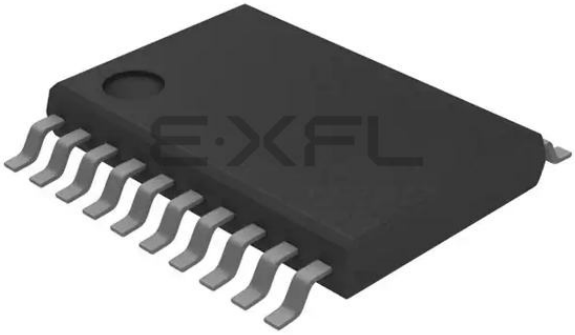


Welcome to E-XFL.COM

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	I²C, LINbus, SPI, UART/USART, USI
Peripherals	Brown-out Detect/Reset, POR, PWM, Temp Sensor, 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)	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/attiny87-a15xz

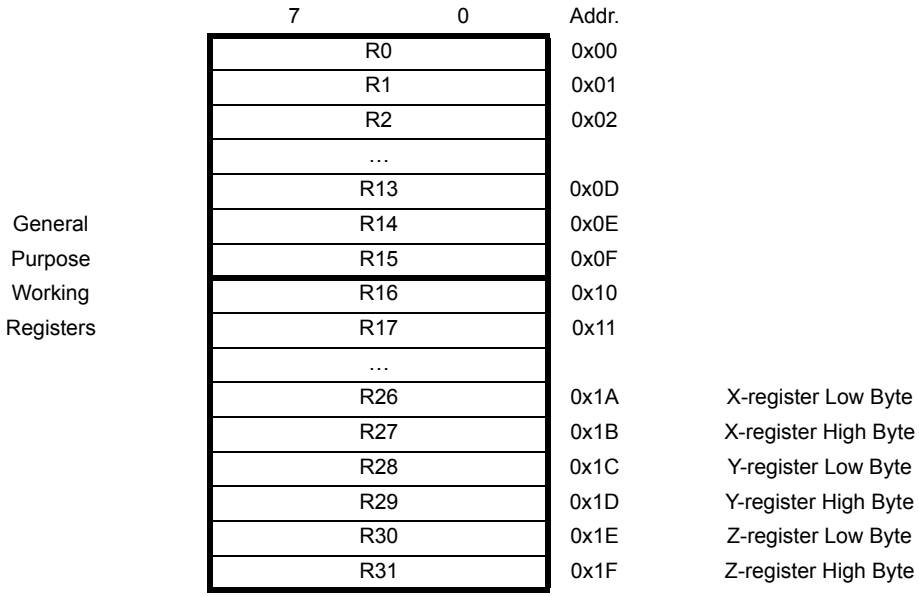
2.4 General Purpose Register File

The register file is optimized for the AVR® enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the register file:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 2-2 shows the structure of the 32 general purpose working registers in the CPU.

Figure 2-2. AVR CPU General Purpose Working Registers



Most of the instructions operating on the register file have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 2-2, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user data space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y- and Z-pointer registers can be set to index any register in the file.

2.4.1 The X-register, Y-register, and Z-register

The registers R26.R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 2-3 on page 12.

7. Interrupts

This section describes the specifics of the interrupt handling as performed in Atmel® ATtiny87/167. For a general explanation of the AVR® interrupt handling, refer to “Reset and Interrupt Handling” on page 13.

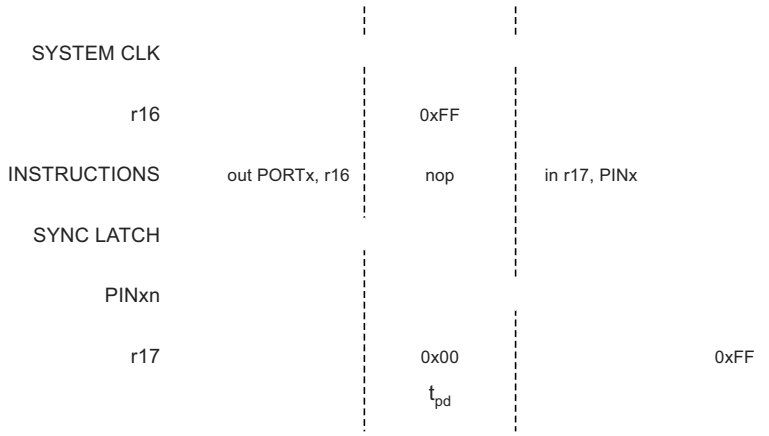
7.1 Interrupt Vectors in ATtiny87/167

Table 7-1. Reset and Interrupt Vectors in ATtiny87/167

Vector Nb.	Program Address		Source	Interrupt Definition
	ATtiny87	ATtiny167		
1	0x0000	0x0000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0001	0x0002	INT0	External Interrupt Request 0
3	0x0002	0x0004	INT1	External Interrupt Request 1
4	0x0003	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0004	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x0005	0x000A	WDT	Watchdog Time-out Interrupt
7	0x0006	0x000C	TIMER1 CAPT	Timer/Counter1 Capture Event
8	0x0007	0x000E	TIMER1 COMPA	Timer/Counter1 Compare Match A
9	0x0008	0x0010	TIMER1 COMPB	Timer/Coutner1 Compare Match B
10	0x0009	0x0012	TIMER1 OVF	Timer/Counter1 Overflow
11	0x000A	0x0014	TIMER0 COMPA	Timer/Counter0 Compare Match A
12	0x000B	0x0016	TIMER0 OVF	Timer/Counter0 Overflow
13	0x000C	0x0018	LIN TC	LIN/UART Transfer Complete
14	0x000D	0x001A	LIN ERR	LIN/UART Error
15	0x000E	0x001C	SPI, STC	SPI Serial Transfer Complete
16	0x000F	0x001E	ADC	ADC Conversion Complete
17	0x0010	0x0020	EE READY	EEPROM Ready
18	0x0011	0x0022	ANALOG COMP	Analog Comparator
19	0x0012	0x0024	USI START	USI Start Condition Detection
20	0x0013	0x0026	USI OVF	USI Counter Overflow

When reading back a software assigned pin value, a nop instruction must be inserted as indicated in Figure 9-5. The out instruction sets the “SYNC LATCH” signal at the positive edge of the clock. In this case, the delay tpd through the synchronizer is 1 system clock period.

