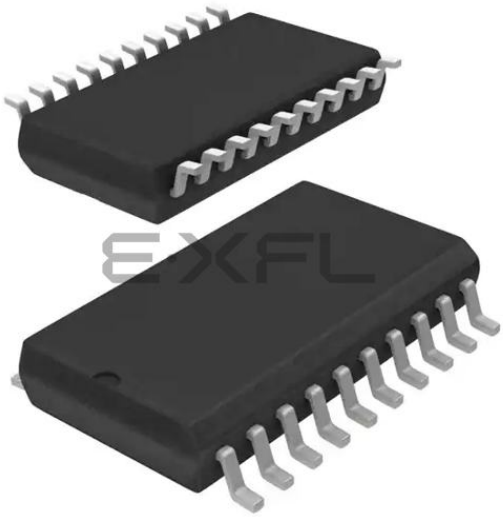


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

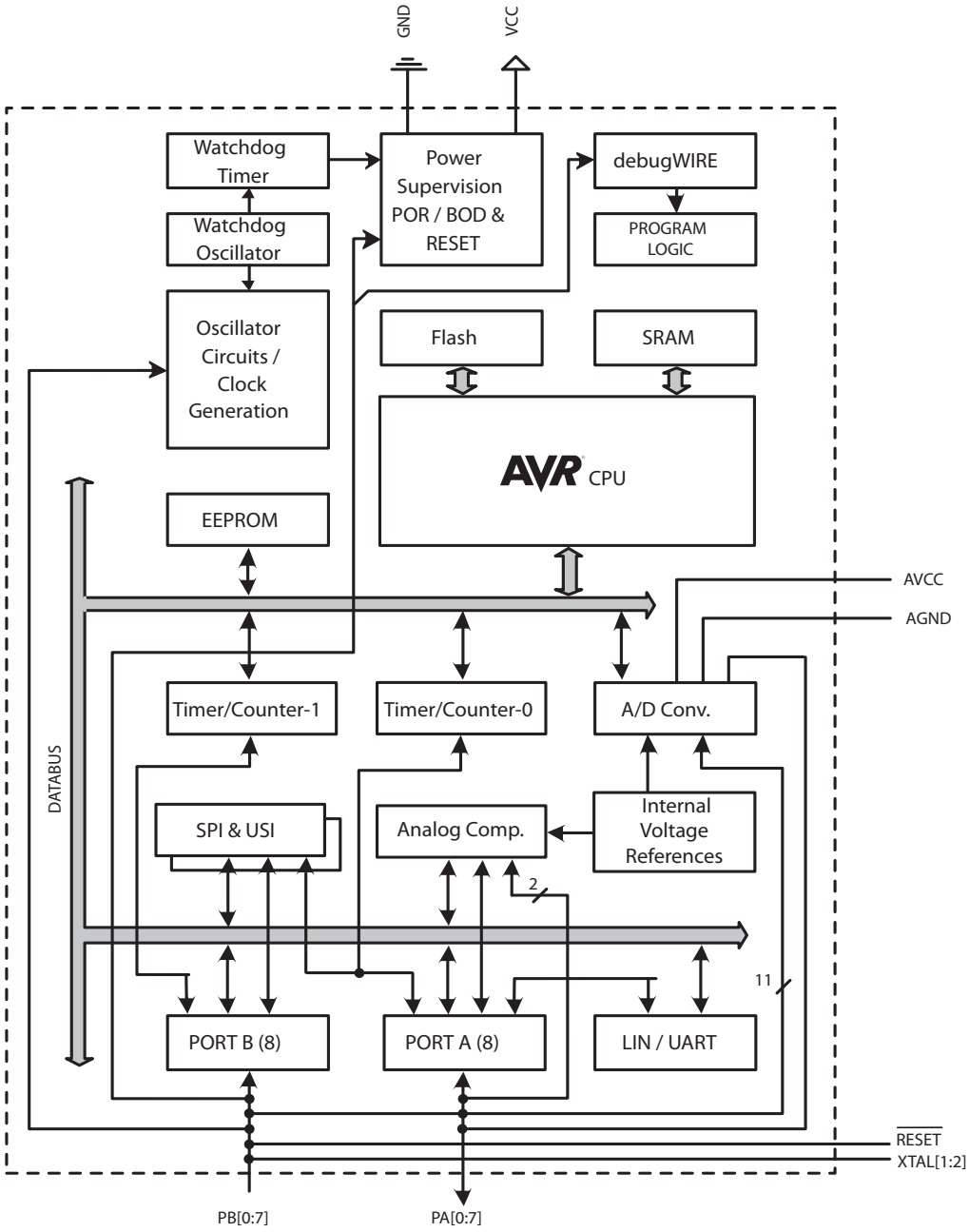
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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)	1.8V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SOIC (0.295", 7.50mm Width)
Supplier Device Package	20-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/attiny87-su

1.3 Block Diagram

Figure 1-1. Block Diagram



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

Assembly Code Example

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

C Code Example

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

2.7.2 Interrupt Response Time

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

A return from an interrupt handling routine takes four clock cycles. During these four clock cycles, the Program Counter (two bytes) is popped back from the Stack, the Stack Pointer is incremented by two, and the I-bit in SREG is set.

