## Features

- **AVR® 8-bit RISC Microcontroller with 83 ns Instruction Cycle Time**
- **USB Hub with One Attached and Four External Ports**
- **USB Function with Two Programmable Endpoints**
- **External Program Memory, 512-byte Data SRAM**
- **32 x 8 General Purpose Working Registers**
- **32 Programmable I/O Port Pins**
- **Programmable Serial UART**
- **Master/Slave SPI Serial Interface**
- **One 8-bit Timer/Counter with Separate Pre-scaler**
- **One 16-bit Timer/Counter with Separate Pre-scaler and Two PWMs**
- **External and Internal Interrupt Sources**
- **Programmable Watchdog Timer**
- **6 MHz Oscillator with On-chip PLL**
- **5V Operation with On-chip 3.3V Power Supply**
- **100-lead LQFP Package**

## Description

The Atmel AT43USB320A is an 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the AT43USB320A achieves throughputs approaching 12 MIPS. The AVR core combines a rich instruction set with 32 general-purpose working registers. All 32 registers are directly connected to the 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 AT43USB320A features an on-chip 512-byte of data memory. It is supported by a standard set of peripherals such as timer/counter modules, watchdog timer and internal and external interrupt sources. The major peripheral included in the AT43USB320A is the USB Hub with an embedded function for use in peripherals such as monitor with remote control as shown in Figure 1.

Note: There are two versions of the AT43USB320A. They are indicated by the internal part numbers 55618D and 55618E. The only difference between the two versions is in the polarity of the SUSPEND pin. The 55618D SUSPEND pin is active low, while the 55618E SUSPEND pin is active high.

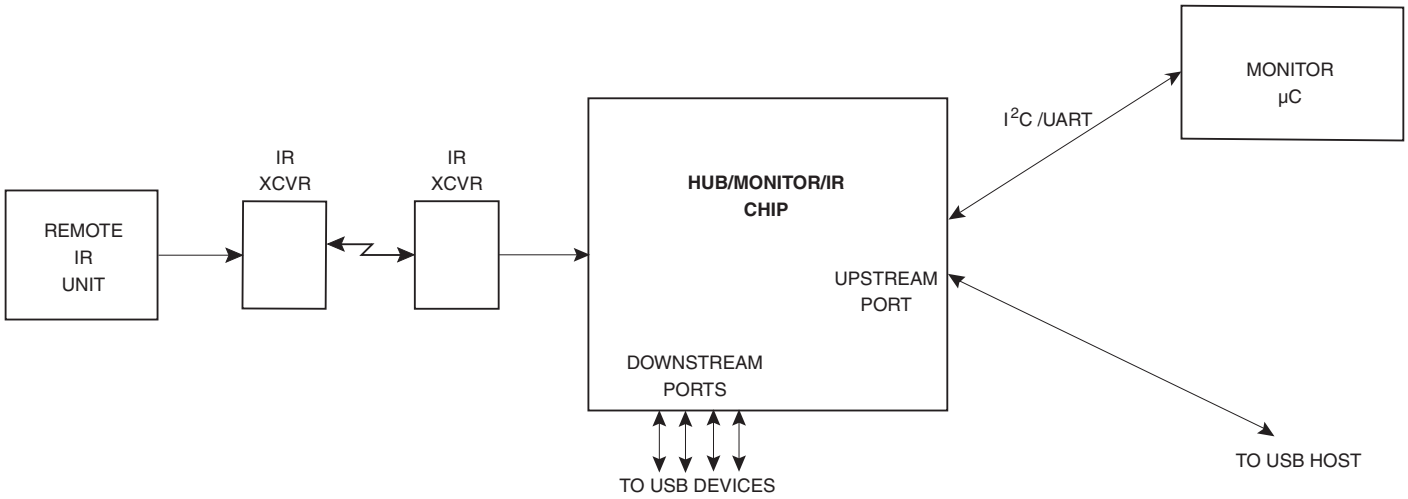
# Full-speed USB Microcontroller with an Embedded Hub

# AT43USB320A

Rev. 1443E–USB–4/04

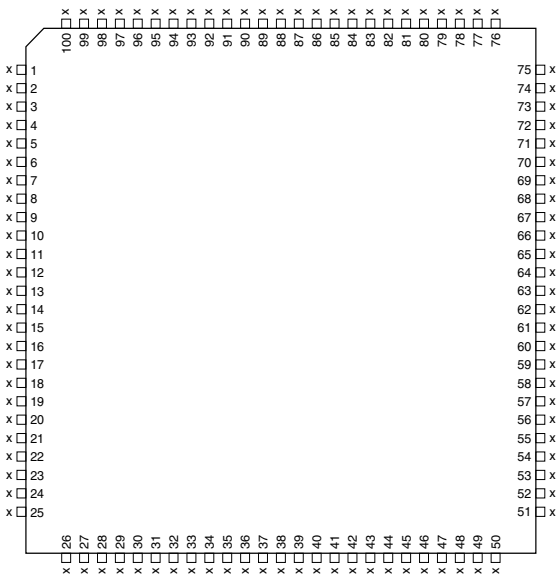
## Hub/Monitor/IR Chip Application

**Figure 1.** Application Example




## Pin Configurations

100-lead LQFP



## Pin Assignment

Type: I = Input
O = Output
B = Bi-directional
V = Power Supply, Ground

| Pin Number | Signal | Type |
|---|---|---|
| 1 | PD2 | B |
| 2 | PD3 | B |
| 3 | PD4 | B |
| 4 | PD5 | B |
| 5 | PD6 | B |
| 6 | PD7 | B |
| 7 | 6/12N | I |
| 8 | LFT | O |
| 9 | XTAL1 | I |
| 10 | XTAL2 | O |
| 11 | VSS | V |
| 12 | TESTN | I |
| 13 | A0 | B |
| 14 | A1 | B |
| 15 | A2 | B |
| 16 | A3 | B |
| 17 | A4 | B |
| 18 | A5 | B |
| 19 | A6 | B |
| 20 | A7 | B |
| 21 | VSS | V |
| 22 | A8 | B |
| 23 | A9 | B |
| 24 | A10 | B |
| 25 | NC | – |
| 26 | NC | – |
| 27 | A11 | B |
| 28 | A12 | B |
| 29 | A13 | B |
| 30 | A14 | B |
| 31 | A15 | B |

| Pin Number | Signal | Type |
|---|---|---|
| 32 | VCC | V |
| 33 | VSS | V |
| 34 | CEXT1 | O |
| 35 | SUSPEND | O |
| 36 | D0 | I |
| 37 | D1 | I |
| 38 | D2 | I |
| 39 | D3 | I |
| 40 | D4 | I |
| 41 | D5 | I |
| 42 | D6 | I |
| 43 | D7 | I |
| 44 | VSS | V |
| 45 | D8 | I |
| 46 | D9 | I |
| 47 | D10 | I |
| 48 | D11 | I |
| 49 | NC | – |
| 50 | NC | – |
| 51 | D12 | I |
| 52 | D13 | I |
| 53 | D14 | I |
| 54 | D15 | I |
| 55 | VSS | V |
| 56 | ICP | V |
| 57 | DP0 | B |
| 58 | DM0 | B |
| 59 | DP1 | B |
| 60 | DM1 | B |
| 61 | VCC | V |
| 62 | VSS | V |

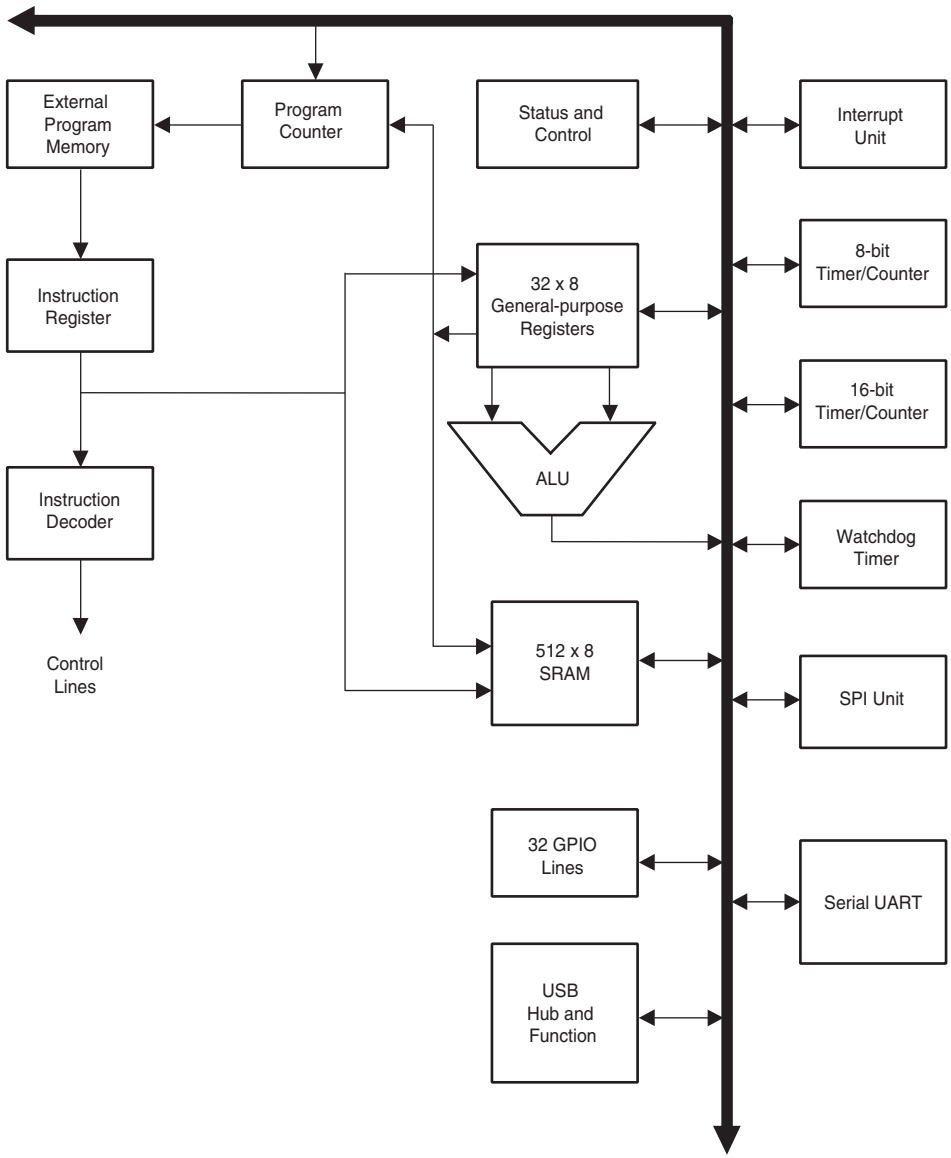
| Pin Number | Signal | Type |
|---|---|---|
| 63 | CEXT2 | O |
| 64 | DP2 | B |
| 65 | DM2 | B |
| 66 | DP3 | B |
| 67 | DM3 | B |
| 68 | DP4 | B |
| 69 | DM4 | B |
| 70 | PA0 | B |
| 71 | PA1 | B |
| 72 | PA2 | B |
| 73 | PA3 | B |
| 74 | PA4 | B |
| 75 | VSS | V |
| 76 | NC | – |
| 77 | PA5 | B |
| 78 | PA6 | B |
| 79 | PA7 | B |
| 80 | PB0 | B |
| 81 | PB1 | B |

| Pin Number | Signal | Type |
|---|---|---|
| 82 | PB2 | B |
| 83 | PB3 | B |
| 84 | VSS | V |
| 85 | PB4 | B |
| 86 | PB5 | B |
| 87 | PB6 | B |
| 88 | PB7 | B |
| 89 | PC0 | B |
| 90 | PC1 | B |
| 91 | PC2 | B |
| 92 | PC3 | B |
| 93 | PC4 | B |
| 94 | PC5 | B |
| 95 | PC6 | B |
| 96 | PC7 | B |
| 97 | PD0 | B |
| 98 | PD1 | B |
| 99 | VSS | V |
| 100 | NC | – |

## Signal Description

| Name | Type | Function |
|---|---|---|
| $V_{CC}$ | Power Supply/Ground | **5V Power Supply** |
| $V_{SS}$ | Power Supply/Ground | **Ground** |
| CEXT1, 2 | Power Supply/Ground | **External Capacitors for Power Supplies** – High quality 0.33 µF capacitors must be connected to CEXT1 and 2 for proper operation of the chip. |
| XTAL1 | Input | **Oscillator Input** – Input to the inverting oscillator amplifier. |
| XTAL2 | Output | **Oscillator Output** – Output of the inverting oscillator amplifier. |
| LFT | Input | **PLL Filter** – For proper operation of the PLL, this pin should be connected through a 0.01 µF capacitor in parallel with a 100Ω resistor in series with a 0.22 µF capacitor to ground (VSS). Both capacitors must be high quality ceramic. |
| DPO | Bi-directional | **Upstream Plus USB I/O** – This pin should be connected to CEXT1 through an external 1.5 kΩ. |
| DMO | Bi-directional | **Upstream Minus USB I/O** |
| DP[1:4] | Bi-directional | **Downstream Plus USB I/O** – Each of these pins should be connected to VSS through an external 15 kΩ resistor. DP[1:4] and DM[1:4] are the differential signal pin pairs to connect downstream USB devices. |
| DM[1:4] | Bi-directional | **Downstream Minus USB I/O** – Each of these pins should be connected to VSS through an external 15 kΩ resistor. |
| PA[0:7] | Bi-directional | **Port A[0:7]** – Bi-directional 8-bit I/O port with 4 mA drive strength. |
| PB[0:7] | Bi-directional | **Port B[0:7]** – Bi-directional 8-bit I/O port with 4 mA drive. PB[0,1,4:7] have dual functions as shown below:<br>**Port Pin** — **Alternate Function**<br>PB0 — T0, Timer/Counter0 External Input<br>PB1 — T1, Timer/Counter1 External Input<br>PB4 — SSN, SPI Slave Port Select or SCL, I2C Serial Bus Clock<br>PB5 — MOSI, SPI Slave Port Select Input<br>PB6 — MISO, SPI Master Data In, Slave Data Out<br>PB7 — SCK, SPI Master Clock Out, Slave Clock In |
| PC[0:7] | Bi-directional | **Port C[0:7]** – Bi-directional 8-bit I/O port with 4 mA drive strength. |
| PD[0:7] | Bi-directional | **Port D[0:7]** – Bi-directional I/O ports with 4 mA drive strength. PD[0:3,5] have dual functions as shown below:<br>**Port Pin** — **Alternate Function**<br>PD0 — RXD, Serial Input Port<br>PD1 — TXD, Serial Input Port<br>PD2 — INT0, External Interrupt 0<br>PD3 — INT1, External Interrupt 1<br>PD5 — OC1A Timer/Counter1 Output Compare A |
| TESTN | Input | **Test Pin** – This pin should be tied to ground. |
| SUSPEND | Output | **Suspend** – This pin is asserted when the AT43USB320A enters the Suspend status. In the 55618D, it is active low and in the 55618E and later versions, it is active high. |

**Figure 2.** The AT43USB320A Enhanced RISC Architecture



## Architectural Overview

The peripherals and features of the AT43USB320A microcontroller are similar to those of the AT90S8515, with the exception of the following modifications:

- External Program Memory
- No EEPROM
- No external data memory accesses
- No Analog Comparation
- Idle mode not supported
- USB Hub with attached function
- No internal pull-ups in the general-purpose I/O pin PA, PB, PC, PD

The embedded USB hardware of the AT43USB320A is a compound device, consisting of a 5 port hub with a permanently attached function on one port. The hub and attached function are two independent USB devices, each having its own device addresses and control endpoints. The hub has its dedicated interrupt endpoint, while the USB function has 2 additional programmable endpoints with separate 8-byte FIFOs.