Figure 9-5. Synchronization When Reading a Software Assigned Pin Value



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 7 as input with pull-ups assigned to port pins 6 and 7. 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<<PB7)|(1<<PB6)|(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<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* Insert nop for synchronization*/
__no_operation();
/* 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, 6, and 7, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

PCINT2/ADC2/OC0A/DO/MISO – Port A, Bit 2

PCINT2: pin change interrupt, source 2.

ADC2: analog to digital converter, channel 2.

OC0A: output compare match A or output PWM A for timer/counter0. The pin has to be configured as an output (DDA2 set (one)) to serve these functions.

DO: three-wire mode USI data output. Three-wire mode data output overrides PORTA2 and it is driven to the port when the data direction bit DDA2 is set. PORTA2 still enables the pull-up, if the direction is input and PORTA2 is set (one).

MISO: master data input, slave data output pin for SPI channel. When the SPI is enabled as a master, this pin is configured as an input regardless of the setting of DDA2. When the SPI is enabled as a slave, the data direction of this pin is controlled by DDA2. When the pin is forced to be an input, the pull-up can still be controlled by PORTA2.

- **PCINT1/ADC1/TXD/TXLIN – Port A, Bit 1**

PCINT1: pin change interrupt, source 1.

ADC1: analog to digital converter, channel 1.

TXD: UART transmit pin. When the UART transmitter is enabled, this pin is configured as an output regardless the value of DDA1. PORTA1 still enables the pull-up, if the direction is input and PORTA2 is set (one).

TXLIN: LIN transmit pin. When the LIN is enabled, this pin is configured as an output regardless the value of DDA1. PORTA1 still enables the pull-up, if the direction is input and PORTA2 is set (one).

- **PCINT0/ADC0/RXD/RXLIN – Port A, Bit 0**

PCINT0: pin change interrupt, source 0.

ADC0: analog to digital converter, channel 0.

RXD: UART receive pin. When the UART receiver is enabled, this pin is configured as an input regardless of the value of DDA0. When the pin is forced to be an input, a logical one in PORTA0 will turn on the internal pull-up.

RXLIN: LIN receive pin. When the LIN is enabled, this pin is configured as an input regardless of the value of DDA0. When the pin is forced to be an input, a logical one in PORTA0 will turn on the internal pull-up.

10.6.2 Compare Output Mode and Waveform Generation

The waveform generator uses the COM0A1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM0A1:0 = 0 tells the waveform generator that no action on the OC0A register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 10-1 on page 96. For fast PWM mode, refer to Table 10-2 on page 96, and for phase correct PWM refer to Table 10-3 on page 96.

A change of the COM0A1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC0A strobe bits.

10.7 Modes of Operation

The mode of operation, i.e., the behavior of the timer/counter and the output compare pins, is defined by the combination of the waveform generation mode (WGM01:0) and compare output mode (COM0A1:0) bits. The compare output mode bits do not affect the counting sequence, while the waveform generation mode bits do. The COM0A1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM0A1:0 bits control whether the output should be set, cleared, or toggled at a compare match (Section 10.6 “Compare Match Output Unit” on page 87).

For detailed timing information refer to Section 10.8 “Timer/Counter Timing Diagrams” on page 92.

10.7.1 Normal Mode

The simplest mode of operation is the normal mode (WGM01:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the timer/counter overflow flag (TOV0) will be set in the same timer clock cycle as the TCNT0 becomes zero. The TOV0 flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV0 flag, the timer resolution can be increased by software. There are no special cases to consider in the normal mode, a new counter value can be written anytime.

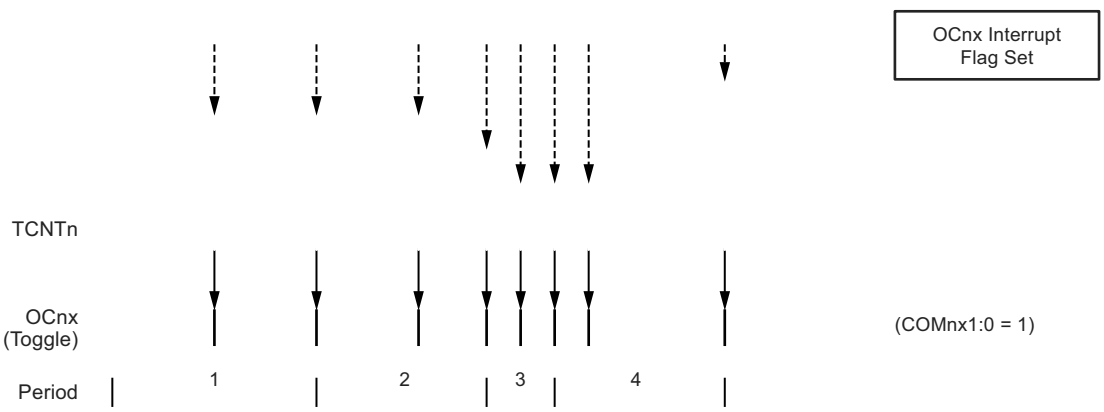
The output compare unit can be used to generate interrupts at some given time. Using the output compare to generate waveforms in normal mode is not recommended, since this will occupy too much of the CPU time.

10.7.2 Clear Timer on Compare Match (CTC) Mode

In clear timer on compare or CTC mode (WGM01:0 = 2), the OCR0A register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT0) matches the OCR0A. The OCR0A defines the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 10-5. The counter value (TCNT0) increases until a compare match occurs between TCNT0 and OCR0A, and then counter (TCNT0) is cleared.

Figure 10-5. CTC Mode, Timing Diagram



10.11.2 Timer/Counter0 Register – TCNT0

Bit	7	6	5	4	3	2	1	0	
	TCNT07	TCNT06	TCNT05	TCNT04	TCNT03	TCNT02	TCNT01	TCNT00	TCNT0
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 timer/counter register gives direct access, both for read and write operations, to the timer/counter unit 8-bit counter. writing to the TCNT0 register blocks (removes) the compare match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a compare match between TCNT0 and the OCR0x register.