Table 4-3. Internal Calibrated RC Oscillator Operating Modes[1]

Frequency Range[2] (MHz)	CKSEL[3:0][3][4] CSEL[3:0][5]
7.6 - 8.4	0010

Notes:
1. If 8 MHz frequency exceeds the specification of the device (depends on V_{CC}), the CKDIV8 fuse can be programmed in order to divide the internal frequency by 8.
2. The frequency ranges are guideline values.
3. The device is shipped with this CKSEL = “0010”.
4. Flash Fuse bits.
5. CLKSELR register bits.

When this Oscillator is selected, start-up times are determined by the SUT Fuses or by CSUT field as shown in Table 4-4.

Table 4-4. Start-up Times for the Internal Calibrated RC Oscillator Clock Selection

SUT[1:0][1] CSUT[1:0][2]	Start-up Time from Power-down/save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00[3]	6 CK	14CK	BOD enabled
01	6 CK	14CK + 4.1 ms	Fast rising power
10[4]	6 CK	14CK + 65 ms	Slowly rising power
11	Reserved		

Notes:
1. Flash Fuse bits
2. CLKSELR register bits
3. This setting is only available if RSTDISBL fuse is not set
4. The device is shipped with this option selected.

4.2.3 128 KHz Internal Oscillator

The 128 KHz internal Oscillator is a low power Oscillator providing a clock of 128 KHz. The frequency is nominal at 3V and 25°C. This clock may be selected as the system clock by programming CKSEL Fuses or CSEL field as shown in Table 4-1 on page 25.

When this clock source is selected, start-up times are determined by the SUT Fuses or by CSUT field as shown in Table 4-5.

Table 4-5. Start-up Times for the 128 kHz Internal Oscillator

SUT[1:0][1] CSUT[1:0][2]	Start-up Time from Power-down/save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00[3]	6 CK	14CK	BOD enabled
01	6 CK	14CK + 4.1 ms	Fast rising power
10	6 CK	14CK + 65 ms	Slowly rising power
11	Reserved		

Notes:
1. Flash Fuse bits
2. CLKSELR register bits
3. This setting is only available if RSTDISBL fuse is not set

- **Bit 1 – EXTRF: External Reset Flag**

This bit is set if an External Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 0 – PORF: Power-on Reset Flag**

This bit is set if a Power-on Reset occurs. The bit is reset only by writing a logic zero to the flag.

To make use of the Reset Flags to identify a reset condition, the user should read and then Reset the MCUSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the Reset Flags.

6.2 Internal Voltage Reference

ATtiny87/167 features an internal bandgap reference. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC.

6.2.1 Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in Table 22-7 on page 246. To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODLEVEL[2:0] Fuses).
2. When the bandgap reference is connected to the Analog Comparator (by setting the ACIRS bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACIRS bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or ADC is used. To reduce power consumption in Power-down mode or in Power-save, the user can avoid the three conditions above to ensure that the reference is turned off before entering in these power reduction modes.

6.3 Watchdog Timer

ATtiny87/167 has an Enhanced Watchdog Timer (WDT). The main features are:

- **Clocked from separate On-chip Oscillator**
- **4 Operating modes**
  - **Interrupt**
  - **System Reset**
  - **Interrupt and System Reset**
  - **Clock Monitoring**
- **Selectable Time-out period from 16ms to 8s**
- **Possible Hardware fuse Watchdog always on (WDTON) for fail-safe mode**

6.3.1 Watchdog Timer Behavior

The Watchdog Timer (WDT) is a timer counting cycles of a separate on-chip 128 KHz oscillator.

The following code example shows one assembly and one C function for turning off the Watchdog Timer. The example assumes that interrupts are controlled (e.g. by disabling interrupts globally) so that no interrupts will occur during the execution of these functions.