The microcontroller always runs from a 12 MHz clock that is generated by the USB hardware. While the nominal and average period of this clock is 83.3 ns, it may have single cycles that deviate by ±20.8 ns during a phase adjustment by the SIE's clock/data separator of the USB hardware.

The microcontroller shares most of the control and status registers of the megaAVR™ Microcontroller Family. The registers for managing the USB operations are mapped into its SRAM space. The I/O section on page 16 summarizes the available I/O registers. The “AVR Register Set” on page 36 covers the AVR registers. Please refer to the Atmel AVR manual for more information.

The fast-access register file concept contains 32 x 8-bit general-purpose working registers with a single clock cycle access time. This means that during one single clock cycle, one Arithmetic Logic Unit (ALU) operation is executed. Two operands are output from the register file, the operation is executed, and the result is stored back in the register file – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing - enabling efficient address calculations. One of the three address pointers is also used as the address pointer for look-up tables in program memory. These added function registers are the 16-bit X-, Y- and Z-registers.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations are also executed in the ALU. Figure 2 on page 6 shows the AT43USB320A AVR Enhanced RISC microcontroller architecture.

In addition to the register operation, the conventional memory addressing modes can be used on the register file as well. This is enabled by the fact that the register file is assigned the 32 lowest Data Space addresses ($00 - $1 F), allowing them to be accessed as though they were ordinary memory locations.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, Timer/Counters, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the register file, $20 - $5F.

The AVR uses a Harvard architecture concept – with separate memories and buses for program and data. The program memory is executed with a single-level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is a downloadable SRAM or a mask programmed ROM.

With the relative jump and call instructions, the whole 24K address space is directly accessed. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM, and consequently, the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the Stack Pointer (SP) in the reset routine (before subroutines or interrupts are executed). The 10-bit SP is read/write accessible in the I/O space.

The 512-byte data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps. A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All interrupts have a separate interrupt vector in the interrupt vector table at the beginning of the program memory. The interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

## The General-purpose Register File

**Table 1.** AVR CPU General Purpose Working Register

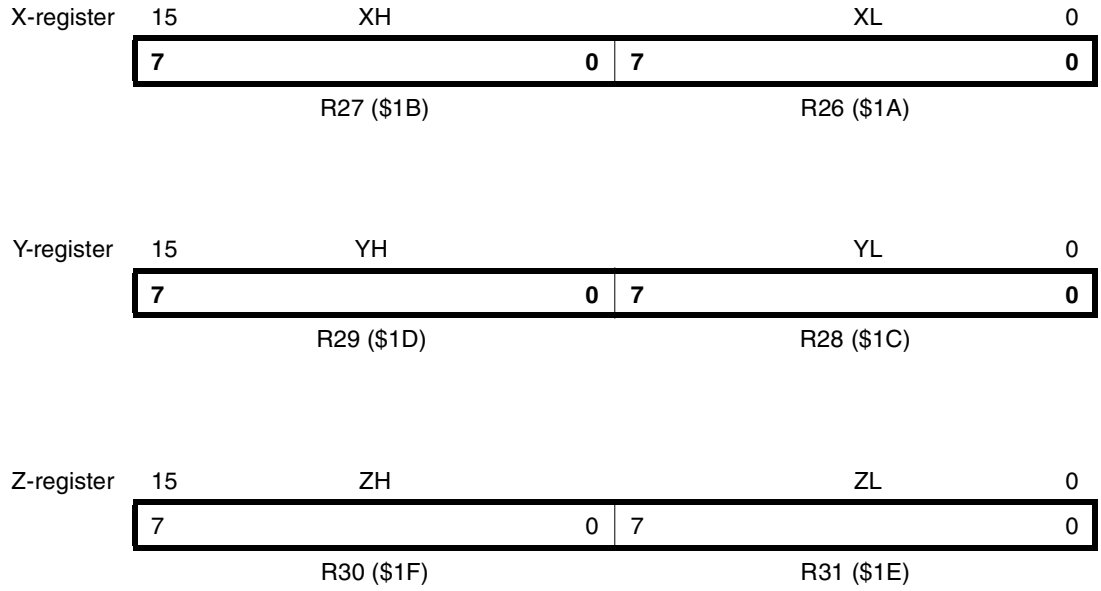
| Register | Address | Comment |
|---|---|---|
| R0 | $00 | |
| R1 | $01 | |
| R2 | $02 | |
| .. | | |
| R13 | $0D | |
| R14 | $0E | |
| R15 | $0F | |
| R16 | $10 | |
| R17 | $11 | |
| .. | | |
| R26 | $1A | X-register low byte |
| R27 | $1B | X-register high byte |
| R28 | $1C | Y-register low byte |
| R29 | $1D | Y-register high byte |
| R30 | $1E | Z-register low byte |
| R31 | $1F | Z-register high byte |

All register operating instructions in the instruction set have direct and single cycle access to all registers. The only exception is the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI, and ORI between a constant and a register, and the LDI instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file – R16..R31. The general SBC, SUB, CP, AND, and OR and all other operations between two registers or on a single register apply to the entire register file.

As shown in Table 1, 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-registers can be set to index any register in the file.

## X-, Y- and Z-Registers

Registers R26..R31 contain some added functions to their general-purpose usage. These registers are address pointers for indirect addressing of the Data Space. The three indirect address registers X, Y, and Z are defined as:




In the different addressing modes these address registers have functions as fixed displacement, automatic increment and decrement (see the descriptions for the different instructions).

## ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all 32 general purpose working registers. Within a single clock cycle, ALU operations between registers in the register file are executed. The ALU operations are divided into three main categories – arithmetic, logical and bit-functions.

## Program Memory

The AT43USB320A operates from an external program memory. Since all instructions are 16- or 32-bit words, the program memory is organized as X16. The AT43USB320A Program Counter (PC) is 16 bits wide, thus addressing the 64K program memory addresses.

Constant tables can be allocated within the entire program memory address space (see the LPM - Load Program Memory instruction description).

## SRAM Data Memory

Table 3 summarizes how the AT43USB320A SRAM Memory is organized. The lower 608 Data Memory locations address the Register file, the I/O Memory and the internal data SRAM. The first 96 locations address the Register File + I/O Memory, and the next 512 locations address the internal data SRAM. The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement and Indirect with Post-increment. In the register file, registers R26 to R31 feature the indirect addressing pointer registers. Direct addressing reaches the entire data space.

The Indirect with Displacement mode features 63 address locations that reach from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented and incremented.

The 32 general purpose working registers, 64 I/O registers and the 1024 bytes of internal data SRAM in the AT43USB320A are all accessible through these addressing modes.

To manage the USB hardware, a special set of registers is assigned. These registers are mapped to SRAM space between addresses $1F00 and 1FFF. Table 3 and Table 4 give an overview of these registers.

**Table 2.** SRAM Organization

| Register File | | Data Address Space |
|---|---|---|
| R0 | | $0000 |
| R1 | | $0001 |
| | | |
| R30 | | $001E |
| R31 | | $001F |

I/O Registers

| | | |
|---|---|---|
| $00 | | $0020 |
| $01 | | $0021 |
| | | |
| $3E | | $005E |
| $3F | | $005F |

Internal SRAM

| |
|---|
| $0060 |
| $0061 |
| |
| $025E |
| $045F |

USB Registers

| |
|---|
| $1F00 |
| |
| $1FFE |
| $1FFF |

**Table 3.** USB Hub and Function Registers

| Address | Name | Function |
|---|---|---|
| $1FFD | FRM_NUM_H | Frame Number High Register |
| $1FFC | FRM_NUM_L | Frame Number Low Register |
| $1FFB | GLB_STATE | Global State Register |
| $1FFA | SPRSR | Suspend/Resume Register |
| $1FF9 | SPRSIE | Suspend/Resume Interrupt Enable Register |
| $1FF7 | UISR | USB Interrupt Status Register |
| $1FF5 | UIAR | USB Interrupt Acknowledge Register |
| $1FF3 | UIER | USB Interrupt Enable Register |
| $1FF2 | UOVCER | Overcurrent Detect Register |
| $1FEF | HADDR | Hub Address Register |
| $1FEE | FADDR | Function Address Register |
| $1FE7 | HENDP0_CNTR | Hub Endpoint 0 Control Register |
| $1FE5 | FENDP0_CNTR | Function Endpoint 0 Control Register |
| $1FE4 | FENDP1_CNTR | Function Endpoint 1 Control Register |
| $1FE3 | FENDP2_CNTR | Function Endpoint 2 Control Register |
| $1FDF | HCSR0 | Hub Controller Endpoint 0 Service Routine Register |
| $1FDD | FCSR0 | Function Controller Endpoint 0 Service Routine Register |
| $1FDC | FCSR1 | Function Controller Endpoint 1 Service Routine Register |
| $1FDB | FCSR2 | Function Controller Endpoint 2 Service Routine Register |
| $1FD7 | HDR0 | Hub Endpoint 0 FIFO Data Register |
| $1FD5 | FDR0 | Function Endpoint 0 FIFO Data Register |
| $1FD4 | FDR1 | Function Endpoint 1 FIFO Data Register |
| $1FD3 | FDR2 | Function Endpoint 2 FIFO Data Register |
| $1FCF | HBYTE_CNT0 | Hub Endpoint 0 Byte Count Register |
| $1FCD | FBYTE_CNT0 | Function Endpoint 0 Byte Count Register |
| $1FCC | FBYTE_CNT1 | Function Endpoint 1 Byte Count Register |
| $1FCB | FBYTE_CNT2 | Function Endpoint 2 Byte Count Register |
| $1FC7 | HSTR | Hub Status Register |
| $1FC5 | HPCON | Hub Port Control Register |
| $1FBC | HPSTAT5 | Hub Port 5 Status Register |
| $1FBB | HPSTAT4 | Hub Port 4 Status Register |
| $1FBA | HPSTAT3 | Hub Port 3 Status Register |
| $1FB9 | HPSTAT2 | Hub Port 2 Status Register |
| $1FB8 | HPSTAT1 | Hub Port 1 Status Register |
| $1FB4 | HPSCR5 | Hub Port 5 Status Change Register |

**Table 3.** USB Hub and Function Registers (Continued)

| Address | Name | Function |
|---|---|---|
| $1FB3 | HPSCR4 | Hub Port 4 Status Change Register |
| $1FB2 | HPSCR3 | Hub Port 3 Status Change Register |
| $1FB1 | HPSCR2 | Hub Port 2 Status Change Register |
| $1FB0 | HPSCR1 | Hub Port 1 Status Change Register |
| $1FAC | PSTATE5 | Hub Port 5 Bus State Register |
| $1FAB | PSTATE4 | Hub Port 4 Bus State Register |
| $1FAA | PSTATE3 | Hub Port 3 Bus State Register |
| $1FA9 | PSTATE2 | Hub Port 2 Bus State Register |
| $1FA8 | PSTATE1 | Hub Port 1 Bus State Register |
| $1FA7 | HCAR0 | Hub Endpoint 0 Control and Acknowledge Register |
| $1FA5 | FCAR0 | Function Endpoint 0 Control and Acknowledge Register |
| $1FA4 | FCAR1 | Function Endpoint 1 Control and Acknowledge Register |
| $1FA3 | FCAR2 | Function Endpoint 2 Control and Acknowledge Register |

**Table 4.** USB Hub and Function Registers

| Name | Address | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| GLB_STATE | $1FFB | – | | | SUSP FLG | RESUME FLG | RMWUPE | CONFG | HADD EN |
| SPRSR | $1FFA | – | – | – | – | – | FRWUP | RSM | GLB SUSP |
| SPRSIE | $1FF9 | – | – | – | – | – | FRWUP IE | RSM IE | GLB SUSP IE |
| UISR | $1FF7 | SOF INT | EOF2 INT | – | FEP3 INT | HEP0 INT | FEP2 INT | FEP1 INT | FEP0 INT |
| UIAR | $1FF5 | SOF INTACK | EOF2 INTACK | – | FEP3 INTACK | HEP0 INTACK | FEP2 INTACK | FEP1 INTACK | FEP0 INTACK |
| UIER | $1FF3 | SOF IE | EOF2 IE | – | FEP3 IE | HEP0 IE | FEP2 IE | FEP1 IE | FEP0 IE |
| HADDR | $1FEF | SAEN | HADD6 | HADD5 | HADD4 | HADD3 | HADD2 | HADD1 | HADD0 |
| FADDR | $1FEE | FEN | FADD6 | FADD5 | FADD4 | FADD3 | FADD2 | FADD1 | FADD0 |
| HENDP0_CNTR | $1FE7 | EPEN | – | – | – | DTGLE | EPDIR | EPTYPE1 | EPTYPE0 |
| FENDP0_CNTR | $1FE5 | EPEN | – | – | – | DTGLE | EPDIR | EPTYPE1 | EPTYPE0 |
| FENDP1_CNTR | $1FE4 | EPEN | – | – | – | DTGLE | EPDIR | EPTYPE1 | EPTYPE0 |
| FENDP2_CNTR | $1FE3 | EPEN | – | – | – | DTGLE | EPDIR | EPTYPE1 | EPTYPE0 |
| HCSR0 | $1FDF | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX CEMPLETE |
| FCSR0 | $1FDD | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX COMPLETE |
| FCSR1 | $1FDC | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX COMPLETE |
| FCSR2 | $1FDB | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX COMPLETE |
| HDR0 | $1FD7 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 |
| FDR0 | $1FD5 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 |
| FDR1 | $1FD4 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 |
| FDR2 | $1FD3 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 |
| HBYTE_CNT0 | $1FCF | – | – | – | BYTCT4 | BYTCT3 | BYTCT2 | BYTCT1 | BYTCT0 |
| FBYTE_CNT0 | $1FCD | – | – | – | BYTCT4 | BYTCT3 | BYTCT2 | BYTCT1 | BYTCT0 |
| FBYTE_CNT1 | $1FCC | – | – | – | BYTCT4 | BYTCT3 | BYTCT2 | BYTCT1 | BYTCT0 |
| FBYTE_CNT2 | $1FCB | – | – | – | BYTCT4 | BYTCT3 | BYTCT2 | BYTCT1 | BYTCT0 |
| HSTR | $1FC7 | – | – | – | – | OVLSC | LPSC | OVI | LPS |
| HPCON | $1FC5 | – | HPCON2 | HPCON1 | HPCON0 | – | HPADD2 | HPADD1 | HPADD0 |
| HPSTAT5 | $1FBC | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT |
| HPSTAT4 | $1FBB | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT |
| HPSTAT3 | $1FBA | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT |
| HPSTAT2 | $1FB9 | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT |
| HPSTAT1 | $1FB8 | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT |
| HPSCR5 | $1FB4 | – | – | – | RSTSC | POCIC | PSSC | PESC | PCSC |
| HPSCR4 | $1FB3 | – | – | – | RSTSC | POCIC | PSSC | PESC | PCSC |
| HPSCR3 | $1FB2 | – | – | – | RSTSC | POCIC | PSSC | PESC | PCSC |
| HPSCR2 | $1FB1 | – | – | – | RSTSC | POCIC | PSSC | PESC | PCSC |
| HPSCR1 | $1FB0 | – | – | – | RSTSC | POCIC | PSSC | PESC | PCSC |
| PSTATE5 | $1FAC | – | – | – | – | – | – | DPSTATE | DMSTATE |
| PSTATE4 | $1FAB | – | – | – | – | – | – | DPSTATE | DMSTATE |
| PSTATE3 | $1FAA | – | – | – | – | – | – | DPSTATE | DMSTATE |
| PSTATE2 | $1FA9 | – | – | – | – | – | – | DPSTATE | DMSTATE |
| PSTATE1 | $1FA8 | – | – | – | – | – | – | DPSTATE | DMSTATE |
| HCAR0 | $1FA7 | CTL DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_SENT-ACK | RX_SETUP_ACK | RX_OUT_PACKET_ACK | TX_COMPLETE-ACK |

