (0x4F)	**-**	**PSR1**	**DTPS11**	**DTPS10**	**CS13**	**CS12**	**CS11**	**CS10**	**TCCR1B**
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	

- **Bit 7 - Res: Reserved Bit**

- **Bit 6 - PSR1: Prescaler Reset Timer/Counter1**

When this bit is set (one), the Timer/Counter prescaler (TCNT1 is unaffected) will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. This bit will always read as zero.

- **Bits 3:0 - CS13, CS12, CS11, CS10: Clock Select Bits 3, 2, 1, and 0**

The Clock Select bits 3, 2, 1, and 0 define the prescaling source of Timer/Counter1.

Table 15-1. Timer/Counter1 Prescale Select

CS13	CS12	CS11	CS10	Asynchronous Clocking Mode	Synchronous Clocking Mode
0	0	0	0	T/C1 stopped	T/C1 stopped
0	0	0	1	PCK	CK
0	0	1	0	PCK/2	CK/2
0	0	1	1	PCK/4	CK/4
0	1	0	0	PCK/8	CK/8
0	1	0	1	PCK/16	CK/16
0	1	1	0	PCK/32	CK/32
0	1	1	1	PCK/64	CK/64
1	0	0	0	PCK/128	CK/128
1	0	0	1	PCK/256	CK/256
1	0	1	0	PCK/512	CK/512
1	0	1	1	PCK/1024	CK/1024
1	1	0	0	PCK/2048	CK/2048
1	1	0	1	PCK/4096	CK/4096
1	1	1	0	PCK/8192	CK/8192
1	1	1	1	PCK/16384	CK/16384

The stop condition provides a timer enable/disable function.

The following code examples show how to do an atomic read of the TCNT1 register contents. Reading any of the OCR1A/B/C/D registers can be done by using the same principle.

.

Assembly Code Example

```
TIM1_ReadTCNT1:
        ; Save global interrupt flag
        in     r18,SREG
        ; Disable interrupts
        cli
        ; Read TCNT1 into r17:r16
        in     r16,TCNT1
        in     r17,TC1H
        ; Restore global interrupt flag
        out    SREG,r18
        ret
```

C Code Example

```
unsigned int TIM1_ReadTCNT1(void)
{
unsigned char sreg;
unsigned int i;
/* Save global interrupt flag */
sreg = SREG;
/* Disable interrupts */
_CLI();
/* Read TCNT1 into i */
i = TCNT1;
i |= ((unsigned int)TC1H << 8);
/* Restore global interrupt flag
SREG = sreg;
return i;
}
```

Note: 1. The example code assumes that the part specific header file is included.
For I/O registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

16.11.4 TCCR1D – Timer/Counter1 Control Register D

Bit	7	6	5	4	3	2	1	0	
0x26 (0x46)	**FPIE1**	**FPEN1**	**FPNC1**	**FPES1**	**FPAC1**	**FPF1**	**WGM11**	**WGM10**	**TCCR1D**
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	

- **Bit 7 - FPIE1: Fault Protection Interrupt Enable**

Setting this bit (to one) enables the fault protection Interrupt.

- **Bit 6– FPEN1: Fault Protection Mode Enable**

Setting this bit (to one) activates the fault protection mode.

- **Bit 5 – FPNC1: Fault Protection Noise Canceler**

Setting this bit activates the fault protection noise canceler. When the noise canceler is activated, the input from the fault protection pin (INT0) is filtered. The filter function requires four successive equal valued samples of the INT0 pin for changing its output. The fault protection is therefore delayed by four oscillator cycles when the noise canceler is enabled.

- **Bit 4 – FPES1: Fault Protection Edge Select**

This bit selects which edge on the fault protection pin (INT0) is used to trigger a fault event. When the FPES1 bit is written to zero, a falling (negative) edge is used as trigger, and when the FPES1 bit is written to one, a rising (positive) edge will trigger the fault.

- **Bit 3 - FPAC1: Fault Protection Analog Comparator Enable**

When written logic one, this bit enables the fault protection function in Timer/Counter1 to be triggered by the analog comparator. The comparator output is in this case directly connected to the fault protection front-end logic, making the comparator utilize the noise canceler and edge select features of the Timer/Counter1 fault protection interrupt. When written logic zero, no connection between the analog comparator and the fault protection function exists. To make the comparator trigger the Timer/Counter1 fault frotection interrupt, the FPIE1 bit in the Timer/Counter1 control register D (TCCR1D) must be set.