Assembly Code Example[1]

```
WDT_off:
  ; Turn off global interrupt
  cli
  ; Reset Watchdog Timer
  wdr
  ; Clear WDRF in MCUSR
  in    r16, MCUSR
  andi  r16, (0xff & (0<<WDRF))
  out   MCUSR, r16
  ; Write logical one to WDCE and WDE
  ; Keep old prescaler setting to prevent unintentional time-out
  lds r16, WDTCR
  ori   r16, (1<<WDCE) | (1<<WDE)
  sts WDTCR, r16
  ; Turn off WDT
  ldi   r16, (0<<WDE)
  sts WDTCR, r16
  ; Turn on global interrupt
  sei
  ret
```

C Code Example[1]

```
void WDT_off(void)
{
  __disable_interrupt();
  __watchdog_reset();
  /* Clear WDRF in MCUSR */
  MCUSR &= ~(1<<WDRF);
  /* Write logical one to WDCE and WDE */
  /* Keep old prescaler setting to prevent unintentional time-out */
  WDTCR |= (1<<WDCE) | (1<<WDE);
  /* Turn off WDT */
  WDTCR = 0x00;
  __enable_interrupt();
}
```

Note: 1. See ”About Code Examples” on page 6.

Note that if the Watchdog is accidentally enabled, for example by a runaway pointer or brown-out condition, the device will be reset and the Watchdog Timer will stay enabled. If the code is not set up to handle the Watchdog, this might lead to an eternal loop of time-out resets. To avoid this situation, the application software should always clear the Watchdog System Reset Flag (WDRF) and the WDE control bit in the initialization routine, even if the Watchdog is not in use.

8.3 Register Description

8.3.1 EICRA – External Interrupt Control Register A

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
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 – Res: Reserved Bits**

These bits are reserved bits in the ATtiny87/167 and will always read as zero.

- **Bits 3:2 – ISC1[1:0]: Interrupt Sense Control 1 Bit 1 and Bit 0**

The External Interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 8-1. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

- **Bits 1:0 – ISC0[1:0]: Interrupt Sense Control 0 Bit 1 and Bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 8-1. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 8-1. Interrupt Sense Control

ISCn1	ISCn0	Description
0	0	The low level of INTn generates an interrupt request.
0	1	Any logical change on INTn generates an interrupt request.
1	0	The falling edge of INTn generates an interrupt request.
1	1	The rising edge of INTn generates an interrupt request.

8.3.2 EIMSK – External Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:2 – Res: Reserved Bits**

These bits are reserved bits in the ATtiny87/167 and will always read as zero.

10.2.1 Definitions

The following definitions are used extensively throughout the section:

BOTTOM	The counter reaches the BOTTOM when it becomes zero (0x00).
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR0A Register. The assignment is dependent on the mode of operation.

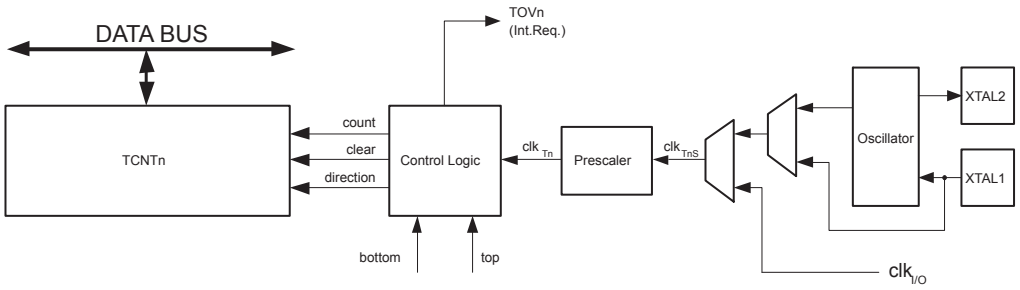
10.3 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal synchronous or an external asynchronous clock source. The clock source is selected by the clock select logic which is controlled by the clock select (CS0[2:0]) bits located in the Timer/Counter control register (TCCR0).The clock source clk_T0 is by default equal to the MCU clock, $clk_{I/O}$. When the AS0 bit in the ASSR Register is written to logic one, the clock source is taken from the Timer/Counter Oscillator connected to XTAL1 and XTAL2 or directly from XTAL1. For details on asynchronous operation, see “ASSR – Asynchronous Status Register” on page 102. For details on clock sources and prescaler, see “Timer/Counter0 Prescaler” on page 99.

10.4 Counter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 10-2 shows a block diagram of the counter and its surrounding environment.

Figure 10-2. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT0 by 1.
direction	Selects between increment and decrement.
clear	Clear TCNT0 (set all bits to zero).
clk_T0	Timer/Counter0 clock.
top	Signalizes that TCNT0 has reached maximum value.
bottom	Signalizes that TCNT0 has reached minimum value (zero).

10.11.6 TIMSK0 – Timer/Counter0 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	–	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:2 – Res: Reserved Bits**

These bits are reserved in the ATtiny87/167 and will always read as zero.

- **Bit 1 – OCIE0A: Timer/Counter0 Output Compare Match A Interrupt Enable**

When the OCIE0A bit is written to one and the I-bit in the Status Register is set (one), the Timer/Counter0 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a compare match in Timer/Counter0 occurs, i.e., when the OCF0A bit is set in the Timer/Counter0 Interrupt Flag Register – TIFR0.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one and the I-bit in the Status Register is set (one), the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter0 Interrupt Flag Register – TIFR0.