**Table 4.** USB Hub and Function Registers (Continued)

| Name | Address | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| FCAR0 | $1FA5 | CTL DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_SENT-ACK | RX_SETUP_ACK | RX_OUT_PACKET_ACK | TX_COMPLETE-ACK |
| FCAR1 | $1FA4 | CTL DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_SENT-ACK | RX_SETUP_ACK | RX_OUT_PACKET_ACK | TX_COMPLETE-ACK |
| FCAR2 | $1FA3 | CTL DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_SENT-ACK | RX_SETUP_ACK | RX_OUT_PACKET_ACK | TX_COMPLETE-ACK |

## I/O Memory

The I/O space definition of the AT43USB320A is shown in the following table:

**Table 5.** I/O Memory Space

| I/O (SRAM) Address | Name | Function |
|---|---|---|
| $3F ($5F) | SREG | Status Register |
| $3E ($5E) | SPH | Stack Pointer High |
| $3D ($5D) | SPL | Stack Pointer Low |
| $3B ($5B) | GIMSK | General Interrupt Mask Register |
| $3A ($5A) | GIFR | General Interrupt Flag Register |
| $39 ($59) | TIMSK | Timer/Counter Interrupt Mask Register |
| $38 ($58) | TIFR | Timer/Counter Interrupt Mask Register |
| $35 ($55) | MCUCR | MCU General Control Register |
| $33 ($53) | TCCR0 | Timer/Counter0 Control Register |
| $32 ($52) | TCNT0 | Timer/Counter0 (8 bit) |
| $2F ($4F) | TCCR1A | Timer/Counter1 Control Register A |
| $2E ($4E) | TTCR1B | Timer/Counter0 Control Register B |
| $2D ($52) | TCNT1H | Timer/Counter1 High Byte |
| $2C ($52) | TCNT1L | Timer/Counter0 Low Byte |
| $2B ($4B) | OCR1AH | Timer/Counter1 Output Compare Register A High Byte |
| $2A ($4A) | OCR1AL | Timer/Counter1 Output Compare Register A Low Byte |
| $29 ($49) | OCR1BH | Timer/Counter1 Output Compare Register B High Byte |
| $28 ($48) | OCR1BL | Timer/Counter1 Output Compare Register B Low Byte |
| $25 ($45) | ICR1H | T/C 1 Input Capture Register High Byte |
| $24 ($44) | ICR1L | T/C 1 Input Capture Register Low Byte |
| $21 ($41) | WDTCR | Watchdog Timer Counter Register |
| $1B ($4B) | PORTA | Data Register, Port A |
| $1A ($3A) | DDRA | Data Direction Register, Port A |
| $19 ($39) | PINA | Input Pins, Port A |
| $18 ($38) | PORTB | Data Register, Port B |
| $17 ($37) | DDRB | Data Direction Register, Port B |
| $16 ($36) | PINB | Input Pins, Port B |
| $13($33) | PORTC | Data Register, Port C |
| $12 ($32) | PORTD | Data Register, Port D |
| $11 ($31) | DDRD | Data Direction Register, Port D |
| $10 ($30) | PIND | Input Pins, Port D |
| $0B (2B) | USR | UART Status Register |
| $0A (2A) | UCR | UART Control Register |
| $09 (29) | UBRR | UART Baud Rate Register |

All AT43USB320A I/O and peripherals, except for the USB hardware registers, are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range $00 – $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set documentations of the AVR for more details. When using the I/O specific commands, IN and OUT, the I/O address $00 – $3F must be used. When addressing I/O registers as SRAM, $20 must be added to this address. All I/O register addresses throughout this document are shown with the SRAM address in parentheses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

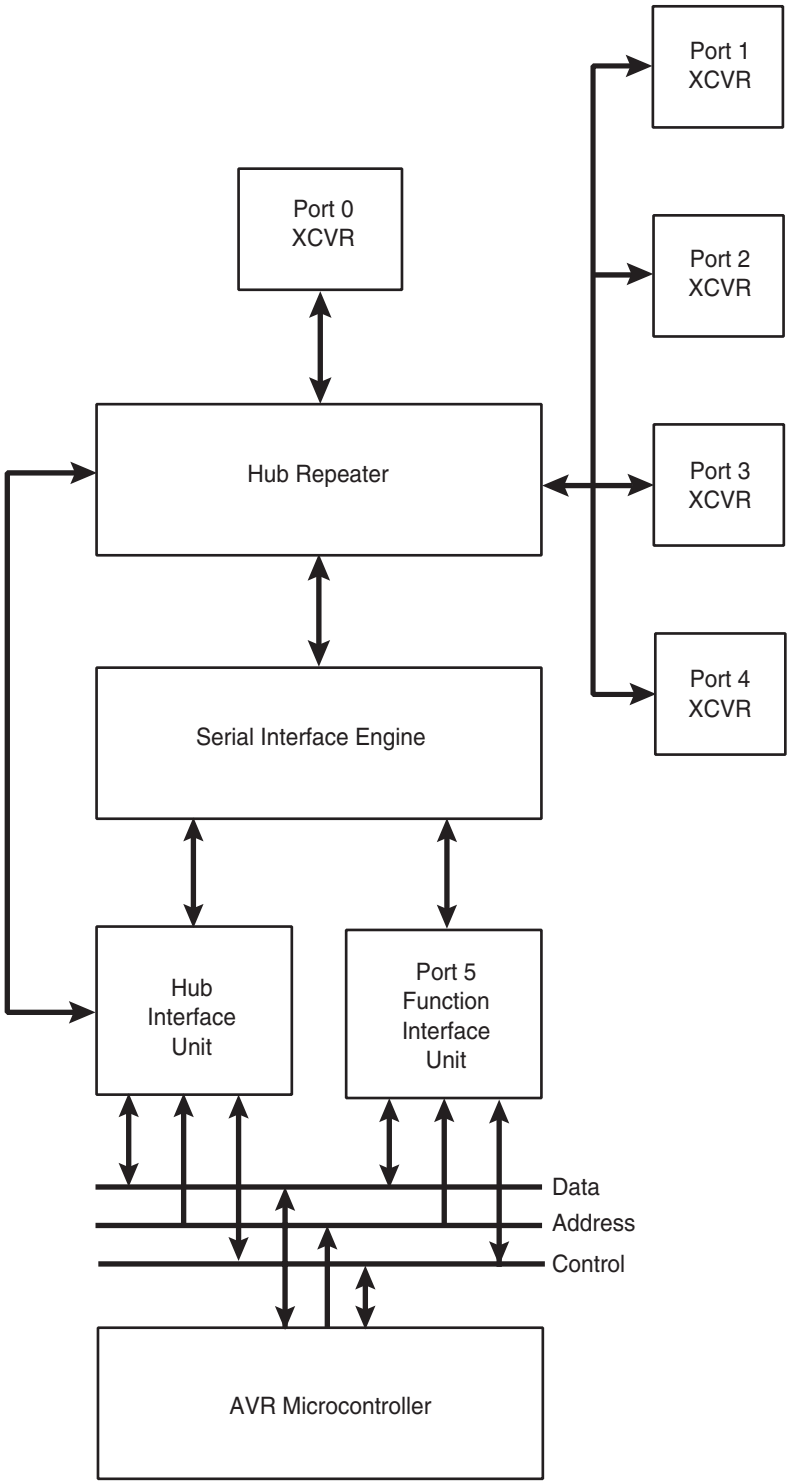
## USB Hub

A block diagram of the USB hardware of the AT43USB320A is shown in Figure 3. The USB hub of the AT43USB320A has 5 downstream ports. The embedded function is permanently attached to Port 5. Ports 1 through 4 are available as external ports. The actual number of ports used is strictly defined by the firmware of the AT43USB320A and can vary from 0 to 4. Because the exact configuration is defined by firmware, ports 1 to 4 may even function as permanently attached ports as long as the Hub Descriptor identifies them as such.

### USB Function

The embedded USB function has its own device address and has a default endpoint plus 2 other programmable endpoints with 8-byte FIFOs. Endpoints 1 - 3 can be programmed as interrupt IN or OUT or bulk IN or OUT endpoints.

**Figure 3.** USB Hardware



## Functional Description

### On-chip Power Supply

The AT43USB320A contains two on-chip power supplies that generate 3.3V with a capacity of 30 mA each from the 5V power input. The on-chip power supplies are intended to supply the AT43USB320A internal circuit and the 1.5K pull-up resistor only and should not be used for other purposes. External 0.33 µF filter capacitors are required at the power supply outputs, CEXT1 and 2. The internal power supplies can be disabled as described in the next paragraph.

The user should be careful when the GPIO pins are required to supply high-load currents. If the application requires that the GPIO supply currents beyond the capability of the on-chip power supply, the AT43USB320A should be supplied by an external 3.3V power supply. In this case, the 5V $V_{CC}$ power supply pin should be left unconnected and the 3.3V power supplied to the chip through the CEXT1 and 2 pins.

### I/O Pin Characteristics

The I/O pins of the AT43USB320A should not be directly connected to voltages less than $V_{SS}$ or more than the voltage at the CEXT pins. If it is necessary to violate this rule, insert a series resistor between the I/O pin and the source of the external signal source that limits the current into the I/O pin to less than 2 mA. Under no circumstance should the external voltage exceed 5.5V. To do so will put the chip under excessive stress.

### Oscillator and PLL

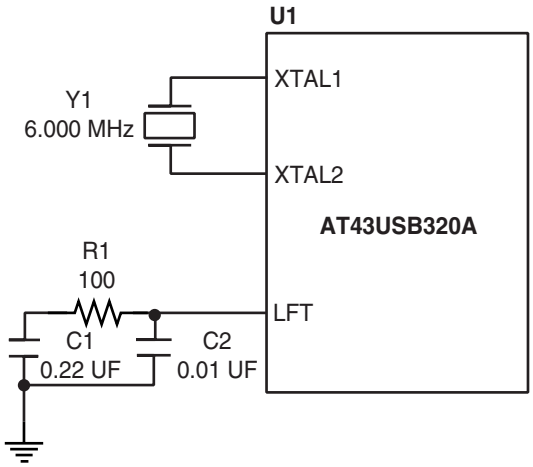
All clock signals required to operate the AT43USB320A are derived from an on-chip oscillator. To reduce EMI and power dissipation, the oscillator is designed to operate with a 6 MHz crystal. An on-chip PLL generates the high frequency for the clock/data separator of the Serial Interface Engine. In the suspended state, the oscillator circuitry is turned off.

The oscillator of the AT43USB320A is a special, low-drive type, designed to work with most crystals without any external components. The crystal must be of the parallel resonance type requiring a load capacitance of about 10 pF. If the crystal requires a higher value capacitance, external capacitors can be added to the two terminals of the crystal and ground to meet the required value. To assure quick start-up, a crystal with a high Q, or low ESR, should be used. To meet the USB hub frequency accuracy and stability requirements for hubs, the crystal should have an accuracy and stability of better than 100 PPM. The use of a ceramic resonator in place of the crystal is not recommended because a resonator would not have the necessary frequency accuracy and stability.

The clock can also be externally sourced. In this case, connect the clock source to the XTAL1 pin, while leaving XTAL2 pin floating. The switching level at the OSC1 pin can be as low as 0.47V and a CMOS device is required to drive this pin to maintain good noise margins at the low switching level.

For proper operation of the PLL, an external RC filter consisting of a series RC network of 100Ω and 0.22 µF in parallel with a 0.01 µF capacitor must be connected from the LFT pin to $V_{SS}$. Use only high-quality ceramic capacitors.

**Figure 4.** Oscillator and PLL




## Reset and Interrupt Handling

The AT43USB320A provides 22 different interrupt sources with 13 separate reset vectors, each with a separate program vector in the program memory space. Eleven of the interrupt sources share 2 interrupt reset vectors. These 11 are the USB related interrupts. All interrupts are assigned individual enable bits which must be set (one) together with the I-bit in the status register in order to enable the interrupt.

The lowest addresses in the program memory space are automatically defined as the Reset and Interrupt vectors. The complete list of vectors is shown in Table 6. The list also determines the priority levels of the different interrupts. The lower the address, the higher is the priority level. RESET has the highest priority, and next is INT0 – the USB Suspend and Resume Interrupt, etc.

**Table 6.** Reset and Interrupt Vectors

| Vector No. | Program Address | Source | Interrupt Definition |
|---|---|---|---|
| 1 | $000 | RESET | External Reset, Power-on Reset and Watchdog Reset |
| 2 | $002 | INT0 | USB Suspend and Resume |
| 3 | $004 | INT1 | External Interrupt Request 1 |
| 4 | $006 | TIMER1 CAPT | Timer/Counter1 Capture Event |
| 5 | $008 | TIMER1 COMPA | Timer/Counter1 Compare Match A |
| 6 | $00A | TIMER1 COMPB | Timer/Counter1 Compare Match B |
| 7 | $00C | TIMER1, OVF | Timer/Counter1 Overflow |
| 8 | $00E | TIMER0, OVF | Timer/Counter0 Overflow |
| 9 | $010 | SPI, STC | SPI Serial Transfer Complete |
| 10 | $012 | UART RX | UART RX Complete |
| 11 | $014 | UART UDRE | UART RX Data Receiver Output |
| 12 | $016 | UART TX | UART TX Complete |
| 13 | $018 | USB HW | USB Hardware |

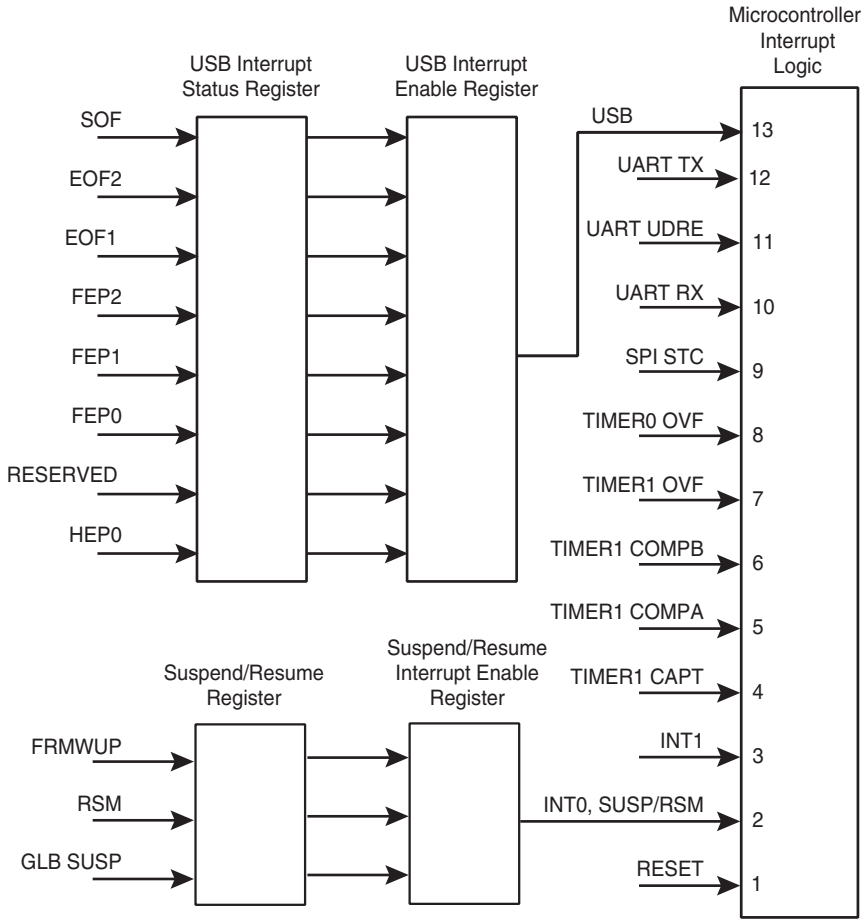
The most typical and general program setup for the Reset and Interrupt Vector Addresses are:

```
Address   Labels   Code                              Comments
$000               jmp                    RESET     ; Reset Handler
$004               jmp                    EXT_INT1  ; IRQ1 Handler
$00E               jmp                    TIM0_OVF  ; Timer0 Overflow
                                                     Handler
$018               jmp                    USB_HW    ; USB Handler
;
$00d      MAIN:    ldi r16, high (RAMEND)           ; Main Program
start
$00e               out SPH, r16
$00f               ldi r16, low (RAMEND)
$010               out SPL, r16
$011               <instr> xxx


...       ...      ...                    ...
```

USB related interrupt events are routed to reset vectors 13 and 2 through a separate set of interrupt, interrupt enable and interrupt mask registers that are mapped to the data SRAM space. These interrupts must be enabled though their control register bits. In the event an interrupt is generated, the source of the interrupt is identified by reading the interrupt registers. The USB frame and transaction related interrupt events, such as Start of Frame interrupt, are grouped in one set of registers: USB Interrupt Flag Register and USB Interrupt Enable Register. The USB Bus reset and suspend/resume are grouped in another set of registers: Suspend/Resume Register and Suspend/Resume Interrupt Enable Register.