- **Bit 2- FPF1: Fault Protection Interrupt Flag**

When the FPIE1 bit is set (one), the fault protection interrupt is enabled. Activity on the pin will cause an interrupt request even, if the fault protection pin is configured as an output. The corresponding interrupt of fault protection interrupt request is executed from the fault protection interrupt vector. The bit FPF1 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, FPF1 is cleared after a synchronization clock cycle by writing a logical one to the flag. When the SREG I-bit, FPIE1 and FPF1 are set, the Fault Interrupt is executed.

- **Bits 1:0 - WGM11, WGM10: Waveform Generation Mode Bits**

This bit associated with the PWMx bits control the counting sequence of the counter, the source for type of waveform generation to be used, see Table 16-19. Modes of operation supported by the Timer/Counter1 are: Normal mode (counter), fast PWM mode, phase and frequency correct PWM and PWM6 modes.

Table 16-19. Waveform Generation Mode Bit Description

PWM1x	WGM11..10	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	xx	Normal	OCR1C	Immediate	TOP
1	00	Fast PWM	OCR1C	TOP	TOP
1	01	Phase and frequency correct PWM	OCR1C	BOTTOM	BOTTOM
1	10	PWM6 / single-slope	OCR1C	TOP	TOP
1	11	PWM6 / dual-slope	OCR1C	BOTTOM	BOTTOM