10.11.3 Output Compare Register A – OCR0A

Bit	7	6	5	4	3	2	1	0	
	OCR0A7	OCR0A6	OCR0A5	OCR0A4	OCR0A3	OCR0A2	OCR0A1	OCR0A0	OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The output compare register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an output compare interrupt, or to generate a waveform output on the OC0A pin.

10.11.4 Asynchronous Status Register – ASSR

Bit	7	6	5	4	3	2	1	0	
	–	EXCLK	AS0	TCN0UB	OCR0AUB	–	TCR0AUB	TCR0BUB	ASSR
Read/Write	R	R/W	R/W	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: Reserved Bit**

This bit is reserved in the Atmel® ATtiny87/167 and will always read as zero.

- **Bit 6 – EXCLK: Enable External Clock Input**

When EXCLK is written to one, and asynchronous clock is selected, the external clock input buffer is enabled and an external clock can be input on XTAL1 pin instead of an external crystal. Writing to EXCLK should be done before asynchronous operation is selected. Note that the crystal oscillator will only run when this bit is zero.

- **Bit 5 – AS0: Asynchronous Timer/Counter0**

When AS0 is written to zero, timer/counter0 is clocked from the I/O clock, clkI/O and the timer/counter0 acts as a synchronous peripheral.

When AS0 is written to one, timer/counter0 is clocked from the low-frequency crystal oscillator (see Section 4.2.5 “Low-frequency Crystal Oscillator” on page 31) or from external clock on XTAL1 pin (see Section 4.2.6 “External Clock” on page 31) depending on EXCLK setting. When the value of AS0 is changed, the contents of TCNT0, OCR0A, and TCCR0A might be corrupted.

AS0 also acts as a flag: timer/counter0 is clocked from the low-frequency crystal or from external clock ONLY IF the calibrated internal RC oscillator or the internal watchdog oscillator is used to drive the system clock. After setting AS0, if the switching is available, AS0 remains to 1, else it is forced to 0.

- **Bit 4 – TCN0UB: Timer/Counter0 Update Busy**

When timer/counter0 operates asynchronously and TCNT0 is written, this bit becomes set. When TCNT0 has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that TCNT0 is ready to be updated with a new value.

12. 16-bit Timer/Counter1

The 16-bit timer/counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement. The main features are:

12.1 Features

- True 16-bit design (i.e., Allows 16-bit PWM)
- Two independent output compare units
- Four controlled output pins per output compare unit
- Double buffered output compare registers
- One input capture unit
- Input capture noise canceler
- Clear timer on compare match (auto reload)
- Glitch-free, phase correct pulse width modulator (PWM)
- Variable PWM period
- Frequency generator
- External event counter
- Four independent interrupt sources (TOV1, OCF1A, OCF1B, and ICF1)

12.2 Overview

Many register and bit references in this section are written in general form.

- A lower case “n” replaces the timer/counter number, in this case 1. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT1 for accessing timer/counter1 counter value and so on.
- A lower case “x” replaces the output compare unit channel, in this case A or B. However, when using the register or bit defines in a program, the precise form must be used, i.e., OCR1A for accessing timer/counter1 output compare channel A value and so on.
- A lower case “i” replaces the index of the output compare output pin, in this case U, V, W or X. However, when using the register or bit defines in a program, the precise form must be used.

The following code examples show how to do an atomic write of the TCNT1 register contents. Writing any of the OCR1A/B or ICR1 registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_WriteTCNT1:
  ; Save global interrupt flag
  in     r18,SREG
  ; Disable interrupts
  cli
  ; Set TCNT1 to r17:r16
  sts    TCNT1H,r17
  sts    TCNT1L,r16
  ; Restore global interrupt flag
  out    SREG,r18
  ret
```

C Code Example[1]