**Figure 5.** AT43USB320A Interrupt Structure




## Reset Sources

The AT43USB320A has four sources of reset:

- **Power-on Reset** – The MCU is reset when the supply voltage is below the power-on reset threshold.
- **External Reset** – The MCU is reset when a low level is present on the RESET pin for more than 50 ns.
- **Watchdog Reset** – The MCU is reset when the watchdog timer period expires and the watchdog is enabled.
- **USB Reset** – A USB bus reset is defined as a SE0 (single ended zero) of at least 4 slow speed USB clock cycles received by Port0. The internal reset pulse to the USB hardware and microcontroller lasts for 24 oscillator periods.

When the USB hardware is reset, the compound device is de-configured and has to be re-enumerated by the host. When the microcontroller is reset, all I/O registers are then set to their initial values, and the program starts execution from address $000. The instruction placed in address $000 must be a JMP instruction to the reset handling routine. If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. The circuit diagram in Figure 6 shows the reset logic. The user can select the start-up time according to typical oscillator start-up. The number of WDT oscillator cycles used for each time-out is shown in Table 7.

**Figure 6.** Reset Logic

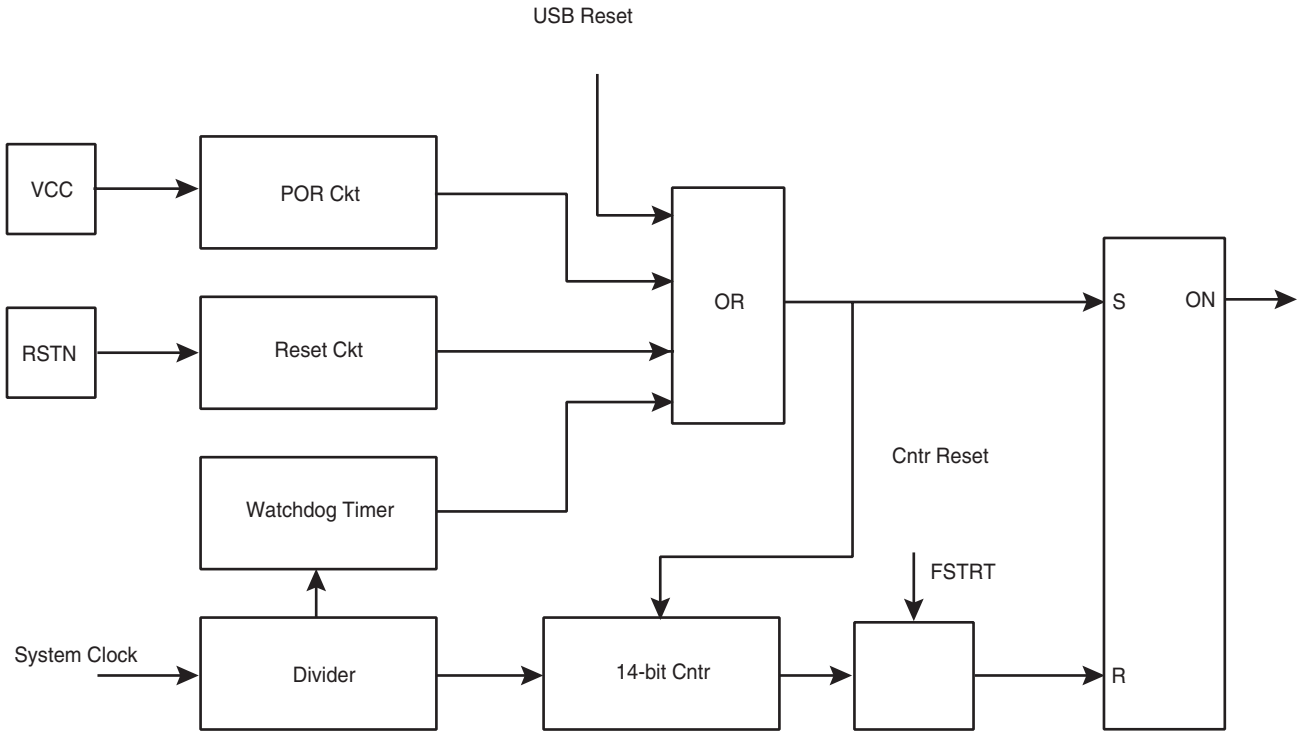



**Table 7.** Number of Watchdog Oscillator Cycles

| FSTRT | Time-out at $V_{CC}$ = 5V | Number of WDT cycles |
|---|---|---|
| Programmed | 1.1 ms | 1K |
| Unprogrammed | 16.0 ms | 16K |

## Power-on Reset

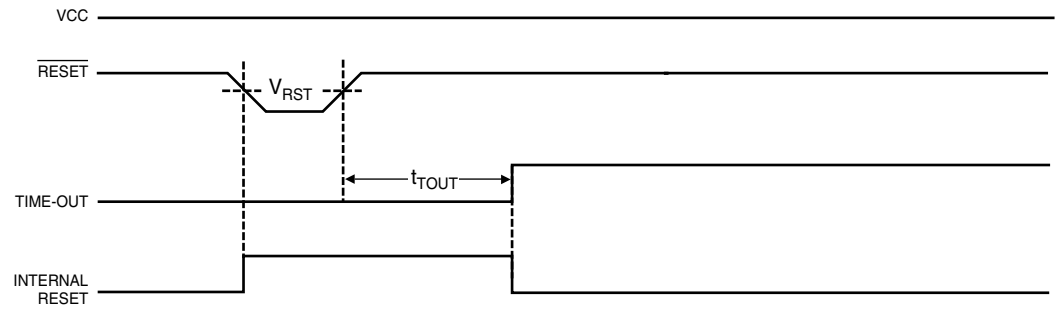
A Power-on Reset (POR) circuit ensures that the device is reset from power-on. An internal timer clocked from the Watchdog timer oscillator prevents the MCU from starting until after a certain period after $V_{CC}$ has reached the power-on threshold voltage, regardless of the $V_{CC}$ rise time.

If the build-in start-up delay is sufficient, RESET can be connected to $V_{CC}$ directly or via an external pull-up resistor. By holding the pin low for a period after $V_{CC}$ has been applied, the Power-on Reset period can be extended.

## External Reset

An external reset is generated by a low-level on the RESET pin. Reset pulses longer than 200 ns will generate a reset. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage - $V_{RST}$ on its positive edge, the delay timer starts the MCU after the Time-out period $t_{TOUT}$ has expired.

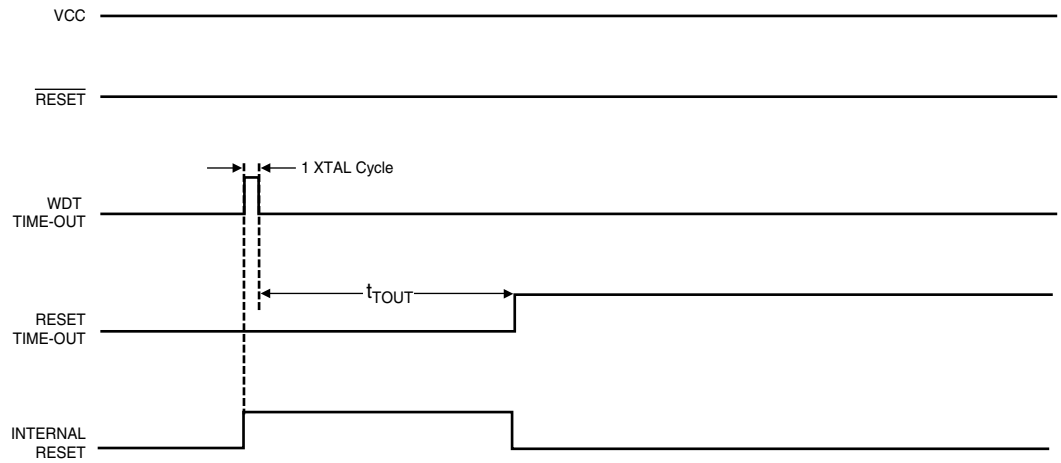
**Figure 7.** External Reset During Operation




## Watchdog Timer Reset

When the watchdog times out, it will generate a short reset pulse of 1 XTAL cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period $t_{TOUT}$.

**Figure 8.** Watchdog Reset During Operation




## Non-USB Related Interrupt Handling

The AT43USB320A has two non-USB 8-bit Interrupt Mask control registers; GIMSK (General Interrupt Mask Register) and TIMSK (Timer/Counter Interrupt Mask Register).

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared (zero) and all interrupts are disabled. The user software can set (one) the I-bit to enable nested interrupts. The I-bit is set (one) when a Return from Interrupt instruction, RETI, is executed.

For Interrupts triggered by events that can remain static (e.g. the Output Compare register1 matching the value of Timer/Counter1) the interrupt flag is set when the event occurs. If the interrupt flag is cleared and the interrupt condition persists, the flag will not be set until the event occurs the next time.

When the Program Counter is vectored to the actual interrupt vector in order to execute the interrupt handling routine, hard-ware clears the corresponding flag that generated the interrupt. Some of the interrupt flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared.

If an interrupt condition occurs when the corresponding interrupt enable bit is cleared (zero), the interrupt flag will be set and remembered until the interrupt is enabled, or the flag is cleared by software.

If one or more interrupt conditions occur when the global interrupt enable bit is cleared (zero), the corresponding interrupt flag(s) will be set and remembered until the global interrupt enable bit is set (one), and will be executed by order of priority.

Note that external level interrupt does not have a flag, and will only be remembered for as long as the interrupt condition is active.

### *General Interrupt Mask Register – GIMSK*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $3B ($5B) | INT1 | INT0 | – | – | – | – | – | – | GIMSK |
| Read/Write | R/W | R/W | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control1 bits 1/0 (ISC11 and ISC10) in the MCU general Control Register (MCUCR) defines whether the external interrupt is activated on rising or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of External Interrupt Request 1 is executed from program memory address $004. See also “External Interrupts” on page 29.

- **Bit 6 – INT0: Interrupt Request 0 (Suspend/Resume Interrupt) Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control0 bits 1/0 (ISC01 and ISC00) in the MCU general Control Register (MCUCR) defines whether the external interrupt is activated on rising or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of Interrupt Request 0 is executed from program memory address $002. See also “External Interrupts” on page 29.

- **Bits 5..0 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and always read as zero.

### *General Interrupt Flag Register – GIFR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $3A ($5A) | INTF1 | INT F0 | – | – | – | – | – | – | GIFR |
| Read/Write | R/W | R/W | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – INTF1: External Interrupt Flag1**

When an event on the INT1 pin triggers an interrupt request, INTF1 becomes set (one). If the I-bit in SREG and the INT1 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $004. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 6 – INTF0: Interrupt Flag0 (Suspend/Resume Interrupt Flag)**

When an event on the INT0 (that is, a USB event-related interrupt) triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in GIMSK are set (one), the MCU will jump to the interrupt vector at address $002. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bits 5..0 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and always read as zero.

### *Timer/Counter Interrupt Mask Register – TIMSK*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $39 ($59) | TOIE1 | OCIE1A | OCIE1NB | – | TICIE1 | – | TOIE0 | – | TIMSK |
| Read/Write | R/W | R/W | R/W | R | R/W | R | R/W | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – TOIE1: Timer/Counter1 Overflow Interrupt Enable**

When the TOIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 Overflow interrupt is enabled. The corresponding interrupt (at vector $006) is executed if an overflow in Timer/Counter1 occurs, i.e., when the TOV1 bit is set in the Timer/Counter Interrupt Flag Register (TIFR).

- **Bit 6 – OCE1A: Timer/Counter1 Output CompareA Match Interrupt Enable**

When the OCIE1A bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareA Match interrupt is enabled. The corresponding interrupt (at vector $004) is executed if a CompareA match in Timer/Counter1 occurs, i.e., when the OCF1A bit is set in the TIFR.

- **Bit 5 – OCIE1B: Timer/Counter1 Output CompareB Match Interrupt Enable**

When the OCIE1B bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 CompareB Match interrupt is enabled. The corresponding interrupt (at vector $005) is executed if a CompareB match in Timer/Counter1 occurs, i.e., when the OCF1B bit is set in the TIFR.

- **Bit 4 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

- **Bit 3 – TICIE1: Timer/Counter1 Input Capture Interrupt Enable**

When the TICIE1 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter1 Input Capture Event Interrupt is enabled. The corresponding interrupt (at vector $003) is executed if a capture-triggering event occurs on pin 31, ICP, i.e., when the ICF1 bit is set in the TIFR.

- **Bit 2 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

- **Bit 1 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is set (one) and the I-bit in the Status Register is set (one), the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt (at vector $007) is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the TIFR.

- **Bit 0 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

***Timer/Counter Interrupt Flag Register – TIFR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $38 ($58) | TOV1 | OCF1A | OCIFB | – | ICF1 | – | TOV0 | – | TIFR |
| Read/Write | R/W | R/W | R/W | R | R/W | R | R/W | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – TOV1: Timer/Counter1 Overflow Flag**

The TOV1 is set (one) when an overflow occurs in Timer/Counter1. TOV1 is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, TOV1 is cleared by writing a logic one to the flag. When the I-bit in SREG, and TOIE1 (Timer/Counter1 Overflow Interrupt Enable), and TOV1 are set (one), the Timer/Counter1 Overflow Interrupt is executed. In PWM mode, this bit is set when Timer/Counter1 changes counting direction at $0000.

- **Bit 6 – OCF1A: Output Compare Flag 1A**

The OCF1A bit is set (one) when compare match occurs between the Timer/Counter1 and the data in OCR1A - Output Compare Register 1A. OCF1A is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, OCF1A is cleared by writing a logic one to the flag. When the I-bit in SREG, and OCIE1A (Timer/Counter1 Compare match InterruptA Enable), and the OCF1A are set (one), the Timer/Counter1 Compare A match Interrupt is executed.

- **Bit 5 – OCF1B: Output Compare Flag 1B**

The OCF1B bit is set (one) when compare match occurs between the Timer/Counter1 and the data in OCR1B - Output Compare Register 1B. OCF1B is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, OCF1B is cleared by writing a logic one to the flag. When the I-bit in SREG, and OCIE1B (Timer/Counter1 Compare match InterruptB Enable), and the OCF1B are set (one), the Timer/Counter1 Compare B match Interrupt is executed.