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

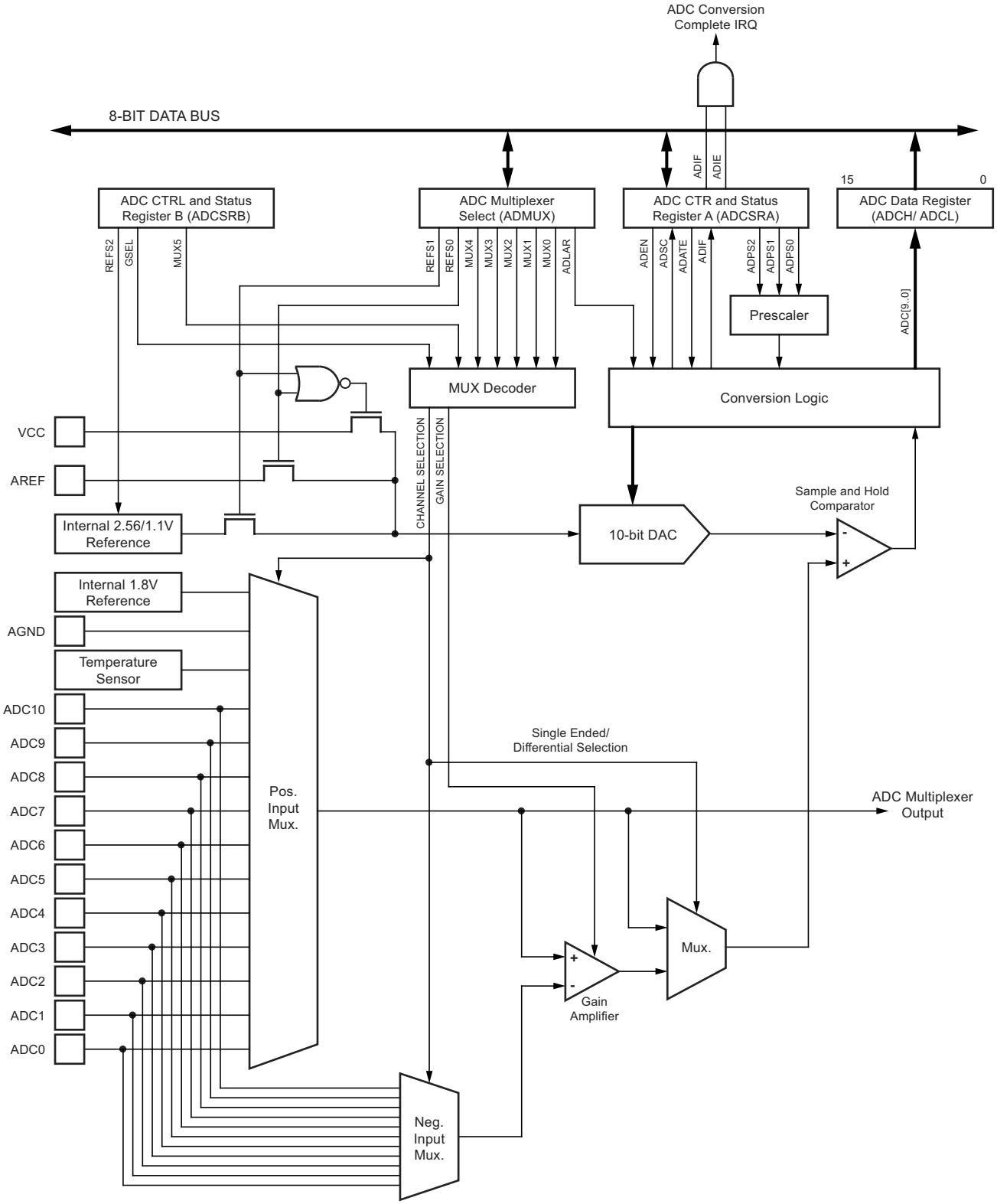
19.2 Overview

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.

Figure 19-1. Analog to Digital Converter Block Schematic



19.7 ADC Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode to reduce noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC Noise Reduction and Idle mode. To make use of this feature, the following procedure should be used:

a. Make sure that the ADC is enabled and is not busy converting. Single conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
b. Enter ADC noise reduction mode (or Idle mode). The ADC will start a conversion once the CPU has been halted.
c. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC conversion complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC conversion complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not be automatically turned off when entering other sleep modes than Idle mode and ADC noise reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption.

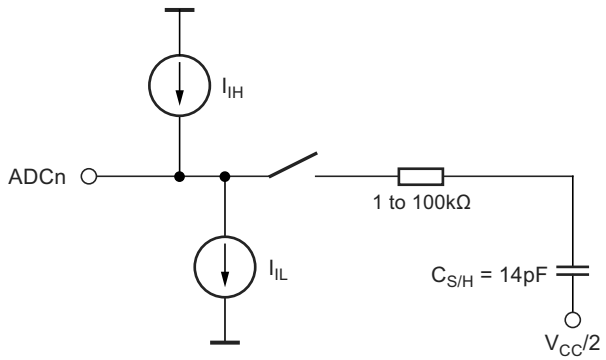
19.7.1 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 19-8 An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long time the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedent sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

Signal components higher than the Nyquist frequency ($f_{ADC}/2$) should not be present to avoid distortion from unpredictable signal convolution. The user is advised to remove high frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 19-8. Analog Input Circuitry



19.7.2 Analog Noise Canceling Techniques

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

a. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.
b. Use the ADC noise canceler function to reduce induced noise from the CPU.
c. If any port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

19.8 ADC Conversion Result

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC result registers (ADCL, ADCH). The form of the conversion result depends on the type of the conversion as there are three types of conversions: single ended conversion, unipolar differential conversion and bipolar differential conversion.

19.8.1 Single Ended Conversion

For single ended conversion, the result is

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference (see Table 19-4 on page 144 and Table 19-5 on page 145). 0x000 represents analog ground, and 0x3FF represents the selected voltage reference minus one LSB. The result is presented in one-sided form, from 0x3FF to 0x000.

19.8.2 Unipolar Differential Conversion

If differential channels and an unipolar input mode are used, the result is

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot 1024}{V_{REF}} \cdot GAIN$$

where V_{POS} is the voltage on the positive input pin, V_{NEG} the voltage on the negative input pin, and V_{REF} the selected voltage reference (see Table 19-4 on page 144 and Table 19-5 on page 145). The voltage on the positive pin must always be larger than the voltage on the negative pin or otherwise the voltage difference is saturated to zero. The result is presented in one-sided form, from 0x000 (0d) to 0x3FF (+1023d). The GAIN is either 1x, 8x, 20x or 32x.

19.8.3 Bipolar Differential Conversion

As default the ADC converter operates in the unipolar input mode, but the bipolar input mode can be selected by writing the BIN bit in the ADCSRB to one. In the bipolar input mode two-sided voltage differences are allowed and thus the voltage on the negative input pin can also be larger than the voltage on the positive input pin. If differential channels and a bipolar input mode are used, the result is

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot 512}{V_{REF}} \cdot GAIN$$

where V_{POS} is the voltage on the positive input pin, V_{NEG} the voltage on the negative input pin, and V_{REF} the selected voltage reference. The result is presented in two's complement form, from 0x200 (–512d) through 0x000 (+0d) to 0x1FF (+511d). The GAIN is either 1x, 8x, 20x or 32x.