10.11.7 TIFR0 – Timer/Counter0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	–	–	–	–	–	–	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:2 – Res: Reserved Bits**

These bits are reserved in the ATtiny87/167 and will always read as zero.

- **Bit 1 – OCF0A: Output Compare Flag 0 A**

The OCF0A bit is set (one) 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, OCIE0 (Timer/Counter0 Compare match Interrupt Enable), and OCF0A are set (one), the Timer/Counter0 Compare match Interrupt is executed.

- **Bit 0 – TOV0: Timer/Counter0 Overflow Flag**

The TOV0 bit is set (one) 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, TOIE0A (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set (one), the Timer/Counter0 Overflow interrupt is executed. In PWM mode, this bit is set when Timer/Counter0 changes counting direction at 0x00.

Register. When writing the ICR1 Register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

For more information on how to access the 16-bit registers refer to “Accessing 16-bit Registers” on page 111.

12.6.1 Input Capture Trigger Source

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

Both the Input Capture pin (ICP1) and the Analog Comparator output (ACO) inputs are sampled using the same technique as for the T1 pin (Figure 11-1 on page 106). The edge detector is also identical. However, when the noise canceler is enabled, additional logic is inserted before the edge detector, which increases the delay by four system clock cycles. Note that the input of the noise canceler and edge detector is always enabled unless the Timer/Counter is set in a Waveform Generation mode that uses ICR1 to define TOP.

An Input Capture can be triggered by software by controlling the port of the ICP1 pin.

12.6.2 Noise Canceler

The noise canceler improves noise immunity by using a simple digital filtering scheme. The noise canceler input is monitored over four samples, and all four must be equal for changing the output that in turn is used by the edge detector.

The noise canceler is enabled by setting the Input Capture Noise Canceler (ICNC1) bit in Timer/Counter Control Register B (TCCR1B). When enabled the noise canceler introduces additional four system clock cycles of delay from a change applied to the input, to the update of the ICR1 Register. The noise canceler uses the system clock and is therefore not affected by the prescaler.

12.6.3 Using the Input Capture Unit

The main challenge when using the Input Capture unit is to assign enough processor capacity for handling the incoming events. The time between two events is critical. If the processor has not read the captured value in the ICR1 Register before the next event occurs, the ICR1 will be overwritten with a new value. In this case the result of the capture will be incorrect.

When using the Input Capture interrupt, the ICR1 Register should be read as early in the interrupt handler routine as possible. Even though the Input Capture interrupt has relatively high priority, the maximum interrupt response time is dependent on the maximum number of clock cycles it takes to handle any of the other interrupt requests.

Using the Input Capture unit in any mode of operation when the TOP value (resolution) is actively changed during operation, is not recommended.

Measurement of an external signal’s duty cycle requires that the trigger edge is changed after each capture. Changing the edge sensing must be done as early as possible after the ICR1 Register has been read. After a change of the edge, the Input Capture Flag (ICF1) must be cleared by software (writing a logical one to the I/O bit location). For measuring frequency only, the clearing of the ICF1 flag is not required (if an interrupt handler is used).

15.4.1 LIN Overview

The LIN/UART controller is designed to match as closely as possible to the LIN software application structure. The LIN software application is developed as independent tasks, several slave tasks and one master task (c.f. Section 15.3.4 on page 163). The ATtiny87/167 conforms to this perspective. The only link between the master task and the slave task will be at the cross-over point where the interrupt routine is called once a new identifier is available. Thus, in a master node, housing both master and slave task, the Tx LIN Header function will alert the slave task of an identifier presence. In the same way, in a slave node, the Rx LIN Header function will alert the slave task of an identifier presence.

When the slave task is warned of an identifier presence, it has first to analyze it to know what to do with the response. Hardware flags identify the presence of one of the specific identifiers from 60 (0x3C) up to 63 (0x3F).

For LIN communication, only four interrupts need to be managed:

- LIDOK: New LIN identifier available,
- LRXOK: LIN response received,
- LTXOK: LIN response transmitted,
- LERR: LIN Error(s).

The wake-up management can be automated using the UART wake-up capability and a node sending a minimum of 5 low bits (0xF0) for LIN 2.1 and 8 low bits (0x80) for LIN 1.3. Pin change interrupt on LIN wake-up signal can be also used to exit the device of one of its sleep modes.

Extended frame identifiers 62 (0x3E) and 63 (0x3F) are reserved to allow the embedding of user-defined message formats and future LIN formats. The byte transfer mode offered by the UART will ensure the upwards compatibility of LIN slaves with accommodation of the LIN protocol.