```
void TIM16_WriteTCNT1(unsigned int i)
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Set TCNT1 to i */
  TCNT1 = i;
  /* Restore global interrupt flag */
  SREG = sreg;
}
```

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

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNT1.

12.3.2 Reusing the Temporary High Byte Register

If writing to more than one 16-bit register where the high byte is the same for all registers written, then the high byte only needs to be written once. However, note that the same rule of atomic operation described previously also applies in this case.

12.4 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the Clock Select logic which is controlled by the Clock Select (CS12:0) bits located in the Timer/Counter control Register B (TCCR1B). For details on clock sources and prescaler, see Section 11. “Timer/Counter1 Prescaler” on page 101.

12.9.1 Normal Mode

The simplest mode of operation is the normal mode (WGM13:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 16-bit value (MAX = 0xFFFF) and then restarts from the BOTTOM (0x0000). In normal operation the timer/counter overflow flag (TOV1) will be set in the same timer clock cycle as the TCNT1 becomes zero. The TOV1 flag in this case behaves like a 17th bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV1 flag, the timer resolution can be increased by software. There are no special cases to consider in the normal mode, a new counter value can be written anytime.

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

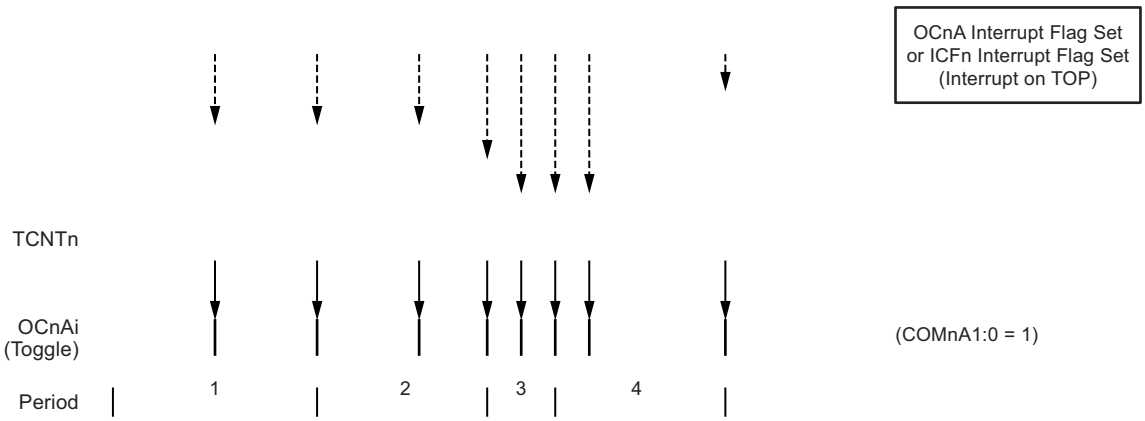
The output compare units can be used to generate interrupts at some given time. Using the output compare to generate waveforms in normal mode is not recommended, since this will occupy too much of the CPU time.

12.9.2 Clear Timer on Compare Match (CTC) Mode

In clear timer on compare or CTC mode (WGM13:0 = 4 or 12), the OCR1A or ICR1 register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT1) matches either the OCR1A (WGM13:0 = 4) or the ICR1 (WGM13:0 = 12). The OCR1A or ICR1 define the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 12-7. The counter value (TCNT1) increases until a compare match occurs with either OCR1A or ICR1, and then counter (TCNT1) is cleared.

Figure 12-7. CTC Mode, Timing Diagram



An interrupt can be generated at each time the counter value reaches the TOP value by either using the OCF1A or ICF1 flag according to the register used to define the TOP value. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing the TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR1A or ICR1 is lower than the current value of TCNT1, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. In many cases this feature is not desirable. An alternative will then be to use the fast PWM mode using OCR1A for defining TOP (WGM13:0 = 15) since the OCR1A then will be double buffered.

For generating a waveform output in CTC mode, the OC1A output can be set to toggle its logical level on each compare match by setting the compare output mode bits to toggle mode (COM1A1:0 = 1). The OC1A value will not be visible on the port pin unless the data direction for the pin is set to output (DDR_OC1A = 1) and OC1Ai is set. The waveform generated will have a maximum frequency of $f_{OC}1_A = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). The waveform frequency is defined by the following equation:

$$f_{OCnA} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

12.11 16-bit Timer/Counter Register Description

12.11.1 Timer/Counter1 Control Register A – TCCR1A

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – COM1A1:0: Compare Output Mode for Channel A**

- **Bit 5:4 – COM1B1:0: Compare Output Mode for Channel B**

The COM1A1:0 and COM1B1:0 control the output compare pins (OC1Ai and OC1Bi respectively) behavior. If one or both of the COM1A1:0 bits are written to one, the OC1Ai output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B1:0 bit are written to one, the OC1Bi output overrides the normal port functionality of the I/O pin it is connected to. However, note that the data direction register (DDR) bit and OC1xi bit (TCCR1D) corresponding to the OC1Ai or OC1Bi pin must be set in order to enable the output driver.

When the OC1Ai or OC1Bi is connected to the pin, the function of the COM1A/B1:0 bits is dependent of the WGM13:0 bits setting. Table 12-1 shows the COM1A/B1:0 bit functionality when the WGM13:0 bits are set to a Normal or a CTC mode (non-PWM).

Table 12-1. Compare Output Mode, non-PWM

OC1Ai OC1Bi	COM1A1 COM1B1	COM1A0 COM1B0	Description
0	x	x	Normal port operation, OC1A/OC1B disconnected.
1	0	0	
	0	1	Toggle OC1A/OC1B on compare match.
	1	0	Clear OC1A/OC1B on compare match (set output to low level).
	1	1	Set OC1A/OC1B on compare match (set output to high level).