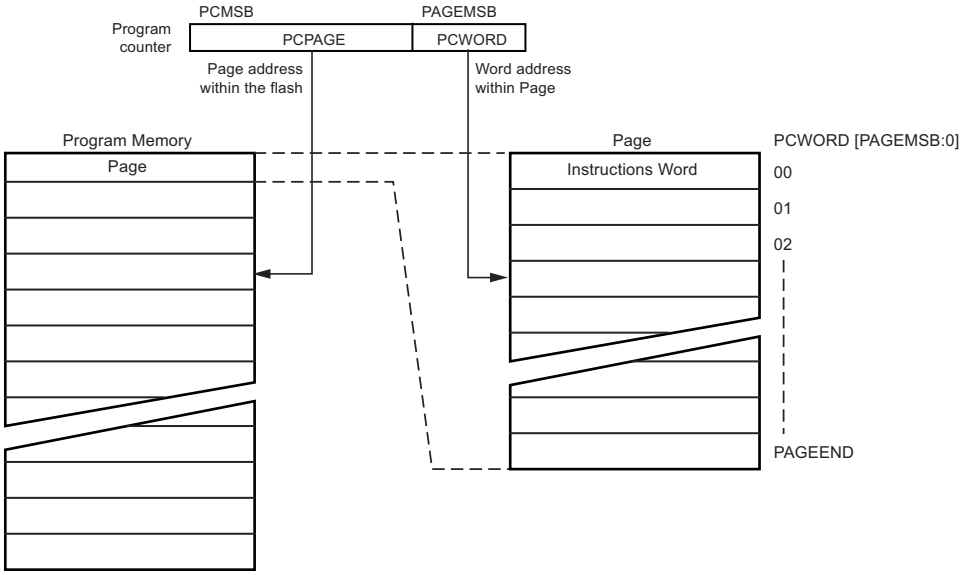
However, if the signal is not bipolar by nature (9 bits + sign as the 10th bit), this scheme loses one bit of the converter dynamic range. Then, if the user wants to perform the conversion with the maximum dynamic range, the user can perform a quick polarity check of the result and use the unipolar differential conversion with selectable differential input pair. When the polarity check is performed, it is sufficient to read the MSB of the result (ADC9 in ADCH). If the bit is one, the result is negative, and if this bit is zero, the result is positive.

19.9 Temperature Measurement

The temperature measurement is based on an on-chip temperature sensor that is coupled to a single ended ADC input. MUX[4..0] bits in ADMUX register enables the temperature sensor. The internal 1.1V voltage reference must also be selected for the ADC voltage reference source in the temperature sensor measurement. When the temperature sensor is enabled, the ADC converter can be used in single conversion mode to measure the voltage over the temperature sensor.

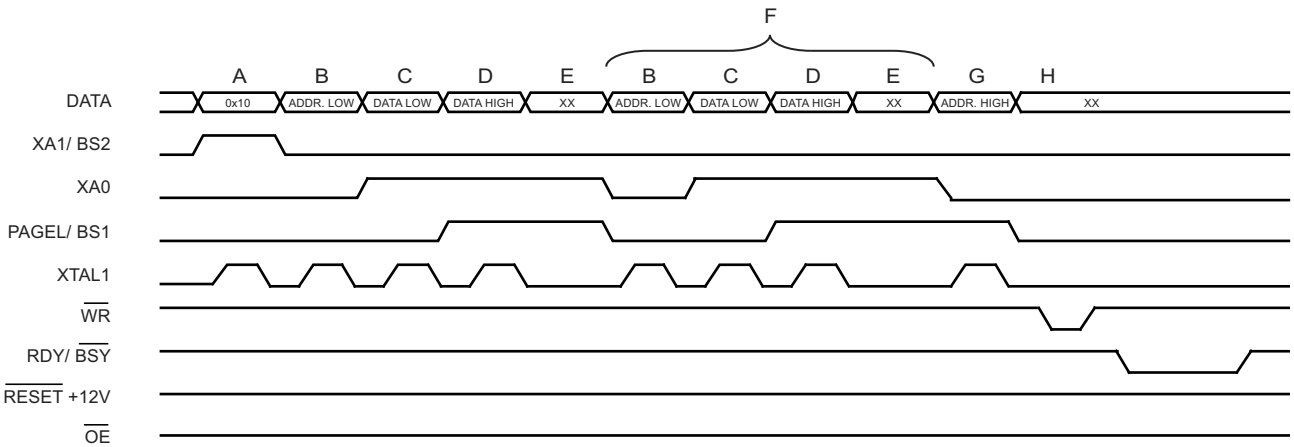
The measured voltage has a linear relationship to the temperature as described in Table 19-2. The voltage sensitivity is approximately 1LSB/°C and the accuracy of the temperature measurement is ±10°C using manufacturing calibration values (TS_GAIN, TS_OFFSET).