15.4.2 UART Overview

The LIN/UART controller can also function as a conventional UART. By default, the UART operates as a full duplex controller. It has local loop back circuitry for test purposes. The UART has the ability to buffer one character for transmit and two for receive. The receive buffer is made of one 8-bit serial register followed by one 8-bit independent buffer register. Automatic flag management is implemented when the application puts or gets characters, thus reducing the software overhead. Because transmit and receive services are independent, the user can save one device pin when one of the two services is not used. The UART has an enhanced baud rate generator providing a maximum error of 2% whatever the clock frequency and the targeted baud rate.

Table 15-1. LIN/UART Command List

LENA	LCMD[2]	LCMD[1]	LCMD[0]	Command	Comment
0	x	x	x	Disable peripheral	
1	0	0	0	Rx Header - LIN Abort	LIN Withdrawal
			1	Tx Header	LCMD[2:0]=000 after Tx
		1	0	Rx Response	LCMD[2:0]=000 after Rx
			1	Tx Response	LCMD[2:0]=000 after Tx
	1	0	0	Byte transfer	no CRC, no Time out LTXDL=LRXDL=0 (LINDLR: read only register)
		1	0	Rx Byte	
		0	1	Tx Byte	
		1	1	Full duplex	

15.4.5 Enable / Disable

Setting the LENA bit in LINCR register enables the LIN/UART controller. To disable the LIN/UART controller, LENA bit must be written to 0. No wait states are implemented, so, the disable command is taken into account immediately.

15.4.6 LIN Commands

Clearing the LCMD[2] bit in LINCR register enables LIN commands.

As shown in Table 15-1 on page 166, four functions controlled by the LCMD[1:0] bits of LINCR register are available (c.f. Figure 15-5 on page 165).

15.4.6.1 Rx Header / LIN Abort Function

This function (or state) is mainly the withdrawal mode of the controller.

When the controller has to execute a master task, this state is the start point before enabling a Tx Header command.

When the controller has only to execute slave tasks, LIN header detection/acquisition is enabled as background function. At the end of such an acquisition (Rx Header function), automatically the appropriate flags are set, and in LIN 1.3, the LINDLR register is set with the uncoded length value.

This state is also the start point before enabling the Tx or the Rx Response command.

A running function (i.e. Tx Header, Tx or Rx Response) can be aborted by clearing LCMD[1:0] bits in LINCR register (See ”Break-in-data” on page 176.). In this case, an abort flag - LABORT - in LINERR register will be set to inform the other software tasks. No wait states are implemented, so, the abort command is taken into account immediately.

Rx Header function is responsible for:

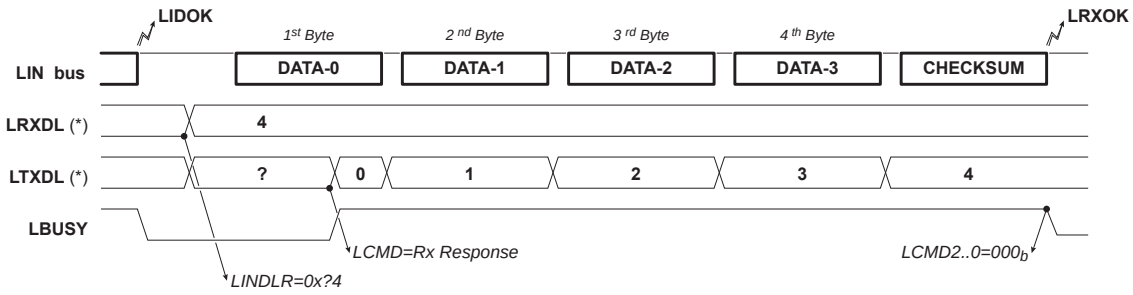
- The BREAK field detection,
- The hardware re-synchronization analyzing the SYNCH field,
- The reception of the PROTECTED IDENTIFIER field, the parity control and the update of the LINDLR register in case of LIN 1.3,
- The starting of the Frame_Time_Out,
- The checking of the LIN communication integrity.

15.5.7.2 Data Length in LIN 1.3

- LRXDL and LTXDL fields are both hardware updated before setting LIDOK by decoding the data length code contained in the received PROTECTED IDENTIFIER (LRXDL = LTXDL).
- Via the above mechanism, a length of 0 or >8 is not possible.