Table 12-2 shows the COM1A/B1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 12-2. Compare Output Mode, Fast PWM (1)

OC1Ai OC1Bi	COM1A1 COM1B1	COM1A0 COM1B0	Description
0	x	x	Normal port operation, OC1A/OC1B disconnected.
1	0	0	
1	0	1	WGM13=0: Normal port operation, OC1A/OC1B disconnected. WGM13=1: Toggle OC1A on compare match, OC1B reserved.
1	1	0	Clear OC1A/OC1B on compare match Set OC1A/OC1B at TOP
1	1	1	Set OC1A/OC1B on compare match Clear OC1A/OC1B at TOP

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at TOP. See Section 12.9.3 “Fast PWM Mode” on page 117 for more details.

- **Bits 3:0 – USICNT3..0: Counter Value**

These bits reflect the current 4-bit counter value. The 4-bit counter value can directly be read or written by the CPU.

The 4-bit counter increments by one for each clock generated either by the external clock edge detector, by a timer/counter0 compare match, or by software using USICLK or USITC strobe bits. The clock source depends of the setting of the USICS1..0 bits. For external clock operation a special feature is added that allows the clock to be generated by writing to the USITC strobe bit. This feature is enabled by write a one to the USICLK bit while setting an external clock source (USICS1 = 1).

Note that even when no wire mode is selected (USIWM1..0 = 0) the external clock input (USCK/SCL) are can still be used by the counter.

14.5.4 USICR – USI Control Register

Bit	7	6	5	4	3	2	1	0	
	USISIE	**USIOIE**	**USIWM1**	**USIWM0**	**USICS1**	**USICS0**	**USICLK**	**USITC**	**USICR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	W	W	
Initial Value	0	0	0	0	0	0	0	0	

The control register includes interrupt enable control, wire mode setting, clock select setting, and clock strobe.

- **Bit 7 – USISIE: Start Condition Interrupt Enable**

Setting this bit to one enables the start condition detector interrupt. If there is a pending interrupt when the USISIE and the global interrupt enable flag is set to one, this will immediately be executed. Refer to the USISIF bit description on page 145 for further details.

- **Bit 6 – USIOIE: Counter Overflow Interrupt Enable**

Setting this bit to one enables the counter overflow interrupt. If there is a pending interrupt when the USIOIE and the global interrupt enable flag is set to one, this will immediately be executed. Refer to the USIOIF bit description on page 145 for further details.

- **Bit 5:4 – USIWM1:0: Wire Mode**

These bits set the type of wire mode to be used. Basically only the function of the outputs are affected by these bits. Data and clock inputs are not affected by the mode selected and will always have the same function. The counter and USI data register can therefore be clocked externally, and data input sampled, even when outputs are disabled. The relations between USIWM1:0 and the USI operation is summarized in Table 14-1 on page 147.

Table 14-1. Relations between USIWM1..0 and the USI Operation

USIWM1	USIWM0	Description
0	0	Outputs, clock hold, and start detector disabled. Port pins operates as normal.
0	1	Three-wire mode. Uses DO, DI, and USCK pins. The *data output* (DO) pin overrides the corresponding bit in the PORT register in this mode. However, the corresponding DDR bit still controls the data direction. When the port pin is set as input the pins pull-up is controlled by the PORT bit. The *data input* (DI) and *serial clock* (USCK) pins do not affect the normal port operation. When operating as master, clock pulses are software generated by toggling the PORT register, while the data direction is set to output. The USITC bit in the USICR register can be used for this purpose.
1	0	Two-wire mode. Uses SDA (DI) and SCL (USCK) pins(1). The *serial data* (SDA) and the *serial clock* (SCL) pins are bi-directional and uses open-collector output drives. The output drivers are enabled by setting the corresponding bit for SDA and SCL in the DDR register. When the output driver is enabled for the SDA pin, the output driver will force the line SDA low if the output of the USI data register or the corresponding bit in the PORT register is zero. Otherwise the SDA line will not be driven (i.e., it is released). When the SCL pin output driver is enabled the SCL line will be forced low if the corresponding bit in the PORT register is zero, or by the start detector. Otherwise the SCL line will not be driven. The SCL line is held low when a start detector detects a start condition and the output is enabled. Clearing the start condition flag (USISIF) releases the line. The SDA and SCL pin inputs is not affected by enabling this mode. Pull-ups on the SDA and SCL port pin are disabled in Two-wire mode.
1	1	Two-wire mode. Uses SDA and SCL pins. Same operation as for the two-wire mode described above, except that the SCL line is also held low when a counter overflow occurs, and is held low until the counter overflow flag (USIOIF) is cleared.