Figure 22-2. Addressing the Flash Which is Organized in Pages(1)



Note: 1. PCPAGE and PCWORD are listed in Table 22-7 on page 159.

Figure 22-3. Programming the Flash Waveforms(1)



Note: 1. “XX” is don’t care. The letters refer to the programming description above.

22.8.5 Programming the EEPROM

The EEPROM is organized in pages, see Table 22-8 on page 159. When programming the EEPROM, the program data is latched into a page buffer. This allows one page of data to be programmed simultaneously. The programming algorithm for the EEPROM data memory is as follows (refer to Section 22.8.4 “Programming the Flash” on page 162 for details on command, address and data loading):

1. A: Load command “0001 0001”.
2. G: Load address high byte (0x00 - 0xFF).
3. B: Load address low byte (0x00 - 0xFF).
4. C: Load data (0x00 - 0xFF).
5. E: Latch data (give PAGEL a positive pulse).

22.8.9 Programming the Fuse High Bits

The algorithm for programming the fuse high bits is as follows (refer to Section 22.8.4 “Programming the Flash” on page 162 for details on command and data loading):

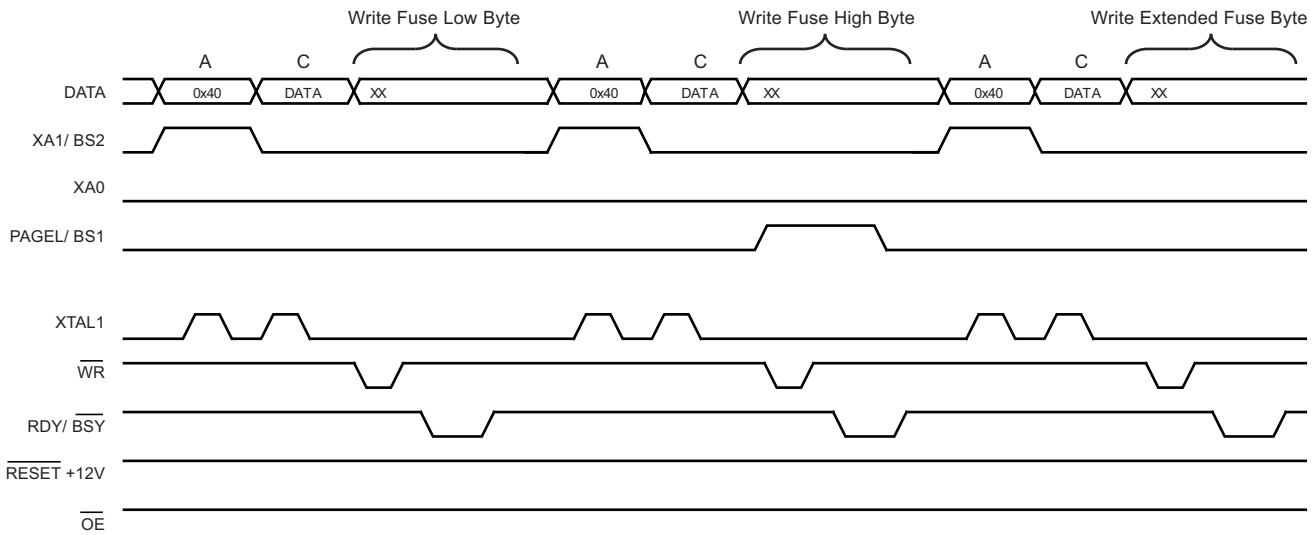
1. A: Load command “0100 0000”.
2. C: Load data low byte. Bit n = “0” programs and bit n = “1” erases the fuse bit.
3. Set BS1 to “1” and BS2 to “0”. This selects high data byte.
4. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.
5. Set BS1 to “0”. This selects low data byte.