15.5.7.3 Data Length in Rx Response

Figure 15-9. LIN2.1 - Rx Response - No error

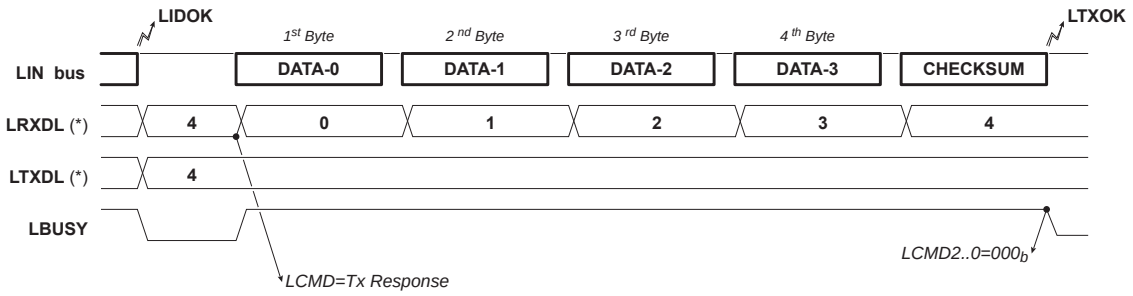


(*) : LRXDL & LTXDL updated by user

- The user initializes LRXDL field before setting the Rx Response command,
- After setting the Rx Response command, LTXDL is reset by hardware,
- LRXDL field will remain unchanged during Rx (during busy signal),
- LTXDL field will count the number of received bytes (during busy signal),
- If an error occurs, Rx stops, the corresponding error flag is set and LTXDL will give the number of received bytes without error,
- If no error occurs, LRXOK is set after the reception of the CHECKSUM, LRXDL will be unchanged (and LTXDL = LRXDL).

15.5.7.4 Data Length in Tx Response

Figure 15-10. LIN1.3 - Tx Response - No error



(*) : LRXDL & LTXDL updated by Rx Response or Tx Response task

- The user initializes LTXDL field before setting the Tx Response command,
- After setting the Tx Response command, LRXDL is reset by hardware,
- LTXDL will remain unchanged during Tx (during busy signal),
- LRXDL will count the number of transmitted bytes (during busy signal),

- **Bit 0 – LRXOK: Receive Performed Interrupt**

  This bit generates an interrupt if its respective enable bit - LENRXOK - is set in LINENIR.

  – 0 = No Rx

  – 1 = Rx Response complete.

  The user clears this bit by writing 1, in order to reset this interrupt.

  In UART mode, this bit is also cleared by reading LINDAT.

15.6.3 LINENIR – LIN Enable Interrupt Register

Bit	7	6	5	4	3	2	1	0	
(0xCA)	–	–	–	–	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 LINERR – LIN Error Register

Bit	7	6	5	4	3	2	1	0	
(0xCB)	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**

17.11 Register Description

17.11.1 ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	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	

- **Bits 7:6 – REFS[1:0]: 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 (AV_{CC} - 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	AV_{CC} 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 “ADCL and ADCH – ADC Data Register” on page 207.

- **Bits 4:0 – MUX[4:0]: Analog Channel and Gain Selection Bits**

These bits select which combination of analog inputs are connected to the ADC. In case of differential input, gain selection is also made with these bits. Refer to Table 17-5 for details. If these bits are changed during a conversion, the change will not go into effect until this conversion is complete (ADIF in ADCSRA register is set).

20.1.3 Performing a Page Write

To execute Page Write, set up the address in the Z-pointer, write “00000101 $_b$” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

- The CPU is halted during the Page Write operation.

20.2 Addressing the Flash During Self-Programming

The Z-pointer is used to address the SPM commands. The Z pointer consists of the Z-registers ZL and ZH in the register file. The number of bits actually used is implementation dependent.

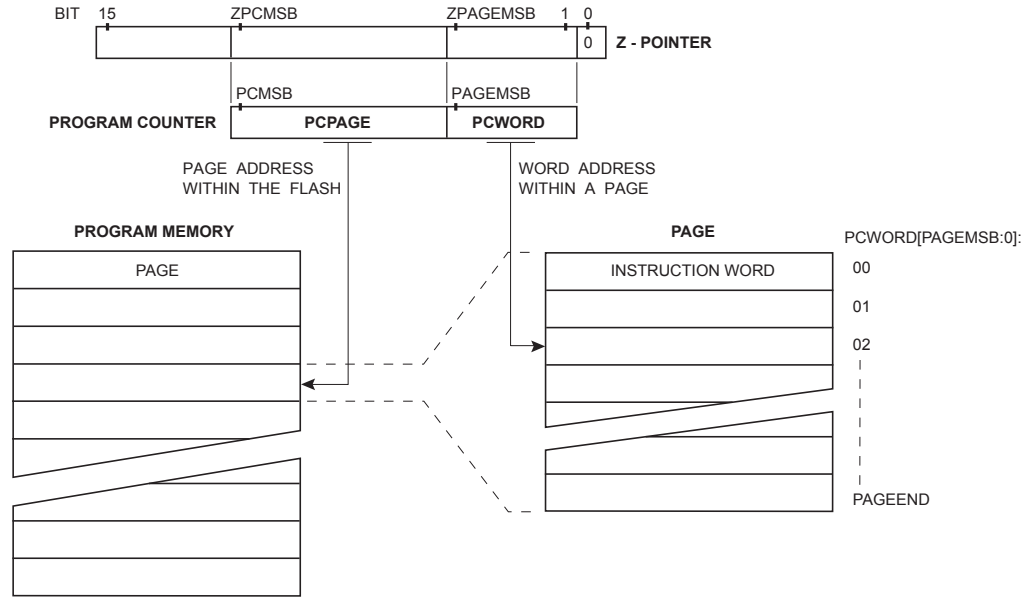
Bit	15	14	13	12	11	10	9	8	
	Z15	Z14	Z13	Z12	Z11	Z10	Z9	Z8	ZH (R31)
	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0	ZL (R30)
Bit	7	6	5	4	3	2	1	0	