- **Bit 4 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

- **Bit 3 – ICF1: - Input Capture Flag 1**

The ICF1 bit is set (one) to flag an input capture event, indicating that the Timer/Counter1 value has been transferred to the input capture register - ICR1. ICF1 is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, ICF1 is cleared by writing a logic one to the flag. When the SREG I-bit, and TICIE1 (Timer/Counter1 Input Capture Interrupt Enable), and ICF1 are set (one), the Timer/Counter1 Capture Interrupt is executed.

- **Bit 2 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

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

The bit TOV0 is set (one) when an overflow occurs in Timer/Counter0. TOV0 is cleared by the 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, and TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set (one), the Timer/Counter0 Overflow interrupt is executed.

- **Bit 0 – Res: Reserved Bit**

This bit is a reserved bit in the AT43USB320A and always reads zero.

## External Interrupts

The external interrupts are triggered by the INT0 and INT1 pins. Observe that, if enabled, the INT0/INT1 interrupt will trigger even if the INT0/INT1 pins are configured as outputs. This feature provides a way of generating a software interrupt. The external interrupts can be triggered by a falling or rising edge or a low level. This is set up as indicated in the specification for the MCU Control Register (MCUCR) and the Interrupt Sense Control Register (ISCR). When the external interrupt is enabled and is configured as level triggered, the interrupt will trigger as long as the pin is held low. The external interrupts are set up as described in the specification for the MCU Control Register (MCUCR).

## Interrupt Response Time

The interrupt execution response for all the enabled AVR interrupts is 4 clock cycles minimum. 4 clock cycles after the interrupt flag has been set, the program vector address for the actual interrupt handling routine is executed. During this 4 clock cycle period, the Program Counter (2 bytes) is pushed onto the Stack, and the Stack Pointer is decremented by 2. The vector is normally a jump to the interrupt routine, and this jump takes 3 clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served.

A return from an interrupt handling routine (same as for a subroutine call routine) takes 4 clock cycles. During these 4 clock cycles, the Program Counter (2 bytes) is popped back from the Stack, the Stack Pointer is incremented by 2, and the I flag in SREG is set. When the AVR exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

*MCU Control Register – MCUCR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $35 ($55) | – | – | SE | SM | ISC11 | ISC10 | ISC01 | ISC00 | MCUCR |
| Read/Write | R | R | 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 – Res: Reserved Bits**
- **Bit 5 – SE: Sleep Enable**

The SE bit must be set (1) 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 set the Sleep Enable SE bit just before the execution of the SLEEP instruction.

- **Bit 4 – SM: Sleep Mode**

This bit selects between the two available sleep modes. When SM is cleared (zero), Idle Mode is selected as Sleep Mode. When SM is set (1), Power Down mode is selected as sleep mode. The AT43USB320A does not support the Idle Mode and SM should always be set to one when entering the Sleep Mode.

- **Bit 3, 2 – ISC11, ISC10: 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 in the GIMSK is set. The level and edges on the external INT1 pin that activate the interrupt are defined in the following table:

**Table 8.** INT1 Sense Control

| ISC11 | ISC10 | Description |
|---|---|---|
| 0 | 0 | The low level of INT1 generates an interrupt request. |
| 0 | 1 | Reserved. |
| 1 | 0 | The falling edge of INT1 generates an interrupt request. |
| 1 | 1 | The rising edge of INT1 generates an interrupt request. |

- **Bit 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 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 in the GIMSK is set. The level and edges on the external INT1 pin that activate the interrupt are defined in the following table:

**Table 9.** INT1 Sense Control

| ISC01 | ISC00 | Description |
|---|---|---|
| 0 | 0 | The low level of INT1 generates an interrupt request. |
| 0 | 1 | Reserved. |
| 1 | 0 | The falling edge of INT1 generates an interrupt request. |
| 1 | 1 | The rising edge of INT1 generates an interrupt request. |

## USB Interrupt Sources

The USB interrupts are described below.

**Table 10.** USB Interrupt Sources

| Interrupt | Description |
|---|---|
| SOF Received | Whenever USB hardware decodes a valid Start of Frame. The frame number is stored in the two Frame Number Registers. |
| EOF2 | Activated whenever the hub's frame timer reaches its EOF2 time point. |
| Function EP0 Interrupt | See “Control Transfers at Control Endpoint EP0” on page 70 for details. |
| Function EP1 Interrupt | For an OUT endpoint it indicates that Function Endpoint 1 has received a valid OUT packet and that the data is in the FIFO. For an IN endpoint it means that the endpoint has received an IN token, sent out the data in the FIFO and received an ACK from the Host. The FIFO is now ready to be written by new data from the microcontroller. |
| Function EP2 Interrupt | For an OUT endpoint it indicates that Function Endpoint 2 has received a valid OUT packet and that the data is in the FIFO. For an IN endpoint it means that the endpoint has received an IN token, sent out the data in the FIFO and received an ACK from the Host. The FIFO is now ready to be written by new data from the microcontroller. |
| Hub EP0 Interrupt | See “Control Transfers at Control Endpoint EP0” on page 70 for details. |
| FRWUP | USB hardware has received a embedded function remote wakeup request. |
| GLB SUSP | USB hardware has received global suspend signaling and is preparing to put the hub in the suspend mode. The microcontroller's firmware should place the embedded function in the suspend state. |
| RSM | USB hardware received resume signaling and is propagating the resume signaling. The microcontroller's firmware should take the embedded function out of the suspended state. |

All interrupts have individual enable, status, and mask bits through the interrupt enable register and interrupt mask register. The Suspend and Resume interrupts are cleared by writing a 0 to the particular interrupt bit. All other interrupts are cleared when the microcontroller sets a bit in an interrupt acknowledge register.

## USB Endpoint Interrupt Sources

An assertion or activation of one or more bits in the endpoint's Control and Status Register triggers the endpoint interrupts. These triggers are different for control and non-control endpoints as described in the table below. Please refer to the Control and Status Register for more information.

**Table 11.** USB Endpoint Interrupt Sources

| Bit | Endpoint type |
|---|---|
| RX_OUT_PACKET | CONTROL, OUT |
| TX_COMPLETE | CONTROL, IN |
| STALL_SENT | CONTROL, IN |
| RX_SETUP | CONTROL |

***USB Interrupt Status Register – UISR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FF7 | SOF INT | EOF2 INT | – | – | HEP0 INT | FE2 INT | FE1 INT | FE0 INT | UISR |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – SOF INT: Start of Frame Interrupt**

This bit is asserted after the USB hardware receives a valid SOF packet.

- **Bit 6 – EOF2 INT: EOF2 Interrupt**

This bit is asserted 10 clocks before the expected start of a frame.

- **Bit 5, 4 – Res: Reserved Bits**

These bits are reserved and always read as zero.

- **Bit 3 – HEP0 INT: Hub Endpoint 0 Interrupt**
- **Bit 2 – FEP2 INT: Function Endpoint 2 Interrupt**
- **Bit 1 – FEP1 INT: Function Endpoint 1 Interrupt**
- **Bit 0 – FEP0 INT: Function Endpoint 0 Interrupt**

The hub and function interrupt bits will be set by the hardware whenever the following bits in the corresponding endpoint's Control and Status Register are modified by the USB hardware:

1. RX OUT Packet is set (control and OUT endpoints)
2. TX Packet Ready is cleared AND TX Complete is set (control and IN endpoints)
3. RX SETUP is set (control endpoints only)
4. TX Complete is set

### *USB Interrupt Acknowledge Register – UIAR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FF5 | SOF INTACK | EOF2 INTACK | – | – | HEP0 INTACK | FEP2 IMSK | FEP1 INTACK | FEP0 INTACK | UIAR |
| Read/Write | W | W | R | W | W | W | W | W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – SOF INTACK: Start of Frame Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the SOF INT bit.

- **Bit 6 – EOF2 INTACK: EOF2 Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the EOF2 INT bit.

- **Bit 5, 4 – Res: Reserved bits**

These bits are reserved and always read as zero.

- **Bit 3 – HEP0 INTACK: Hub Endpoint 0 Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the HEP0 INT bit.

- **Bit 2 – FEP2 INTACK: Function Endpoint 2 Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the FEP2 bit.

- **Bit 1 – FEP1 INTACK: Function Endpoint 1 Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the FEP1 bit.

- **Bit 0 – FEP0 INTACK: Function Endpoint 0 Interrupt Acknowledge**

The microcontroller firmware writes a 1 to this bit to clear the FEP0 INT bit.

### *USB Interrupt Enable Register – UIER*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FF3 | SOF IE | EOF2 IE | – | – | HEP0 IE | FEP2 IE | FEP1 IE | FEP0 IE | UIER |
| Read/Write | R/W | R/W | R | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – SOF IE: Enable Start of Frame Interrupt**

When the SOF IE bit is set (1), the Start of Frame Interrupt is enabled.

- **Bit 6 – EOF2 IE: Enable EOF2 Interrupt**

When the EOF2 IE bit is set (1), the EOF2 Interrupt is enabled.

- **Bit 5, 4 – Res: Reserved bit**

These bits are reserved and always read as zero.

- **Bit 3 – HEP0 IE: Enable Endpoint 0 Interrupt**

When the HEP0 IE bit is set (1), the Hub Endpoint 0 Interrupt is enabled.

- **Bit 2 – FEP2 IE: Enable Endpoint 2 Interrupt**

When the FE2 IE bit is set (1), the Function Endpoint 2 Interrupt is enabled.

- **Bit 1 – FEP1 IE: Enable Endpoint 1 Interrupt**

When the FE1 IE bit is set (1), the Function Endpoint 1 Interrupt is enabled.

- **Bit 0 – FEP0 IE: Enable Endpoint 0 Interrupt**

When the FE0 IE bit is set (1), the Function Endpoint 0 Interrupt is enabled.

### *Suspend/Resume Register – SPRSR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FFA | – | – | – | – | – | FRWUP | RSM | GLB SUSP | SPRSR |
| Read/Write | R | R | R | R | R/W | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..3 – Res: Reserved Bits**

These bits are reserved and are always read as zeros.

- **Bit 2 – FRWUP: Function Remote Wakeup**

The USB hardware sets this bit to signal that External Interrupt 1 is detected indicating remote wakeup. An interrupt is generated if the FRWUP IE bit of the SPRSIE register is set.

- **Bit 1 – RSM: Resume**

The USB hardware sets this bit when a USB resume signaling is detected at any of its port except Port 1. An interrupt is generated if the RSM IE bit of the SPRSIE register is set.

- **Bit 0 – GLB SUSP: Global Suspend**

The USB hardware sets this bit when a USB global suspend signaling is detected. An interrupt is generated if the GLBSUSP IE bit of the SPRSIE register is set.

***Suspend/Resume Interrupt Enable Register – SPRSIE***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FF9 | – | – | – | – | – | FRWUP | RSM | GLB SUSP | SPRSIE |
| Read/Write | R | R | R | R | R/W | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..3 – Res: Reserved Bits**

These bits are reserved and are always read as zeros.

- **Bit 3 – BUS INT EN: USB Reset Interrupt Enable**

When the BUS INT EN bit is set, the USB and microcontroller resets are separated. A USB bus reset (SE0 for longer than 3 ms) will reset the USB hardware only and not the microcontroller. However, an interrupt to the microcontroller will be generated and bit 3 of SPRSR is set.

- **Bit 2 – FRWUP IE: Function Remote Wakeup Interrupt Enable**

Setting the FRWUP IE bit will initiate an interrupt whenever the FRWUP bit of SPRSR is set.

- **Bit 1 – RSM IE: Resume Interrupt Enable**

Setting the RSM IE bit will initiate an interrupt whenever the RSM bit of SPRSR is set.

- **Bit 0 – GLB SUSP IE: Global Suspend Interrupt Enable**

Setting the GLB SUSP IE bit will initiate an interrupt whenever the GLB SUSP bit of SPRSR is set.

# AVR Register Set

## Status Register and Stack Pointer

***Status Register – SREG***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $3F ($5F) | I | T | H | S | V | N | Z | C | SREG |
| 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 – I: Global Interrupt Enable**

The global interrupt enable bit must be set (one) for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the global interrupt enable bit is cleared (zero), none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by the hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts.

- **Bit 6 – T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T bit as source and destination for the operated bit. A bit from a register in the register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the register file by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The half carry flag H indicates a half carry in some arithmetic operations. See the Instruction Set Description for detailed information.

- **Bit 4 – S: Sign Bit, S = N⊕V**

The S-bit is always an exclusive or between the negative flag N and the two's complement overflow flag V. See the Instruction Set Description for detailed information.

- **Bit 3 – V: Two's Complement Overflow Flag**

The two's complement overflow flag V supports two's complement arithmetics. See the Instruction Set Description for detailed information.

- **Bit 2 – N: Negative Flag**

The negative flag N indicates a negative result after the different arithmetic and logic operations. See the Instruction Set Description for detailed information.

- **Bit 1 – Z: Zero Flag**

The zero flag Z indicates a zero result after the different arithmetic and logic operations. See the Instruction Set Description for detailed information.

- **Bit 0 – C: Carry Flag**

The carry flag C indicates a carry in an arithmetic or logic operation. See the Instruction Set Description for detailed information.

Note that the status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt routine. This must be handled by software.

***Stack Pointer Register – SP***

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | |
|---|---|---|---|---|---|---|---|---|---|
| $3E ($5E) | **I** | **T** | **H** | **S** | **V** | **N** | **Z** | **C** | **SPH** |
| $3D ($5D) | **SP7** | **SP6** | **SP5** | **SP4** | **SP3** | **SP2** | **SP1** | **SP0** | **SPL** |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The Stack Pointer points to the data SRAM stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The stack pointer must be set to point above $60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when an address is pushed onto the Stack with subroutine calls and interrupts. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction and it is incremented by two when an address is popped from the Stack with return from subroutine RET or return from interrupt RETI.

## Sleep Modes

To enter the sleep modes, the SE bit in MCUCR must be set (one) and a SLEEP instruction must be executed. If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU awakes, executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the register file, SRAM and I/O memory are unaltered. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector.

### Power Down Mode

When the SM bit is set (one), the SLEEP instruction forces the MCU into the Power Down Mode. In this mode, the external oscillator is stopped, while the external interrupts and the Watchdog (if enabled) continue operating. Only an external reset or an external level interrupt on INT0 or INT1 can wake up the MCU.

Note that when a level triggered interrupt is used for wake-up from power down, the low level must be held for a time longer than the reset delay time-out period $t_{TOUT}$. Otherwise, the MCU will fail to wake up.