22.8.10 Programming the Extended Fuse Bits

The algorithm for programming the extended fuse bits is as follows (refer to Section 22.8.4 “Programming the Flash” on page 162 for details on command and data loading):

1. 1. A: Load command “0100 0000”.
2. 2. C: Load data low byte. Bit n = “0” programs and bit n = “1” erases the fuse bit.
3. 3. Set BS1 to “0” and BS2 to “1”. This selects extended data byte.
4. 4. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.
5. 5. Set BS2 to “0”. This selects low data byte.

Figure 22-5. Programming the FUSES Waveforms



22.8.11 Programming the Lock Bits

The algorithm for programming the Lock bits is as follows (refer to Section 22.8.4 “Programming the Flash” on page 162 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.

22.9.1 Serial Programming Algorithm

When writing serial data to the ATtiny261/461/861, data is clocked on the rising edge of SCK.

When reading data from the ATtiny261/461/861, data is clocked on the falling edge of SCK. See Figure 23-6 on page 179 and Figure 23-7 on page 179 for timing details.

To program and verify the ATtiny261/461/861 in the serial programming mode, the following sequence is recommended (see four byte instruction formats in Table 22-15 on page 169):

1. Power-up sequence:
   Apply power between V_{CC} and GND while $\overline{\text{RESET}}$ and SCK are set to “0”. In some systems, the programmer can not guarantee that SCK is held low during power-up. In this case, $\overline{\text{RESET}}$ must be given a positive pulse of at least two CPU clock cycles duration after SCK has been set to “0”.
2. Wait for at least 20ms and enable serial programming by sending the programming enable serial instruction to pin MOSI.
3. The serial programming instructions will not work if the communication is out of synchronization. When in sync. the second byte (0x53), will echo back when issuing the third byte of the programming enable instruction. Whether the echo is correct or not, all four bytes of the instruction must be transmitted. If the 0x53 did not echo back, give $\overline{\text{RESET}}$ a positive pulse and issue a new programming enable command.
4. The flash is programmed one page at a time. The memory page is loaded one byte at a time by supplying the 5 LSB of the address and data together with the load program memory page instruction. To ensure correct loading of the page, the data low byte must be loaded before data high byte is applied for a given address. The program memory page is stored by loading the write program memory page instruction with the 6 MSB of the address. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_FLASH} before issuing the next page (see Table 22-14). Accessing the serial programming interface before the flash write operation completes can result in incorrect programming.
5. **A:** The EEPROM array is programmed one byte at a time by supplying the address and data together with the appropriate Write instruction. An EEPROM memory location is first automatically erased before new data is written. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_EEPROM} before issuing the next byte (see Table 22-14). In a chip erased device, no 0xFFs in the data file(s) need to be programmed.
   B: The EEPROM array is programmed one page at a time. The Memory page is loaded one byte at a time by supplying the 2 LSB of the address and data together with the load EEPROM memory page instruction. The EEPROM memory page is stored by loading the write EEPROM memory page instruction with the 6 MSB of the address. When using EEPROM page access only byte locations loaded with the load EEPROM memory page instruction is altered. The remaining locations remain unchanged. If polling (RDY/$\overline{\text{BSY}}$) is not used, the used must wait at least t_{WD_EEPROM} before issuing the next page (See Table 22-8). In a chip erased device, no 0xFF in the data file(s) need to be programmed.
6. Any memory location can be verified by using the read instruction which returns the content at the selected address at serial output MISO.
7. At the end of the programming session, $\overline{\text{RESET}}$ can be set high to commence normal operation.
8. Power-off sequence (if needed):
   Set $\overline{\text{RESET}}$ to “1”.
   Turn V_{CC} power off.

Table 22-14. Minimum Wait Delay Before Writing the Next Flash or EEPROM Location

Symbol	Minimum Wait Delay
t_{WD_FLASH}	4.5ms
t_{WD_EEPROM}	4.0ms
t_{WD_ERASE}	4.0ms
t_{WD_FUSE}	4.5ms

22.9.2 Serial Programming Instruction set

Table 22-15 and Figure 22-8 on page 170 describes the Instruction set.