Note: 1. The DI and USCK pins are renamed to *serial data* (SDA) and *serial clock* (SCL) respectively to avoid confusion between the modes of operation.

- **Bit 3:2 – USICS1:0: Clock Source Select**

These bits set the clock source for the USI data register and counter. The data output latch ensures that the output is changed at the opposite edge of the sampling of the data input (DI/SDA) when using external clock source (USCK/SCL). When software strobe or timer/counter0 compare match clock option is selected, the output latch is transparent and therefore the output is changed immediately. Clearing the USICS1:0 bits enables software strobe option. When using this option, writing a one to the USICLK bit clocks both the USI data register and the counter. For external clock source (USICS1 = 1), the USICLK bit is no longer used as a strobe, but selects between external clocking and software clocking by the USITC strobe bit.

Table 14-2 on page 148 shows the relationship between the USICS1..0 and USICLK setting and clock source used for the USI data register and the 4-bit counter.

15.5.14 Message Filtering

Message filtering based upon the whole identifier is not implemented. Only a status for frame headers having 0x3C, 0x3D, 0x3E and 0x3F as identifier is available in the LINSIR register.

Table 15-4. Frame Status

LIDST[2..0]	Frame Status
$0xx_b$	No specific identifier
100_b	60 (0x3C) identifier
101_b	61 (0x3D) identifier
110_b	62 (0x3E) identifier
111_b	63 (0x3F) identifier

The LIN protocol says that a message with an identifier from 60 (0x3C) up to 63 (0x3F) uses a classic checksum (sum over the data bytes only). Software will be responsible for switching correctly the LIN13 bit to provide/check this expected checksum (the insertion of the ID field in the computation of the CRC is set - or not - just after entering the Rx or Tx response command).

15.5.15 Data Management

15.5.15.1 LIN FIFO Data Buffer

To preserve register allocation, the LIN data buffer is seen as a FIFO (with address pointer accessible). This FIFO is accessed via the LINDX[2..0] field of LINSEL register through the LINDAT register.

LINDX[2..0], the data index, is the address pointer to the required data byte. The data byte can be read or written. The data index is automatically incremented after each LINDAT access if the $\overline{\text{LAINC}}$ (active low) bit is cleared. A roll-over is implemented, after data index=7 it is data index=0. Otherwise, if $\overline{\text{LAINC}}$ bit is set, the data index needs to be written (updated) before each LINDAT access.

The first byte of a LIN frame is stored at the data index=0, the second one at the data index=1, and so on. Nevertheless, LINSEL must be initialized by the user before use.

15.5.15.2 UART Data Register

The LINDAT register is the data register (no buffering - no FIFO). In write access, LINDAT will be for data out and in read access, LINDAT will be for data in.

In UART mode the LINSEL register is unused.

15.5.16 OCD Support

When a debugger break occurs, the state machine of the LIN/UART controller is stopped (included frame time-out) and further communication may be corrupted.

15.6.3 LIN Enable Interrupt Register - LINENIR

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	LENERR	LENIDOK	LENTXOK	LENRXOK	LINENIR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:4 - Reserved Bits**

These bits are reserved for future use. For compatibility with future devices, they must be written to zero when LINENIR is written.

- **Bit 3 - LENERR: Enable Error Interrupt**
  - 0 = Error interrupt masked,
  - 1 = Error interrupt enabled.

- **Bit 2 - LENIDOK: Enable Identifier Interrupt**
  - 0 = Identifier interrupt masked,
  - 1 = Identifier interrupt enabled.

- **Bit 1 - LENTXOK: Enable Transmit Performed Interrupt**
  - 0 = Transmit performed interrupt masked,
  - 1 = Transmit performed interrupt enabled.

- **Bit 0 - LENRXOK: Enable Receive Performed Interrupt**
  - 0 = Receive performed interrupt masked,
  - 1 = Receive performed interrupt enabled.

15.6.4 LIN Error Register - LINERR

Bit	7	6	5	4	3	2	1	0	
	LABORT	LTOERR	LOVERR	LFERR	LSERR	LPERR	LCERR	LBERR	LINERR
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 - LABORT: Abort Flag**
  - 0 = No warning,
  - 1 = LIN abort command occurred.

This bit is cleared when LERR bit in LINSIR is cleared.

- **Bit 6 - LTOERR: Frame_Time_Out Error Flag**
  - 0 = No error,
  - 1 = Frame_time_out error.

This bit is cleared when LERR bit in LINSIR is cleared.

- **Bit 5 - LOVERR: Overrun Error Flag**
  - 0 = No error,
  - 1 = Overrun error.

This bit is cleared when LERR bit in LINSIR is cleared.