Since the Flash is organized in pages (see Table 21-7 on page 227), the Program Counter can be treated as having two different sections. One section, consisting of the least significant bits, is addressing the words within a page, while the most significant bits are addressing the pages. This is shown in Figure 20-1.

Note that the Page Erase and Page Write operations are addressed independently. Therefore it is of major importance that the software addresses the same page in both the Page Erase and Page Write operation.

The LPM instruction uses the Z-pointer to store the address. Since this instruction addresses the Flash byte-by-byte, also the LSB (bit Z0) of the Z-pointer is used.

Figure 20-1. Addressing the Flash During SPM (1)



Note: 1. The different variables used in Table 20-2 are listed in Table 21-7 on page 227.

21.7.15 Reading the Temperature Sensor Parameter Bytes

The algorithm for reading the Temperature Sensor parameter bytes is as follows (refer to "Programming the Flash" on page 231 for details on Command and Address loading):

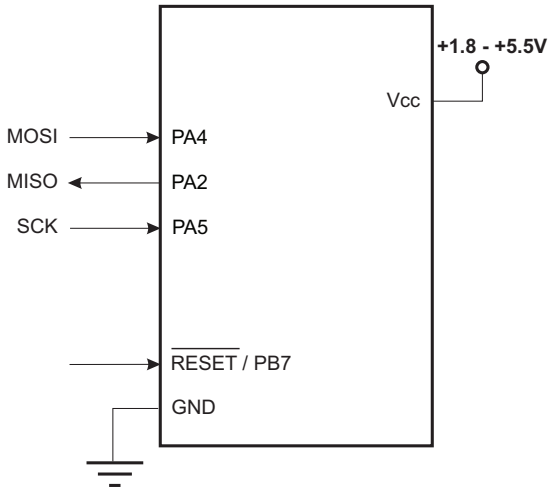
1. **A:** Load Command "0000 1000 $_b$".
2. **B:** Load Address Low Byte, 0x0003 or 0x0005.
3. Set $\overline{OE}$ to "0", and BS1 to "1". The Temperature Sensor parameter byte can now be read at DATA.
4. Set $\overline{OE}$ to "1".

21.8 Serial Downloading

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

Note: In Table 21-13, 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 (Note:)



Note: 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.

Table 21-15. Serial Programming Instruction Set (Continued)

Instruction/Operation	Instruction Format			
	Byte 1	**Byte 2**	**Byte 3**	**Byte4**
Write EEPROM Memory	0xC0	0x00	00aa aaaa $_b$	data byte in
Write EEPROM Memory Page (page access)	0xC2	0x00	00aa aa00 $_b$	0x00
Write Lock bits	0xAC	0xE0	0x00	data byte in
Write Fuse bits	0xAC	0xA0	0x00	data byte in
Write Fuse High bits	0xAC	0xA8	0x00	data byte in
Write Extended Fuse Bits	0xAC	0xA4	0x00	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 21-8 on page 241.

T_A = -40°C to +85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted) (Continued)

Symbol	Parameter	Condition	Min.	Typ.[1]	Max.	Units
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{pu}	I/O Pin Pull-up Resistor		20		50	kΩ
I_{CC}	Power Supply Current [6] Active Mode (external clock)	8 MHz, V_{CC} = 5V		5.5	7.0	mA
		4 MHz, V_{CC} = 3V		1.8	2.5	mA
		1 MHz, V_{CC} = 2V		0.3	0.6	mA
	Power Supply Current [6] Idle Mode (external clock)	8 MHz, V_{CC} = 5V		1.8	2.5	mA
		4 MHz, V_{CC} = 3V		0.5	0.8	mA
		1 MHz, V_{CC} = 2V		0.05	0.2	mA
	Power Supply Current [7] Power-down Mode	WDT enabled, V_{CC} = 3V		5	10	µA
		WDT disabled, V_{CC} = 3V		0.15	4	µA
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = V_{CC}/2	-10	10	40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACID}	Analog Comparator Propagation Delay Common Mode V_{CC}/2	V_{CC} = 2.7V		170		ns
		V_{CC} = 5.0V		180		ns

Notes:
1. “Typ.", typical values at 25°C. Maximum values are characterized values and not test limits in production.
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 (10 mA at V_{CC}= 5V, 5 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   - The sum of all IOL, for all ports, should not exceed 120 mA.
   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 (10 mA at V_{CC} = 5V, 5 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   - The sum of all IOH, for all ports, should not exceed 120 mA.
   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 are with external clock using methods described in “Minimizing Power Consumption” on page 45. Power Reduction is enabled (PRR = 0xFF) and there is no I/O drive.
7. BOD Disabled.