Table 22-15. Serial Programming Instruction Set

Instruction/Operation	Instruction Format			
	Byte 1	Byte 2	Byte 3	Byte4
Programming enable	$AC	$53	$00	$00
Chip erase (program memory/EEPROM)	$AC	$80	$00	$00
Poll RDY/$\overline{BSY}$	$F0	$00	$00	data byte out
Load Instructions				
Load extended address byte(1)	$4D	$00	Extended adr	$00
Load program memory page, high byte	$48	adr MSB	adr LSB	high data byte in
Load program memory page, low byte	$40	adr MSB	adr LSB	low data byte in
Load EEPROM memory page (page access)	$C1	$00	0000 000aa	data byte in
Read Instructions				
Read program memory, high byte	$28	adr MSB	adr LSB	high data byte out
Read program memory, low byte	$20	adr MSB	adr LSB	low data byte out
Read EEPROM memory	$A0	$00	00aa aaaa	data byte out
Read lock bits	$58	$00	$00	data byte out
Read signature byte	$30	$00	0000 000aa	data byte out
Read fuse bits	$50	$00	$00	data byte out
Read fuse high bits	$58	$08	$00	data byte out
Read extended fuse bits	$50	$08	$00	data byte out
Read calibration byte	$38	$00	$00	data byte out
Write Instructions(6)				
Write program memory page	$4C	adr MSB	adr LSB	$00
Write EEPROM memory	$C0	$00	00aa aaaa	data byte in
Write EEPROM memory page (page access)	$C2	$00	00aa aa00	$00
Write lock bits	$AC	$E0	$00	data byte in
Write fuse bits	$AC	$A0	$00	data byte in
Write fuse high bits	$AC	$A8	$00	data byte in
Write extended fuse bits	$AC	$A4	$00	data byte in

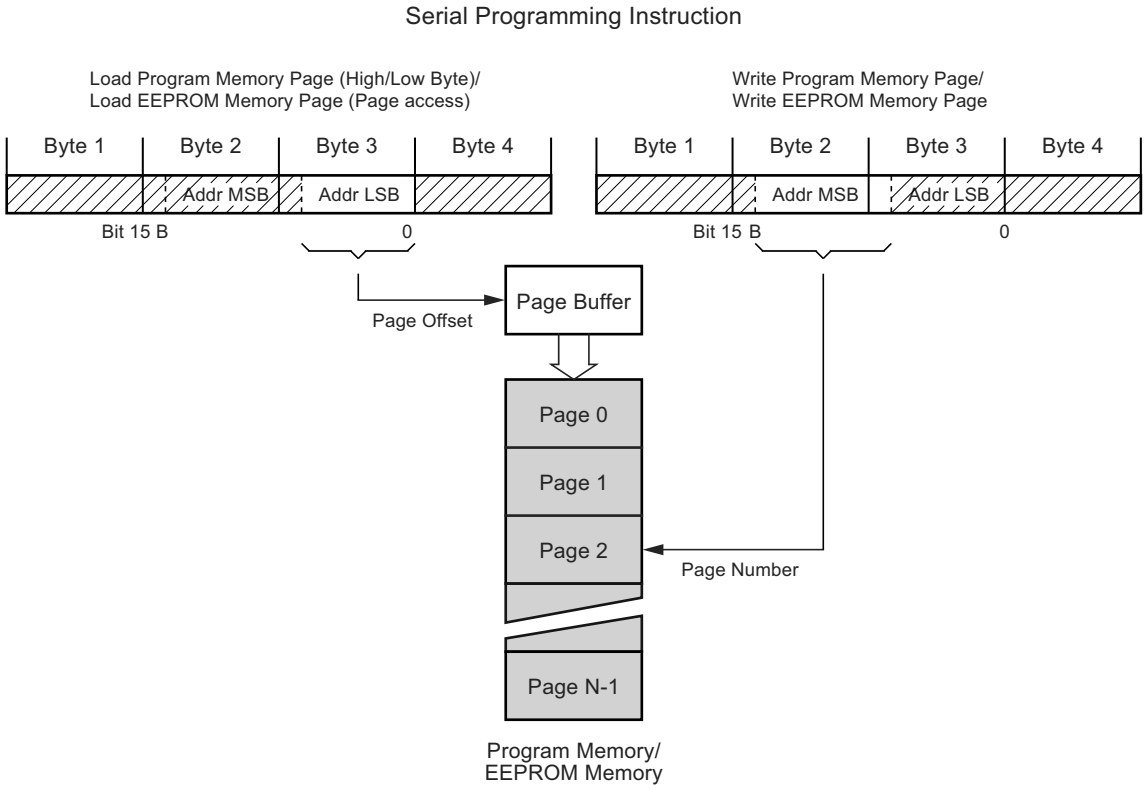
Notes:
1. Not all instructions are applicable for all parts.
2. a = address
3. Bits are programmed ‘0’, unprogrammed ‘1’.
4. To ensure future compatibility, unused Fuses and Lock bits should be unprogrammed (‘1’).
5. Refer to the corresponding section for fuse and lock bits, calibration and signature bytes and page size.
6. Instructions accessing program memory use a word address. This address may be random within the page range.
7. See http://www.atmel.com/avr for application notes regarding programming and programmers.