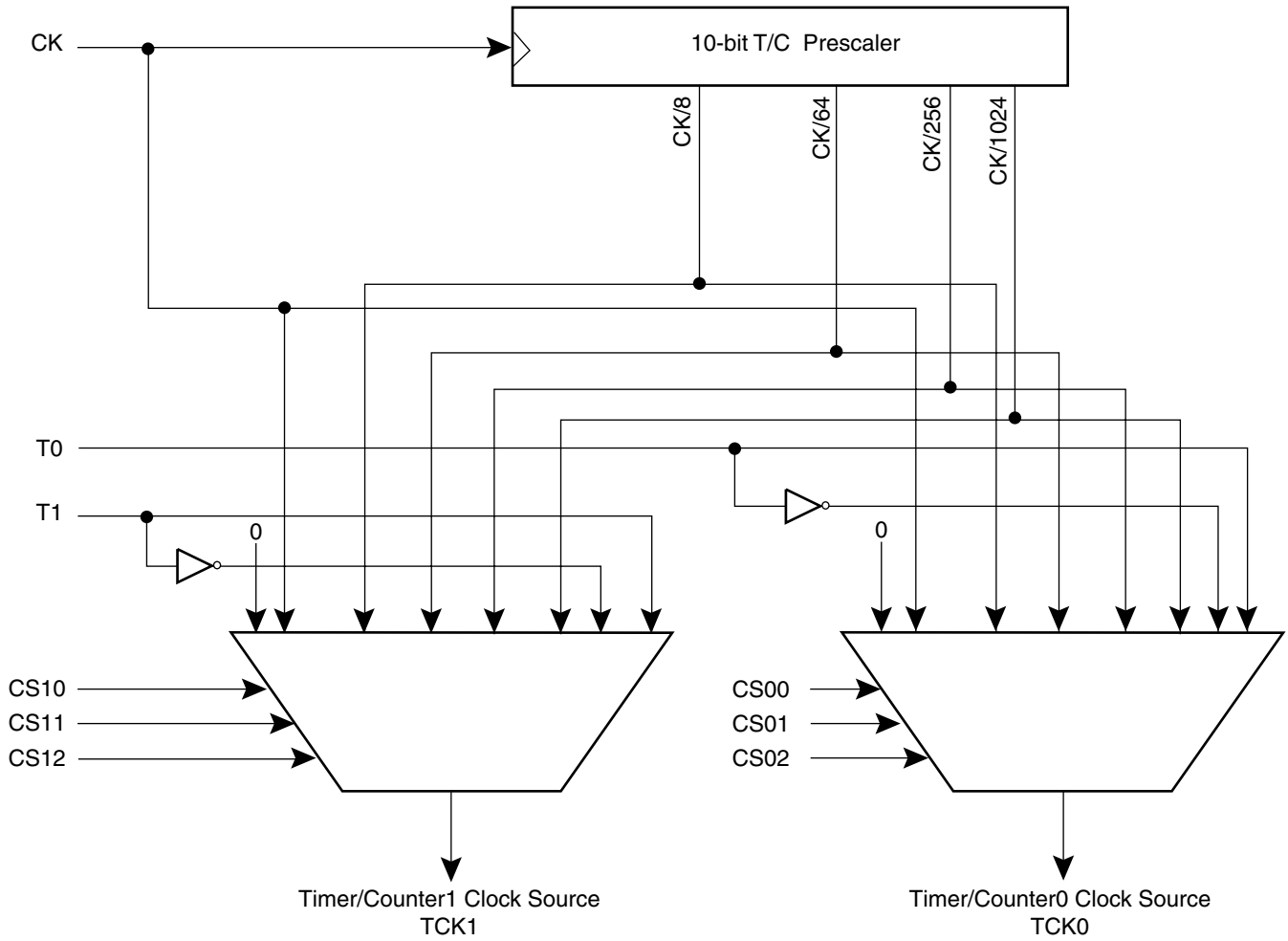
## Timer/Counters

The AT43USB320A provides two general-purpose Timer/Counters - one 8-bit T/C and one 16-bit T/C. The Timer/Counters have individual prescaling selection from the same 10-bit prescaling timer. Both Timer/Counters can either be used as a timer with an internal clock timebase or as a counter with an external pin connection which triggers the counting.

## Timer/Counter Prescaler

The four different prescaled selections are: CK/8, CK/64, CK/256 and CK/1024 where CK is the oscillator clock. For the two Timer/Counters, added selections as CK, external source and stop, can be selected as clock sources.

**Figure 9.** Timer/Counter Prescaler



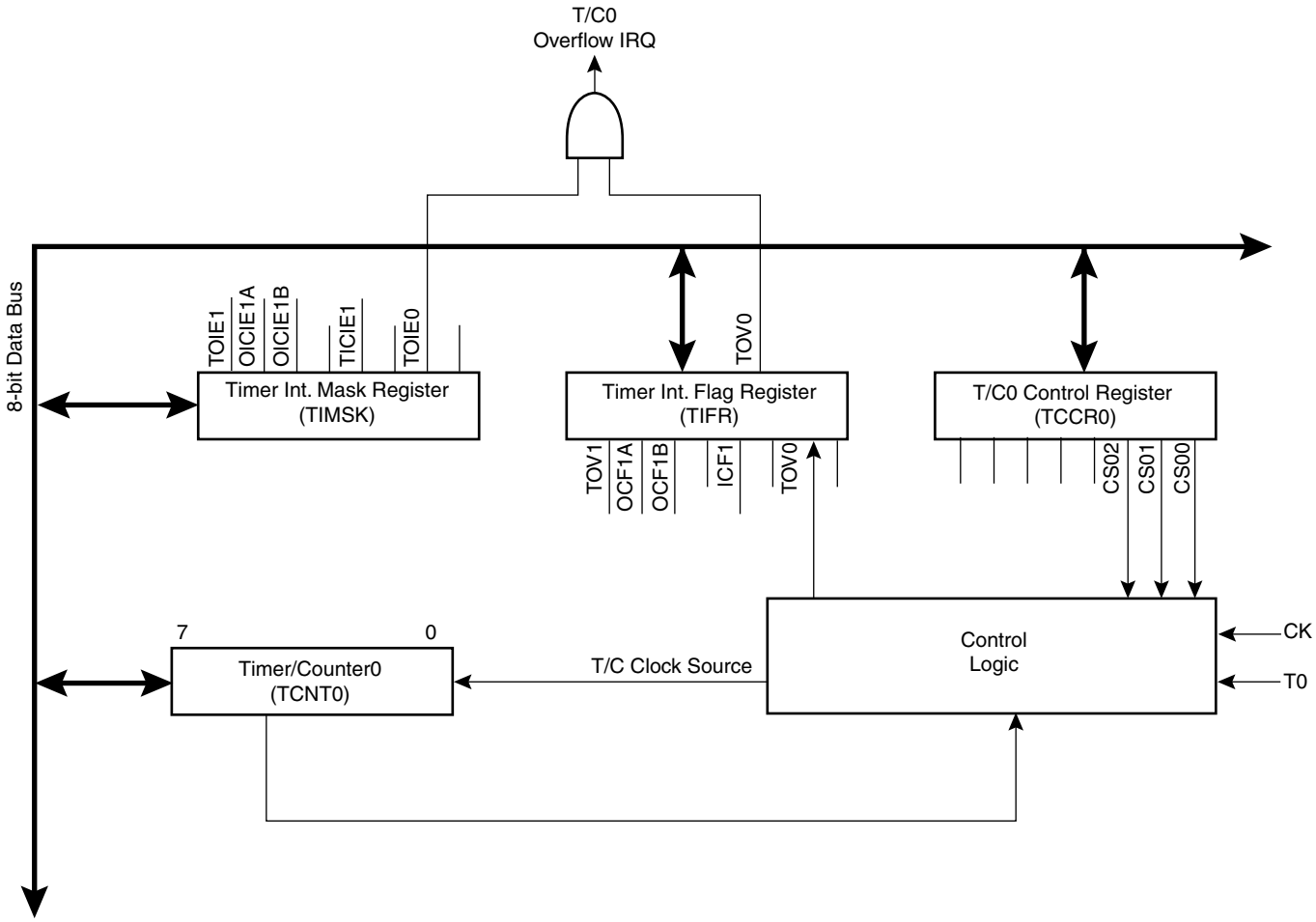
## 8-bit Timer/Counter0

The 8-bit Timer/Counter0 can select clock source from CK, prescaled CK or an external pin. In addition it can be stopped as described in the specification for the Timer/Counter0 Control Register (TCCR0). The overflow status flag is found in the Timer/Counter Interrupt Flag Register (TIFR). Control signals are found in the Timer/Counter0 Control Register (TCCR0). The interrupt enable/disable settings for Timer/Counter0 are found in the Timer/Counter Interrupt Mask Register - TIMSK.

When Timer/Counter0 is externally clocked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock.

The 8-bit Timer/Counter0 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter0 useful for lower speed functions or exact timing functions with infrequent actions.

**Figure 10.** Timer/Counter0 Block Diagram



### *Timer/Counter0 Control Register – TCCR0*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $33 ($53) | – | – | – | – | – | CS02 | CS01 | CS00 | **TCCR0** |
| Read/Write | R | R | R | R | R | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and always read as zero.

- **Bits 2, 1, 0 – CS02, CS01, CS00: Clock Select0, bit 2, 1 and 0**

The Clock Select0 bits 2, 1 and 0 define the prescaling source of Timer/Counter0.

**Table 12.** Clock 0 Prescale Select

| CS02 | CS01 | CS00 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Stop, the Timer/Counter0 is stopped |
| 0 | 0 | 1 | CK |
| 0 | 1 | 0 | CK/8 |
| 0 | 1 | 1 | CK/64 |
| 1 | 0 | 0 | CK/256 |
| 1 | 0 | 1 | CK/1024 |
| 1 | 1 | 0 | External Pin T0, falling edge |
| 1 | 1 | 1 | External Pin T0, rising edge |

The Stop condition provides a Timer Enable/Disable function. The CK down divided modes are scaled directly from the CK oscillator clock. If the external pin modes are used for Timer/Counter0, transitions on PB0/(T0) will clock the counter even if the pin is configured as an output. This feature can give the user SW control of the counting.

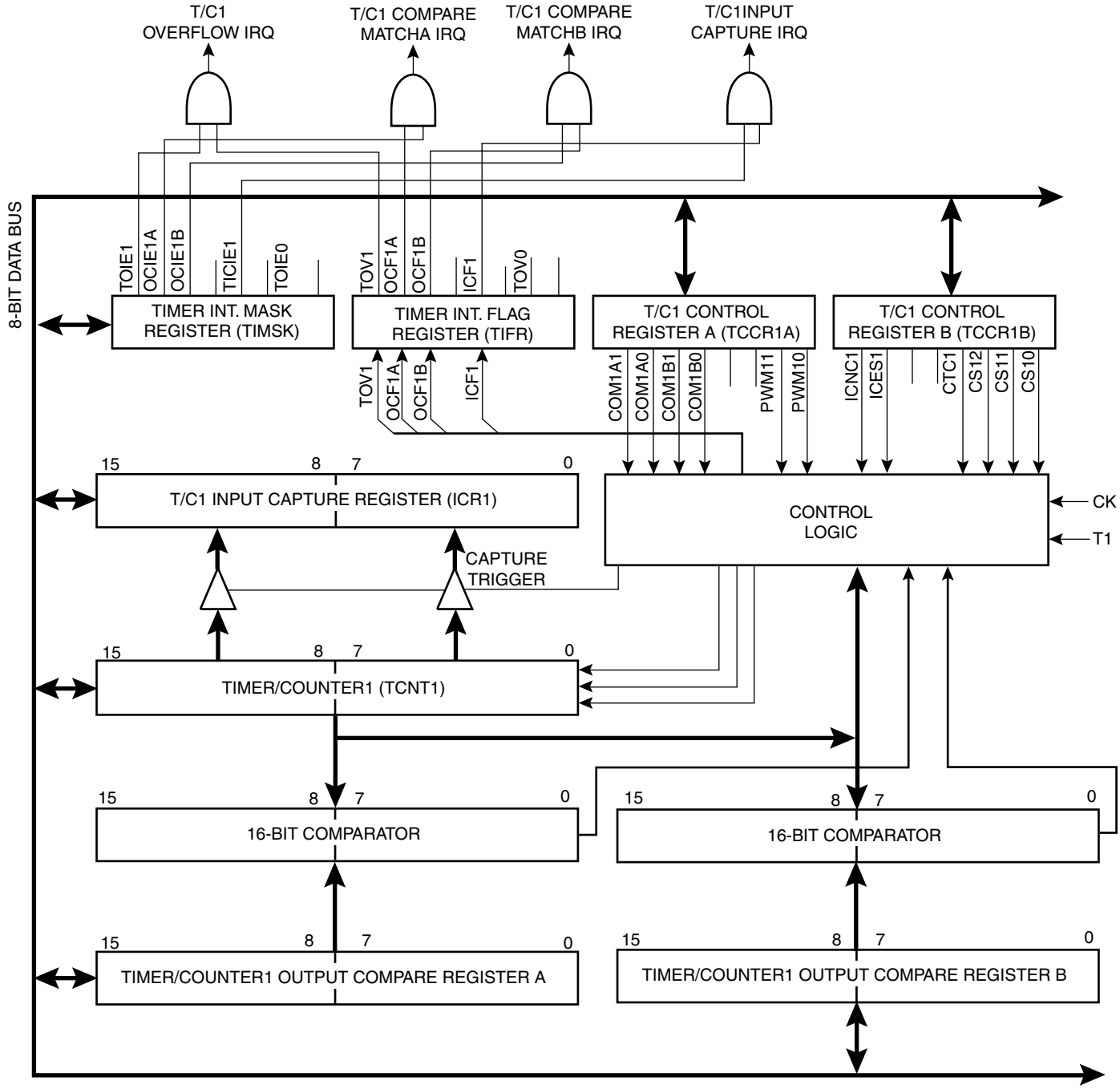
### *Timer/Counter0 – TCNT0*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $32 ($52) | **MSB** | – | – | – | – | – | – | **LSB** | **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/Counter0 is realized as an up-counter with read and write access. If the Timer/Counter0 is written and a clock source is present, the Timer/Counter0 continues counting in the clock cycle following the write operation.

## 16-bit Timer/Counter1

**Figure 11.** Timer/Counter1 Block Diagram



## 16-bit Timer/Counter1 Operation

The 16-bit Timer/Counter1 can select clock source from CK, prescaled CK or an external pin. In addition, it can be stopped as described in the specification for the Timer/Counter1 Control Registers (TCCR1A and TCCR1B). The different status flags (overflow, compare match and capture event) are found in the Timer/Counter Interrupt Flag Register (TIFR). Control signals are found in the Timer/Counter1 Control Registers (TCCR1A and TCCR1B). The interrupt enable/disable settings for Timer/Counter1 are found in the Timer/Counter Interrupt Mask Register (TIMSK).

When Timer/Counter1 is externally clocked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock.

The 16-bit Timer/Counter1 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities makes the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

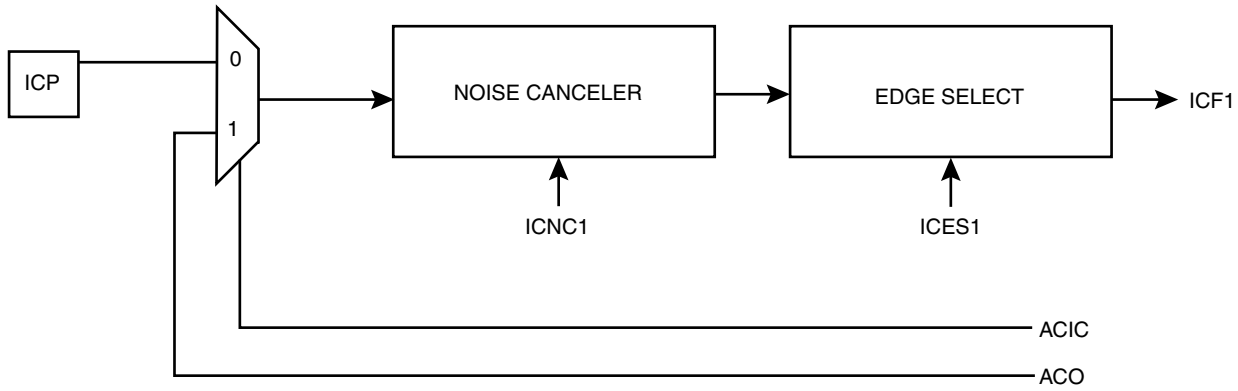
The Timer/Counter1 supports two Output Compare functions using the Output Compare Register 1 A and B (OCR1A and OCR1B) as the data sources to be compared to the Timer/Counter1 contents. The Output Compare functions include optional clearing of the counter on compareA match, and actions on the Output Compare pins on both compare matches.

Timer/Counter1 can also be used as a 8-, 9- or 10-bit Pulse With Modulator. In this mode the counter and the OCR1A/OCR1B registers serve as a dual glitch-free stand-alone PWM with centered pulses.

The Input Capture function of Timer/Counter1 provides a capture of the Timer/Counter1 contents to the Input Capture Register - ICR1, triggered by an external event on the Input Capture Pin (ICP/PF3). The actual capture event settings are defined by the Timer/Counter1 Control Register (TCCR1B). In addition, the Analog Comparator can be set to trigger the Input Capture. Refer to .