22.10 SPI Timing Characteristics

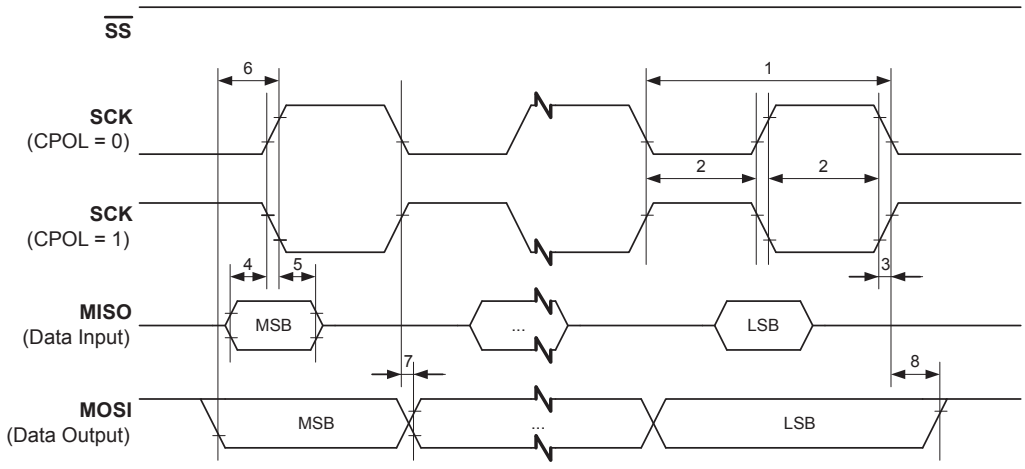
See Figure 22-6 and Figure 22-7 for details.

Table 22-12. SPI Timing Parameters

	Description	Mode	Min.	Typ.	Max.	
1	SCK period	Master		See Table 13-4		
2	SCK high/low	Master		50% duty cycle		
3	Rise/Fall time	Master		3.6		
4	Setup	Master		10		
5	Hold	Master		10		
6	Out to SCK	Master		0.5 • t_{sck}		ns
7	SCK to out	Master		10		
8	SCK to out high	Master		10		
9	SS low to out	Slave		15		
10	SCK period	Slave	4 • t_{ck}			
11	SCK high/low (1)	Slave	2 • t_{ck}			
12	Rise/Fall time	Slave			1.6	µs
13	Setup	Slave	10			
14	Hold	Slave	t_{ck}			
15	SCK to out	Slave		15		ns
16	SCK to $\overline{SS}$ high	Slave	20			
17	$\overline{SS}$ high to tri-state	Slave		10		
18	SS low to SCK	Slave	2 • t_{ck}			

Note: In SPI Programming mode the minimum SCK high/low period is:
- 2 t_{CLCL} for f_{CK} < 12 MHz
- 3 t_{CLCL} for f_{CK} >12 MHz

Figure 22-6. SPI Interface Timing Requirements (Master Mode)



29. Datasheet Revision History

29.1 Rev. 8265D – 01/2014

Updated:

- The datasheet updated with Atmel “new blue logo” and addresses
- Corrected the speed (Frequency) specified in the table “ATtiny167” on page 277 to 16 MHz
- The Figure 23-2 on page 255, Figure 23-5 on page 256 and Figure 23-11 on page 260 updated according to the 16 MHz Frequency

29.2 Rev. 8265C – 03/12

Updated:

- “Memory Size Summary” on page 2
- The datasheet status changes from “Preliminary” to “Complete”
- “Ordering Information” on page 276. 32M1-A package replaced by 32NP
- “Packaging Information” on page 278. 32M1-A package replaced by 32NP
- The whole document with Atmel new template that include blue logo and new addresses on the last page.

29.3 Rev. 8265B – 09/10

Updated:

- Section 9.3.3 “Alternate Functions of Port A” on page 76, Bit 3 and Bit 4
- Section 26.2 “ATtiny167” on page 277, ordering codes
- Section 10.11.1 “TCCR0A – Timer/Counter0 Control Register A” on page 99, Bit 1 and Bit 0 descriptions
- Bit syntax throughout the datasheet, e.g. from CS02:0 to CS0[2:0]

29.4 Rev. 8265A – 08/10

Initial revision.