The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 14.5 ADC clock cycles after the start of an first conversion. When a conversion is complete, the result is written to the ADC data registers, and ADIF is set. In single conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

When auto triggering is used, the prescaler is reset when the trigger event occurs. This assures a fixed delay from the trigger event to the start of conversion. In this mode, the sample-and-hold takes place 2 ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic.

In free running mode, a new conversion will be started immediately after the conversion completes, while ADSC remains high. For a summary of conversion times, see Table 17-1 on page 181.

Figure 17-4. ADC Timing Diagram, First Conversion (Single Conversion Mode)

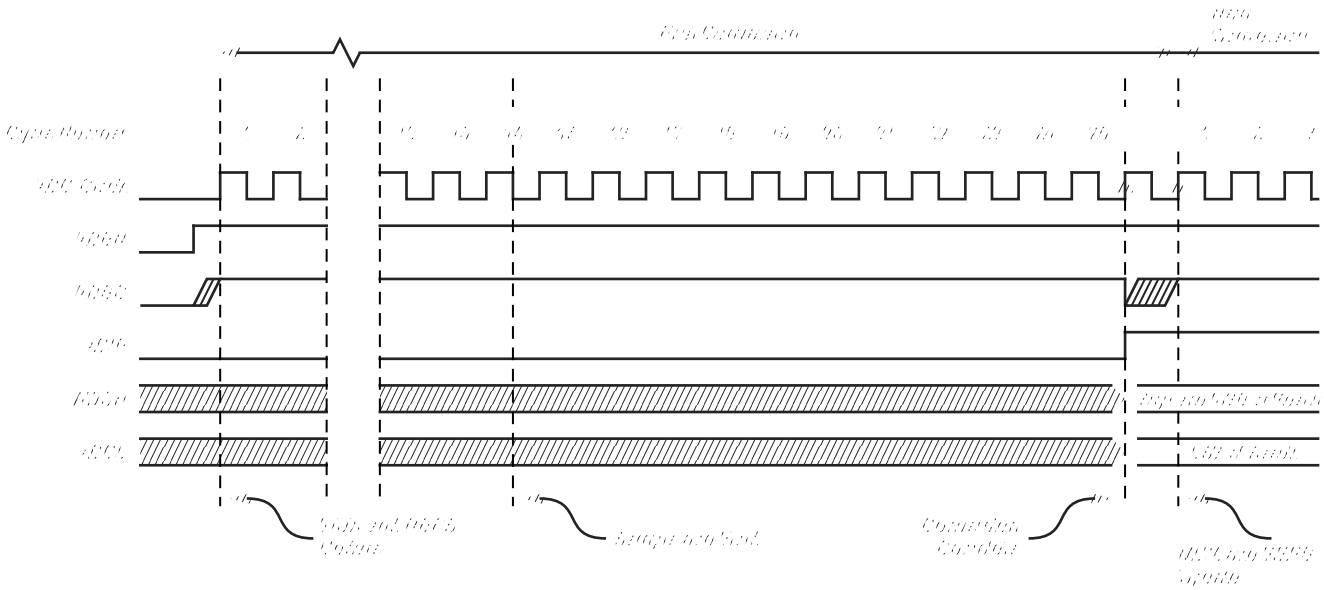
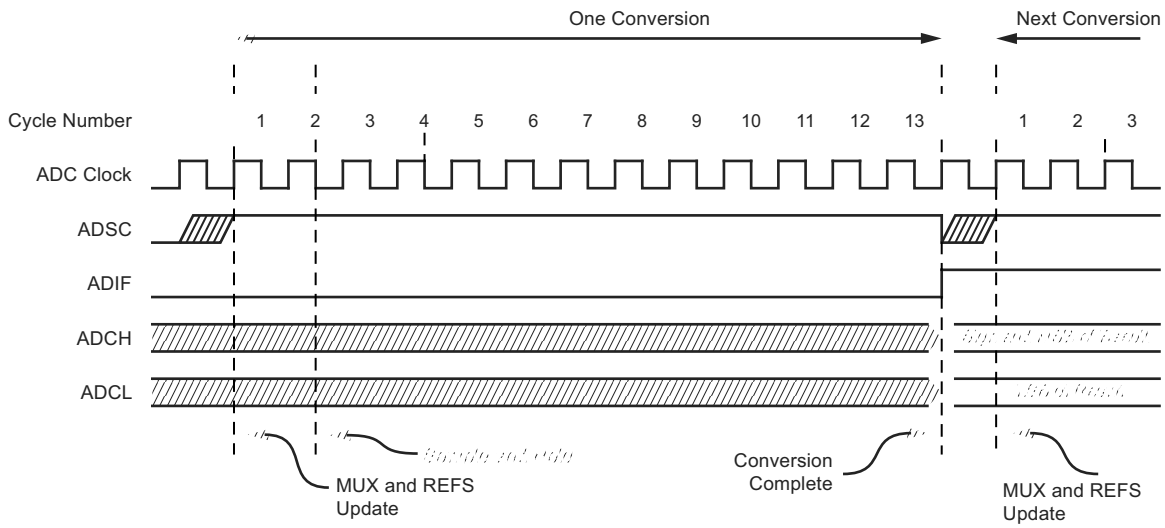
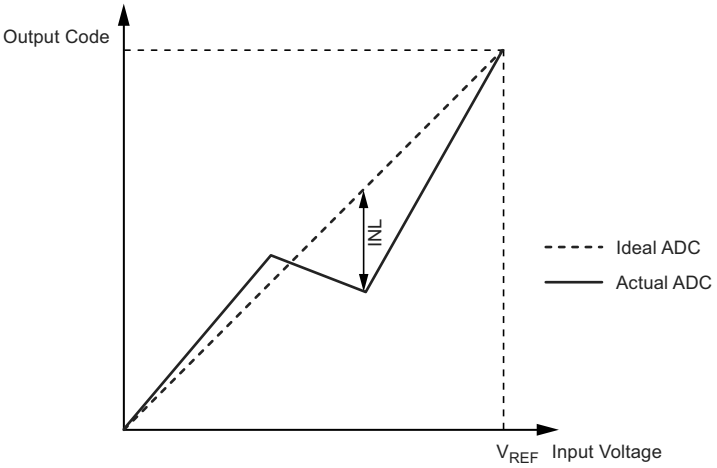


Figure 17-5. ADC Timing Diagram, Single Conversion



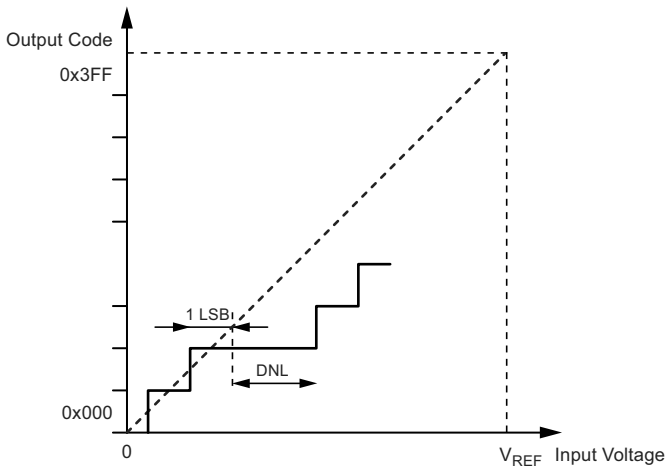
- Integral non-linearity (INL): After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSB.

Figure 17-11. Integral Non-linearity (INL)



- Differential non-linearity (DNL): The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSB). Ideal value: 0 LSB.

Figure 17-12. Differential Non-linearity (DNL)



- Quantization error: Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSB wide) will code to the same value. Always ±0.5 LSB.
- Absolute accuracy: The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of offset, gain error, differential error, non-linearity, and quantization error. Ideal value: ±0.5 LSB.

17.10 Internal Voltage Reference Output

The internal voltage reference is output on XREF pin as described in Table 17-3 on page 188 if the ADC is turned on (see Section 6.2.1 “Voltage Reference Enable Signals and Start-up Time” on page 51). Addition of an external filter capacitor (5 to 10nF) on XREF pin may be necessary. XREF current load must be from 1µA to 100µA with VCC from 2.7V to 5.5V for XREF = 1.1V and with VCC from 4.5V to 5.5V for XREF = 2.56V.

XREF pin can be coupled to an analog input of the ADC (see Section 1.6 “Pin Configuration” on page 6).

Table 17-3. Internal Voltage Reference Output

XREFEN (1)	REFS1 (2)	REFS0 (2)	Voltage Reference Output ($I_{load} \leq$ 100 µA)
0	x	x	Hi-Z, the pin can be used as AREF input or other alternate functions.
1 (1)	0	1	XREF = 1.1V (3)
1 (1)	1	1	XREF = 2.56V (3)(4)

Notes:
1. See “Bit 1 – XREFEN: Internal Voltage Reference Output Enable” on page 193
2. See “Bit 7:6 – REFS1:REFS0: Voltage Reference Selection Bits” on page 188
3. In these configurations, the pin pull-up must be turned off and the pin digital output must be set in Hi-Z.
4. Vcc in range 4.5 - 5.5V.

17.11 Register Description

17.11.1 ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
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 7:6 – REFS1:REFS0: Voltage Reference Selection Bits**

These bits and AREFEN bit from the analog miscellaneous control register (AMISCR) select the voltage reference for the ADC, as shown in Table 17-4. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA register is set). Whenever these bits are changed, the next conversion will take 25 ADC clock cycles. 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.

Table 17-4. Voltage Reference Selections for ADC

REFS1	REFS0	AREFEN	Voltage Reference (V_{REF}) Selection
X	0	0	AVcc used as voltage reference, diconnected from AREF pin.
X	0	1	External voltage reference at AREF pin (AREF ≥ 2.0V)
0	1	0	Internal 1.1V voltage reference
1	1	0	Internal 2.56V voltage reference

- **Bit 5 – ADLAR: ADC Left Adjust Result**

The ADLAR bit affects the presentation of the ADC conversion result in the ADC data register. Write one to ADLAR to left adjust the result. Otherwise, the result is right adjusted. Changing the ADLAR bit will affect the ADC data register immediately, regardless of any ongoing conversions. For a complete description of this bit, see Section 17.11.3 “ADCL and ADCH – The ADC Data Register” on page 191.

18.1.3 DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
	ADC7D / AIN1D	ADC6D / AIN0D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
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 – AIN1D, AIN0D: AIN1D and AIN0D Digital Input Disable**

When this bit is written logic one, the digital input buffer on the corresponding analog compare pin is disabled. The corresponding PIN register bit will always read as zero when this bit is set. When an analog signal is applied to the AIN0/1 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

18.2 Analog Comparator Inputs

18.2.1 Analog Compare Positive Input