If the noise canceler function is enabled, the actual trigger condition for the capture event is monitored over 4 samples, and all 4 must be equal to activate the capture flag.

**Figure 12.** ICP Pin Schematic Diagram




ACIC: COMPARATOR IC ENABLE
ACC0: COMPARATOR OUTPUT

### *Timer/Counter1 Control Register A – TCCR1A*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $2F ($4F) | COM1A1 | COM1A0 | COM1B1 | COM1B0 | – | – | PWM11 | PWM10 | 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 | |

- **Bits 7, 6 – COM1A1, COM1A0: Compare Output Mode1A, Bits 1 and 0**

The COM1A1 and COM1A0 control bits determine any output pin action following a compare match in Timer/Counter1. Any output pin actions affect pin OC1A (Output CompareA) pin 1. This is an alternative function to an I/O port and the corresponding direction control bit must be set (one) to control the output pin. The control configuration is shown in Table 13.

- **Bits 5, 4 – COM1B1, COM1B0: Compare Output Mode1B, Bits 1 and 0**

The COM1B1 and COM1B0 control bits determine any output pin action following a compare match in Timer/Counter1. Any output pin actions affect pin OC1B (Output CompareB). The following control configuration is given:

**Table 13.** Compare 1 Mode Select[2]

| COM1X1 | COM1X0 | Description |
|---|---|---|
| 0 | 0 | Timer/Counter1 disconnected from output pin OC1X.[1] |
| 0 | 1 | Toggle the OC1X output line.[1] |
| 1 | 0 | Clear the OC1X output line (to zero).[1] |
| 1 | 1 | Set the OC1X output line (to one).[1] |

Notes: 1. X = A or B
2. In PWM mode, these bits have a different function. Refer to Table 17 for a detailed description.

- **Bits 3..2 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and always read zero.

- **Bits 1..0 – PWM11, PWM10: Pulse Width Modulator Select Bits 1 and 0**

These bits select PWM operation of Timer/Counter1 as specified in Table 14.

**Table 14.** PWM Mode Select

| PWM11 | PWM10 | Description |
|---|---|---|
| 0 | 0 | PWM operation of Timer/Counter1 is disabled. |
| 0 | 1 | Timer/Counter1 is an 8-bit PWM. |
| 1 | 0 | Timer/Counter1 is a 9-bit PWM. |
| 1 | 1 | Timer/Counter1 is a 10-bit PWM. |

### *Timer/Counter1 Control Register B – TCCR1B*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $2E ($4E) | ICNC1 | ICES1 | – | – | CTC1 | CS12 | CS11 | CS10 | TCCR1B |
| 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 – ICNC1: Input Capture1 Noise Canceler (4 CKs)**

When the ICNC1 bit is cleared (zero), the input capture trigger noise canceler function is disabled. The input capture is triggered at the first rising/falling edge sampled on the ICP (input capture pin) as specified. When the ICNC1 bit is set (one), four successive samples are measured on the ICP and all samples must be high/low according to the input capture trigger specification in the ICES1 bit. The actual sampling frequency is the 12 MHz system clock frequency.

- **Bit 6 – ICES1: Input Capture1 Edge Select**

While the ICES1 bit is cleared (zero), the Timer/Counter1 contents are transferred to the Input Capture Register (ICR1) on the falling edge of the ICP. While the ICES1 bit is set (one), the Timer/Counter1 contents are transferred to the ICR1 on the rising edge of the ICP.

- **Bits 5, 4 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and always read zero.

- **Bit 3 – CTC1: Clear Timer/Counter1 on Compare Match**

When the CTC1 control bit is set (one), the Timer/Counter1 is reset to $0000 in the clock cycle after a compareA match. If the CTC1 control bit is cleared, Timer/Counter1 continues counting and is unaffected by a compare match. Since the compare match is detected in the CPU clock cycle following the match, this function will behave differently when a prescaling higher than 1 is used for the timer. When a prescaling of 1 is used, and the compareA register is set to C, the timer will count as follows if CTC1 is set:

... | C-2 | C-1 | C | 0 | 1 | ...

When the prescaler is set to divide by 8, the timer will count like this:

... | C-2, C-2, C-2, C-2, C-2, C-2, C-2, C-2 | C-1, C-1, C-1, C-1, C-1, C-1, C-1, C-1 | C, 0, 0, 0, 0, 0, 0, 0 | ...

In PWM mode, this bit has no effect.

- **Bits 2, 1, 0 – CS12, CS11, CS10: Clock Select1, Bit 2, 1 and 0**

The Clock Select1 bits 2, 1 and 0 define the prescaling source of Timer/Counter1.

**Table 15.** Clock 1 Prescale Select

| CS12 | CS11 | CS10 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Stop, the Timer/Counter1 is stopped. |
| 0 | 0 | 1 | CK |
| 0 | 1 | 0 | CK/8 |
| 0 | 1 | 1 | CK/64 |
| 1 | 0 | 0 | CK/256 |

**Table 15.** Clock 1 Prescale Select (Continued)

| CS12 | CS11 | CS10 | Description |
|---|---|---|---|
| 1 | 0 | 1 | CK/1024 |
| 1 | 1 | 0 | External Pin T1, falling edge |
| 1 | 1 | 1 | External Pin T1, rising edge |

The Stop condition provides a Timer Enable/Disable function. The CK down divided modes are scaled directly from the 12 MHz system clock. If the external pin modes are used for Timer/Counter1, transitions on PB1/(T1) will clock the counter even if the pin is configured as an output. This feature can give the user SW control of the counting.

### Timer/Counter1 – TCNT1H and TCNT1L

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | |
|---|---|---|---|---|---|---|---|---|---|
| $2D ($4D) | **MSB** | – | – | – | – | – | – | – | **TCNT1H** |
| $2C ($4C) | – | – | – | – | – | – | – | **LSB** | **TCNT1L** |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

This 16-bit register contains the prescaled value of the 16-bit Timer/Counter1. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary register (TEMP). This temporary register is also used when accessing OCR1A, OCR1B and ICR1. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program and from interrupt routines if interrupts are allowed from within interrupt routines.

- **TCNT1 Timer/Counter1 Write:**

When the CPU writes to the high byte TCNT1H, the written data is placed in the TEMP register. Next, when the CPU writes the low byte TCNT1L, this byte of data is combined with the byte data in the TEMP register, and all 16 bits are written to the TCNT1 Timer/Counter1 register simultaneously. Consequently, the high byte TCNT1H must be accessed first for a full 16-bit register write operation.

- **TCNT1 Timer/Counter1 Read:**

When the CPU reads the low byte TCNT1L, the data of the low byte TCNT1L is sent to the CPU and the data of the high byte TCNT1H is placed in the TEMP register. When the CPU reads the data in the high byte TCNT1H, the CPU receives the data in the TEMP register. Consequently, the low byte TCNT1L must be accessed first for a full 16-bit register read operation.

The Timer/Counter1 is realized as an up or up/down (in PWM mode) counter with read and write access. If Timer/Counter1 is written to and a clock source is selected, the Timer/Counter1 continues counting in the timer clock cycle after it is preset with the written value.

### *Timer/Counter1 Output Compare Register – OCR1AH and OCR1AL*

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | |
|---|---|---|---|---|---|---|---|---|---|
| $2B ($4B) | **MSB** | – | – | – | – | – | – | – | **OCR1AH** |
| $2A ($4A) | – | – | – | – | – | – | – | **LSB** | **OCR1AL** |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

### *Timer/Counter1 Output Compare Register – OCR1BH and OCR1BL*

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | |
|---|---|---|---|---|---|---|---|---|---|
| $29 ($49) | **MSB** | – | – | – | – | – | – | – | **OCR1BH** |
| $28 ($48) | – | – | – | – | – | – | – | **LSB** | **OCR1BL** |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The output compare registers are 16-bit read/write registers.

The Timer/Counter1 Output Compare Registers contain the data to be continuously compared with Timer/Counter1. Actions on compare matches are specified in the Timer/Counter1 Control and Status register.A compare match does only occur if Timer/Counter1 counts to the OCR value. A software write that sets TCNT1 and OCR1A or OCR1B to the same value does not generate a compare match.

A compare match will set the compare interrupt flag in the CPU clock cycle following the compare event.

Since the Output Compare Registers OCR1A and OCR1B are 16-bit registers, a temporary register TEMP is used when OCR1A/B are written to ensure that both bytes are updated simultaneously. When the CPU writes the high byte, OCR1AH or OCR1BH, the data is temporarily stored in the TEMP register. When the CPU writes the low byte, OCR1AL or OCR1BL, the TEMP register is simultaneously written to OCR1AH or OCR1BH. Consequently, the high byte OCR1AH or OCR1BH must be written first for a full 16-bit register write operation.

The TEMP register is also used when accessing TCNT1, and ICR1. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program and from interrupt routines if interrupts are allowed from within interrupt routines.

*Timer/Counter1 Input Capture Register – ICR1H and ICR1L*

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | |
|---|---|---|---|---|---|---|---|---|---|
| $25 ($45) | **MSB** | – | – | – | – | – | – | – | **ICR1H** |
| $24 ($44) | – | – | – | – | – | – | – | **LSB** | **ICR1L** |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Read/Write | R | R | R | R | R | R | R | R | |
| | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The input capture register is a 16-bit read-only register.

When the rising or falling edge (according to the input capture edge setting - ICES1) of the signal at the input capture pin (ICP) is detected, the current value of the Timer/Counter1 is transferred to the Input Capture Register (ICR1). At the same time, the Input Capture Flag (ICF1) is set (one).

Since the ICR1 is a 16-bit register, a temporary register TEMP is used when ICR1 is read to ensure that both bytes are read simultaneously. When the CPU reads the low byte ICR1L, the data is sent to the CPU and the data of the high byte ICR1H is placed in the TEMP register. When the CPU reads the data in the high byte ICR1H, the CPU receives the data in the TEMP register. Consequently, the low byte ICR1L must be accessed first for a full 16-bit register read operation.

The TEMP register is also used when accessing TCNT1, OCR1A and OCR1B. If the main program and also interrupt routines perform access to registers using TEMP, interrupts must be disabled during access from the main program and from interrupt routines, if interrupts are allowed from within interrupt routines.

**Timer/Counter1 In PWM Mode**

When the PWM mode is selected, Timer/Counter1, the Output Compare Register1A (OCR1A) and the Output Compare Register1B (OCR1B) form a dual 8-, 9- or 10-bit, free-running, glitch-free and phase correct PWM with outputs on the PD5 (OC1A) and OC1B pins. Timer/Counter1 acts as an up/down counter, counting up from $0000 to TOP (see Table 16), where it turns and counts down again to zero before the cycle is repeated. When the counter value matches the contents of the 10 least significant bits of OCR1A or OCR1B, the PD5(OC1A)/OC1B pins are set or cleared according to the settings of the COM1A1/COM1A0 or COM1B1/COM1B0 bits in the Timer/Counter1 Control Register TCCR1A. Refer to Table 17 for details.

**Table 16.** Timer TOP Values and PWM Frequency

| PWM Resolution | Timer TOP value | Frequency |
|---|---|---|
| 8-bit | $00FF (255) | $f_{TCK1}/510$ |
| 9-bit | $01FF (511) | $f_{TCK1}/1022$ |
| 10-bit | $03FF(1023) | $f_{TCK1}/2046$ |

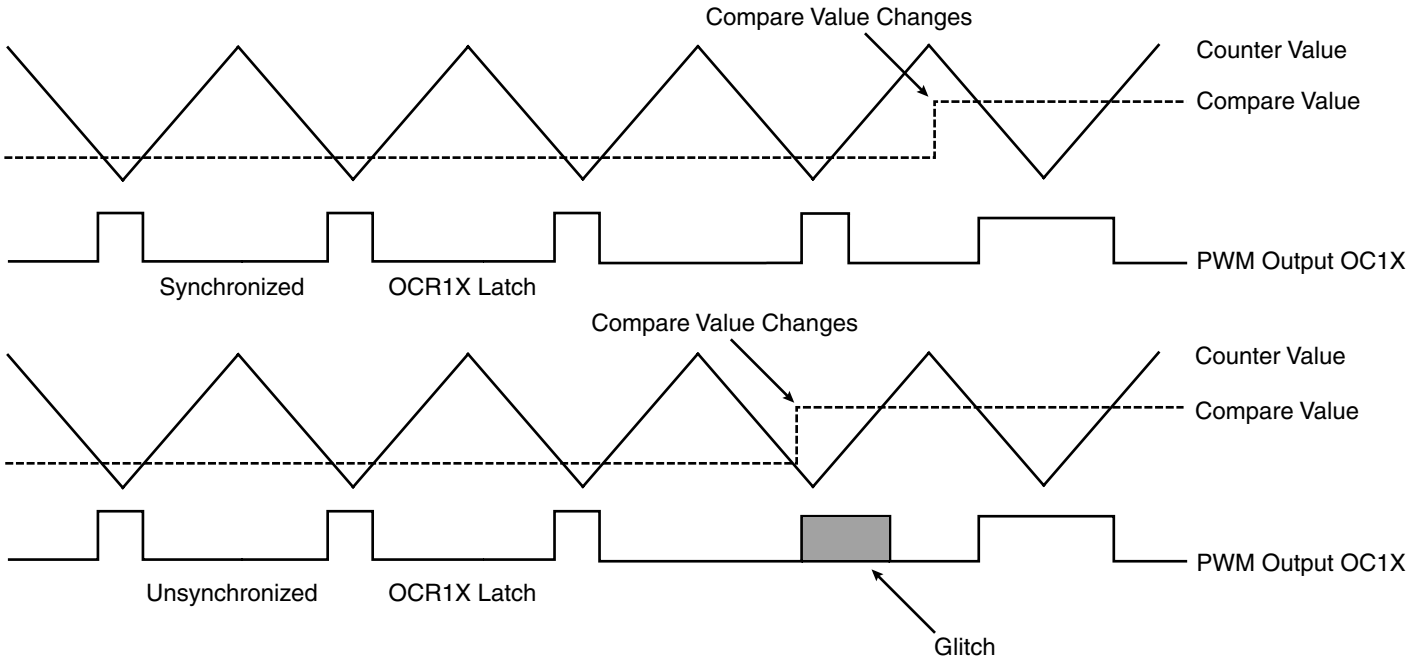
**Table 17.** Compare1 Mode Select in PWM Mode

| COM1X1 | COM1X0 | Effect on OCX1 |
|---|---|---|
| 0 | 0 | Not connected |
| 0 | 1 | Not connected |
| 1 | 0 | Cleared on compare match, up-counting. Set on compare match, down-counting (non-inverted PWM). |
| 1 | 1 | Cleared on compare match, down-counting. Set on compare match, up-counting (inverted PWM). |

Note: X = A or B

Note that in the PWM mode, the 10 least significant OCR1A/OCR1B bits, when written, are transferred to a temporary location. They are latched when Timer/Counter1 reaches the value TOP. This prevents the occurrence of odd-length PWM pulses (glitches) in the event of an unsynchronized OCR1A/OCR1B write. See Figure 13 for an example.

**Figure 13.** Effects on Unsynchronized OCR1 Latching




Note: X = A or B

During the time between the write and the latch operation, a read from OCR1A or OCR1B will read the contents of the temporary location. This means that the most recently written value always will read out of OCR1A/B

When the OCR1 contains $0000 or TOP, the output OC1A/OC1B is updated to low or high on the next compare match, according to the settings of COM1A1/COM1A0 or COM1B1/COM1B0. This is shown in Table 18.