If the LSB in RDY/BSY data byte out is ‘1’, a programming operation is still pending. Wait until this bit returns ‘0’ before the next instruction is carried out.

Within the same page, the low data byte must be loaded prior to the high data byte.

After data is loaded to the page buffer, program the EEPROM page, see Figure 22-8 on page 170.

Figure 22-8. Serial Programming Instruction Example



23. Electrical Characteristics

23.1 Absolute Maximum Ratings

Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Parameters	Min.	Typ.	Max.	Unit
Operating temperature	–55		+125	°C
Storage temperature	–65		+150	°C
Voltage on any pin except $\overline{\text{RESET}}$ with respect to ground	–0.5		V_{CC} + 0.5	V
Voltage on $\overline{\text{RESET}}$ with respect to ground	–0.5		+13.0	V
Maximum operating voltage		6.0		V
DC current per I/O pin		40.0		mA
DC current V_{CC} and GND pins		200.0		mA
Injection current at V_{CC} = 0V		±5.0(1)		mA
Injection current at V_{CC} = 5V		±1.0		mA

Notes: 1. Maximum current per port = ±30mA

23.2 DC Characteristics

T_A = –40°C to +125°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)(1)

Parameter	Condition	Symbol	Min.	Typ.	Max.	Unit
Input low-voltage	Except XTAL1 and $\overline{\text{RESET}}$ pin	V_{IL}	–0.5		$0.2V_{CC}$	V
Input high-voltage	Except XTAL1 and $\overline{\text{RESET}}$ pin	V_{IH}	$0.7V_{CC}$(3)		V_{CC} + 0.5	V
Input low-voltage	XTAL1 pin, external clock selected	V_{IL1}	–0.5		$0.1V_{CC}$	V
Input high-voltage	XTAL1 pin, external clock selected	V_{IH1}	$0.8V_{CC}$(3)		V_{CC} + 0.5	V
Input low-voltage	$\overline{\text{RESET}}$ pin	V_{IL2}	–0.5		$0.2V_{CC}$	V
Input high-voltage	$\overline{\text{RESET}}$ pin	V_{IH2}	$0.9V_{CC}$(3)		V_{CC} + 0.5	V
Input low-voltage	$\overline{\text{RESET}}$ pin as I/O	V_{IL3}	–0.5		$0.2V_{CC}$	V
Input high-voltage	$\overline{\text{RESET}}$ pin as I/O	V_{IH3}	$0.7V_{CC}$(3)		V_{CC} + 0.5	V

Notes:
1. All DC characteristics contained in this data sheet are based on simulation and characterization of ATtiny261/461/861 AVR microcontrollers manufactured in a typical process technology. These values are preliminary values representing design targets, and will be updated after characterization of actual automotive silicon.
2. “Max” means the highest value where the pin is guaranteed to be read as low.
3. “Min” means the lowest value where the pin is guaranteed to be read as high.
4. Although each I/O port can sink more than the test conditions (10mA at V_{CC} = 5V, 5mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOL, for all ports, should not exceed 60mA.
   If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
5. Although each I/O port can source more than the test conditions (10mA at V_{CC} = 5V, 5mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   1] The sum of all IOH, for all ports, should not exceed 60mA.
   If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.
6. Values using methods described in Section 8.7 “Minimizing Power Consumption” on page 35. Power reduction is enabled (PRR = 0xFF) and there is no I/O drive.
7. BOD disabled.

Figure 24-2. Active Supply Current versus Frequency (1 - 16MHz)

24.2 Idle Supply Current

Figure 24-3. Idle Supply Current versus Frequency (1 - 16MHz)