It is possible to select any of the inputs of the ADC positive input multiplexer to replace the positive input to the analog comparator. The ADC multiplexer is used to select this input, and consequently, the ADC must be switched off to utilize this feature. If the analog comparator multiplexer enable bit (ACME in ADCSRB register) is set and the ADC is switched off (ADEN in ADCSRA register is zero), MUX[4..0] in ADMUX register select the input pin to replace the positive input to the analog comparator, as shown in Table 18-2 on page 196. If ACME is cleared or ADEN is set, AIN1 pin is applied to the positive input to the analog comparator.

Table 18-2. Analog Comparator Positive Input

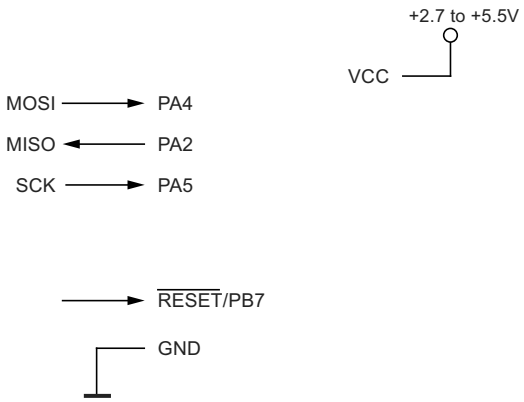
ACME	ADEN	MUX[4..0]	Analog Comparator Positive Input - Comment	
0	x	x xxxx$_b$	AIN1	ADC Switched On
x	1	x xxxx$_b$	AIN1	
1	0	0 0000$_b$	ADC0	ADC Switched Off.
1	0	0 0001$_b$	ADC1	
1	0	0 0010$_b$	ADC2	
1	0	0 0011$_b$	ADC3 / ISRC	
1	0	0 0100$_b$	ADC4	
1	0	0 0101$_b$	ADC5	
1	0	0 0110$_b$	ADC6	
1	0	0 0111$_b$	ADC7	
1	0	0 1000$_b$	ADC8	
1	0	0 1001$_b$	ADC9	
1	0	0 1010$_b$	ADC10	
1	0	Other	This doesn't make sense - Don't use.	

21.8 Serial Downloading

Both the flash and EEPROM memory arrays can be programmed using the serial SPI bus while $\overline{\text{RESET}}$ is pulled to GND. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After $\overline{\text{RESET}}$ is set low, the programming enable instruction needs to be executed first before program/erase operations can be executed.

Note: In Table 21-13 on page 218, the pin mapping for SPI programming is listed. Not all parts use the SPI pins dedicated for the internal SPI interface.

Figure 21-7. Serial Programming and Verify (1)



Note: 1. If the device is clocked by the internal oscillator, it is no need to connect a clock source to the XTAL1 pin

Table 21-13. Pin Mapping Serial Programming

Symbol	Pin Name	I/O	Function
MOSI	PA4	I	Serial data in
MISO	PA2	O	Serial data out
SCK	PA5	I	Serial clock

When programming the EEPROM, an auto-erase cycle is built into the self-timed programming operation (in the serial mode **ONLY**) and there is no need to first execute the chip erase instruction. The chip erase operation turns the content of every memory location in both the program and EEPROM arrays into 0xFF.

Depending on CKSEL fuses, a valid clock must be present. The minimum low and high periods for the serial clock (SCK) input are defined as follows:

Low: > **2** CPU clock cycles for f_{ck} < 12MHz, **3** CPU clock cycles for f_{ck} ≥ 12Hz

High: > **2** CPU clock cycles for f_{ck} < 12MHz, **3** CPU clock cycles for f_{ck} ≥ 12MHz

21.8.1 Serial Programming Algorithm

When writing serial data to the Atmel® ATtiny87/167, data is clocked on the rising edge of SCK.

When reading data from the ATtiny87/167, data is clocked on the falling edge of SCK. See Figure 21-7 and Figure 21-8 on page 221 for timing details.

To program and verify the Atmel ATtiny87/167 in the serial programming mode, the following sequence is recommended (see four byte instruction formats in Table 21-15 on page 220):

1. Power-up sequence: Apply power between Vcc 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.

4. Comparison between ADC inputs and voltage references

   In the analog comparator module, comparing any ADC input (ADC[10..0]) with voltage references (2.56V, 1.28V, 1.10V, 0.64V or 0.32V) fails.

   Regardless, AIN1 input can be compared with the voltage references and any ADC input can be compared with AIN0 input.

   Problem Fix/Workaround

   Do not use this configuration.

5. Register bits of DIDR1

   ADC8D, ADC9D and ADC10D (digital input disable) initially located at bit 4 up to 6 are instead located at bit 0 up to 2. These register bits are also in write only mode.

   Problem Fix/Workaround

   Allow for the change in bit locations and the access mode restriction.

6. LIN Break Delimiter

   In SLAVE MODE, a BREAK field detection error can occur under following conditions.

   The problem occurs if 2 conditions occur simultaneously:

   a. The DOMINANT part of the BREAK is (N+0.5)*Tbit long with N=13, 14,15, ...
   b. The RECESSIVE part of the BREAK (BREAK DELIMITER) is equal to 1*Tbit. (see note below)

   The BREAK_high is not detected, and the 2nd bit of the SYNC field is interpreted as the BREAK DELIMITER.

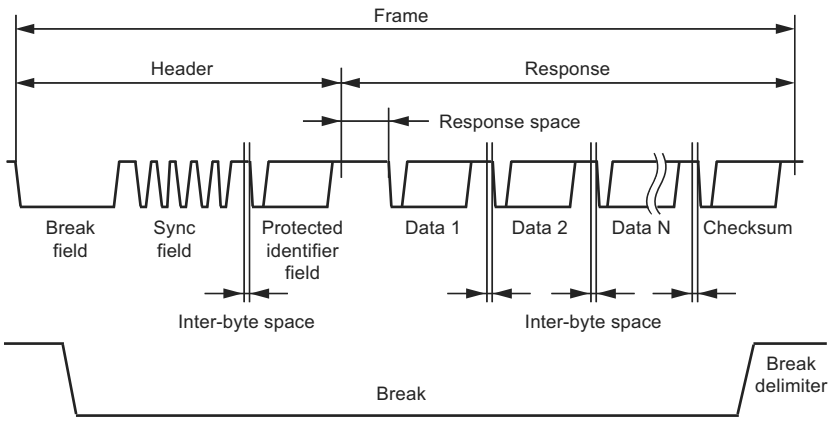
   The error is detected as a framing error on the first bits of the PID or on subsequent Data or a Checksum error.

   There is no error if BREAK_high is greater than 1 × Tbit + 18%.

   There is no problem in master mode.

Note: LIN2.1 protocol specification paragraph 2.3.1.1 Break field says: “A break field is always generated by the master task(in the master node) and it shall be at least 13 nominal bit times of dominant value, followed by a break delimiter, as shown in Figure 29-1. **The break delimiter shall be at least one nominal bit time long.**”

Figure 29-1. The Break Field



Workaround

None