Note: If the compare register contains the TOP value and the prescaler is not in use (CS12..CS10 = 001), the PWM output will not produce any pulse at all, because up-counting and down-counting values are reached simultaneously. When the prescaler is in use (CS12..CS10 = 001 or 000), the PWM output goes active when the counter reaches the TOP

value, but the down-counting compare match is not interpreted to be reached before the next time the counter reaches the TOP value, making a one-period PWM pulse.

**Table 18.** PWM Outputs OCR1X = $0000 or Top

| COM1X1 | COM1X0 | OCR1X | Output OC1X |
|---|---|---|---|
| 1 | 0 | $0000 | L |
| 1 | 0 | TOP | H |
| 1 | 1 | $0000 | H |
| 1 | 1 | TOP | L |

Note: X = A or B

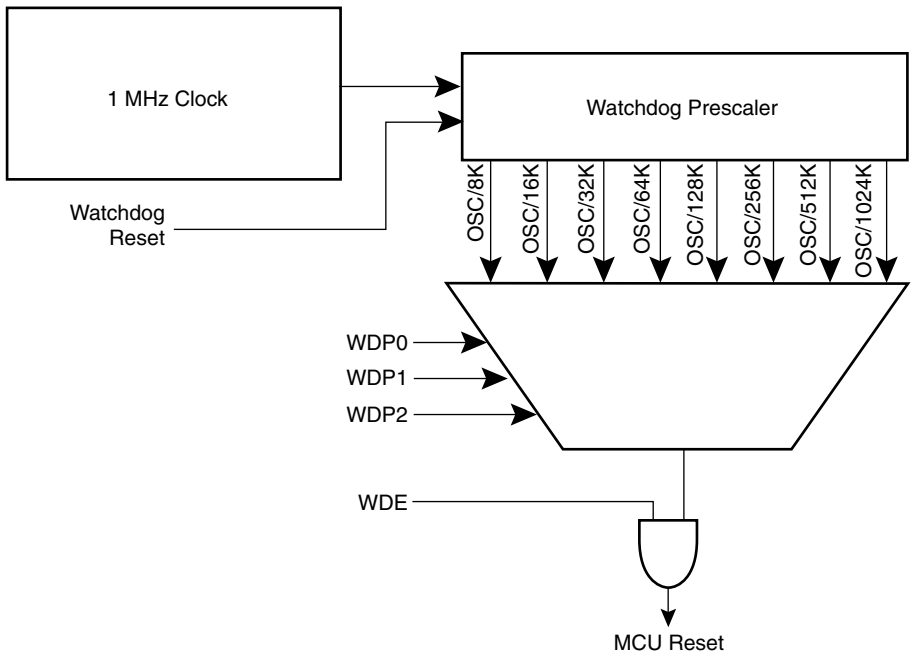
In PWM mode, the Timer Overflow Flag1, TOV1, is set when the counter advances from $0000. Timer Overflow Interrupt1 operates exactly as in normal Timer/Counter mode, i.e. it is executed when TOV1 is set provided that Timer Overflow Interrupt1 and global interrupts are enabled. This also applies to the Timer Output Compare1 flags and interrupts.

## Watchdog Timer

The Watchdog Timer is clocked from a 1 MHz clock derived from the 6 MHz on chip oscillator. By controlling the Watchdog Timer prescaler, the Watchdog reset interval can be adjusted, see Table 19 for a detailed description. The WDR (Watchdog Reset) instruction resets the Watchdog Timer. Eight different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog reset, the AT43USB320A resets and executes from the reset vector.

To prevent unintentional disabling of the watchdog, a special turn-off sequence must be followed when the watchdog is disabled. Refer to the description of the Watchdog Timer Control Register for details.

**Figure 14.** Watchdog Timer



### *Watchdog Timer Control Register – WDTCR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $21 ($41) | – | – | – | WDTOE | WDE | WDP2 | WDP1 | WDP0 | WDTCR |
| Read/Write | R | R | R | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bits 7..5 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and will always read as zero.

- **Bit 4 – WDTOE: Watch Dog Turn-Off Enable**

This bit must be set (one) when the WDE bit is cleared. Otherwise, the watchdog will not be disabled. Once set, the hardware will clear this bit to zero after four clock cycles. Refer to the description of the WDE bit for a watchdog disable procedure.

- **Bit 3 – WDE: Watch Dog Enable**

When the WDE is set (one) the Watchdog Timer is enabled, and if the WDE is cleared (zero) the Watchdog Timer function is disabled. WDE can only be cleared if the WDTOE bit is set (one). To disable an enabled watchdog timer, the following procedure must be followed:

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

- **Bits 2..0 – WDP2, WDP1, WDP0: Watch Dog Timer Prescaler 2, 1 and 0**

The WDP2, WDP1 and WDP0 bits determine the Watchdog Timer prescaling when the Watchdog Timer is enabled. The different prescaling values and their corresponding Time-out Periods are shown in Table 19.

**Table 19.** Watchdog Timer Prescale Select

| WDP2 | WDP1 | WDP0 | Number of WDT Oscillator cycles | Time-out |
|---|---|---|---|---|
| 0 | 0 | 0 | 8K cycles | 8.2 ms |
| 0 | 0 | 1 | 16K cycles | 16.4 ms |
| 0 | 1 | 0 | 32K cycles | 33.8 ms |
| 0 | 1 | 1 | 64K cycles | 65.6 ms |
| 1 | 0 | 0 | 128K cycles | 0.131 s |
| 1 | 0 | 1 | 256K cycles | 0.262 s |
| 1 | 1 | 0 | 512K cycles | 0.524 s |
| 1 | 1 | 1 | 1,024K cycles | 1.048 s |

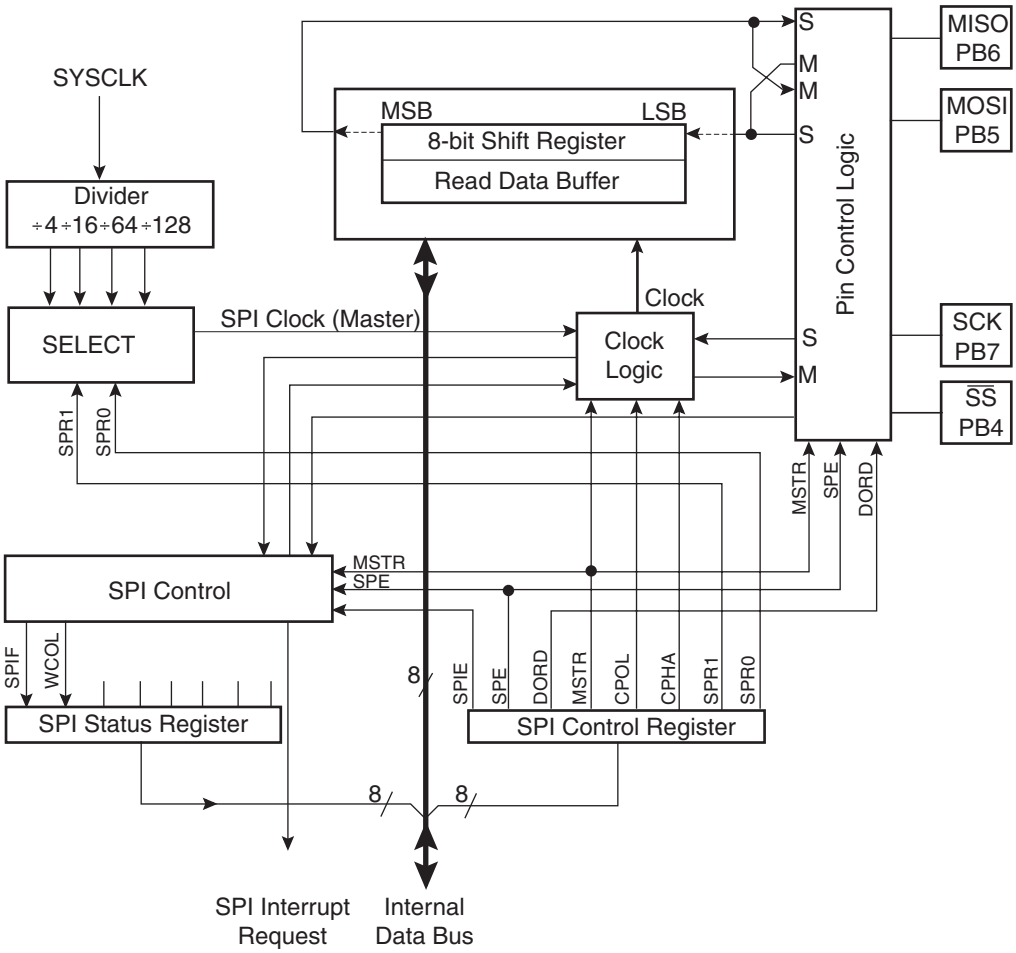
Note: The WDR (Watchdog Reset) instruction should always be executed before the Watchdog Timer is enabled. This ensures that the reset period will be in accordance with the Watchdog Timer prescale settings. If the Watchdog Timer is enabled without reset, the watchdog timer may not start to count from zero. To avoid unintentional MCU reset, the Watchdog Timer should be disabled or reset before changing the Watchdog Timer Prescale Select.

## Serial Peripheral Interface (SPI)

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the AT43USB320A and peripheral devices or between several AVR devices. The AT43USB320A SPI features include the following:

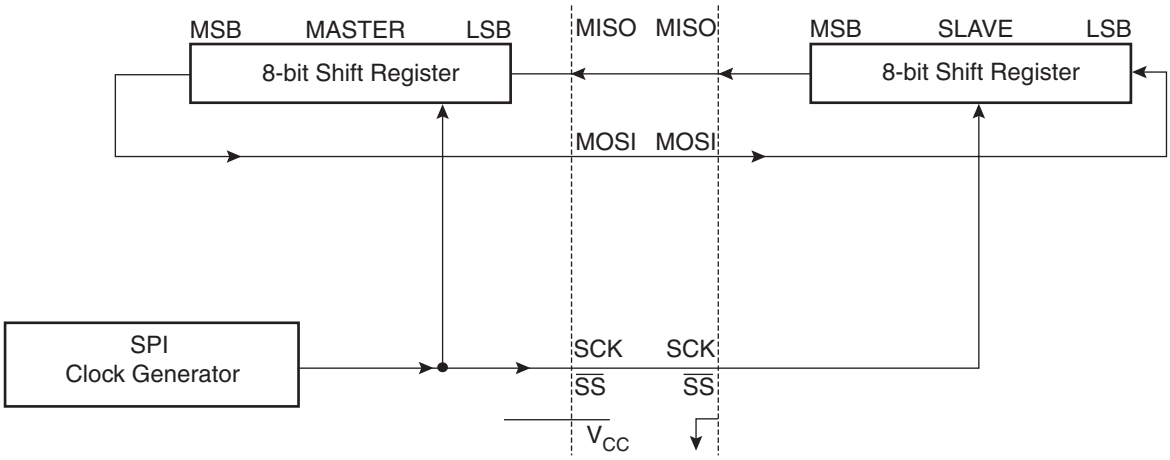
- Full-duplex, 3-wire Synchronous Data Transfer
- Master or Slave Operation
- LSB First or MSB First Data Transfer
- Four Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wakeup from Idle Mode (Slave Mode Only)

**Figure 15.** SPI Block Diagram



The interconnection between master and slave CPUs with SPI is shown in Figure 16. The PB7(SCK) pin is the clock output in the master mode and is the clock input in the slave mode. Writing to the SPI data register of the master CPU starts the SPI clock generator, and the data written shifts out of the PB5(MOSI) pin and into the PB5(MOSI) pin of the slave CPU. After shifting one byte, the SPI clock generator stops, setting the end of transmission flag (SPIF). If the SPI interrupt enable bit (SPIE) in the SPCR register is set, an interrupt is requested. The Slave Select input, PB4(SS), is set low to select an individual slave SPI device. The two shift registers in the Master and the Slave can be considered as one distributed 16-bit circular shift register. This is shown in Figure 16. When data is shifted from the master to the slave, data is also shifted in the opposite direction, simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

**Figure 16.** SPI Master/Slave Interconnection




The system is single buffered in the transmit direction and double buffered in the receive direction. This means that bytes to be transmitted cannot be written to the SPI Data Register before the entire shift cycle is completed. When receiving data, however, a received byte must be read from the SPI Data Register before the next byte has been completely shifted in. Otherwise, the first byte is lost.

When the SPI is enabled, the data direction of the MOSI, MISO, SCK and SS pins is overridden according to the following table:

**Table 20.** SPI Pin Overrides

| Pin | Direction, Master SPI | Direction, Slave SPI |
|---|---|---|
| MOSI | User Defined | Input |
| MISO | Input | User Defined |
| SCK | User Defined | Input |
| SSN | User Defined | Input |

Note: See “Port B” on page 64. for a detailed description of how to define the direction of the user defined SPI pins.

**SS Pin Functionality**

When the SPI is configured as a master (MSTR in SPCR is set), the user can determine the direction of the SS pin. If SS is configured as an output, the pin is a general output pin which does not affect the SPI system. If SS is configured as an input, it must be held high to ensure Master SPI operation. If the SS pin is driven low by peripheral circuitry when the SPI is configured as master with the SS pin defined as an input, the SPI system interprets this as another master selecting the SPI as a slave and starting to send data to it. To avoid bus contention, the SPI system takes the following actions:

1. The MSTR bit in SPCR is cleared and the SPI system becomes a slave. As a result of the SPI becoming a slave, the MOSI and SCK pins become inputs.
2. The SPIF flag in SPSR is set, and if the SPI interrupt is enabled and the I-bit in SREG are set, the interrupt routine will be executed.

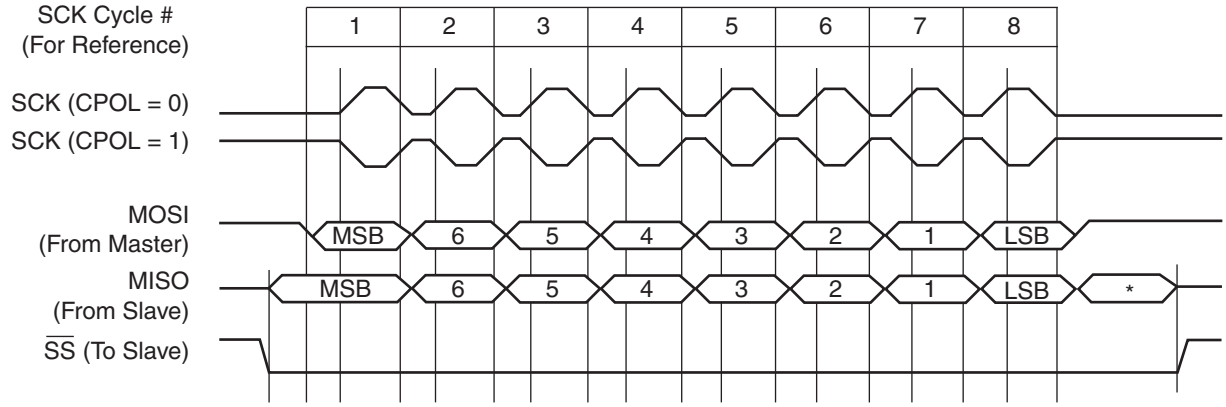
Thus, when interrupt-driven SPI transmittal is used in master mode, and there exists a possibility that SS is driven low, the interrupt should always check that the MSTR bit is still set. Once the MSTR bit has been cleared by a slave select, it must be set by the user to re-enable SPI master mode.

When the SPI is configured as a slave, the SS pin is always input. When SS is held low, the SPI is activated and MISO becomes an output if configured so by the user. All other pins are inputs. When SS is driven high, all pins are inputs, and the SPI is passive, which means that it will not receive incoming data. Note that the SPI logic will be reset once the SS pin is brought high. If the SS pin is brought high during a transmission, the SPI will stop sending and receiving immediately and both data received and data sent must be considered as lost.

**Data Modes**

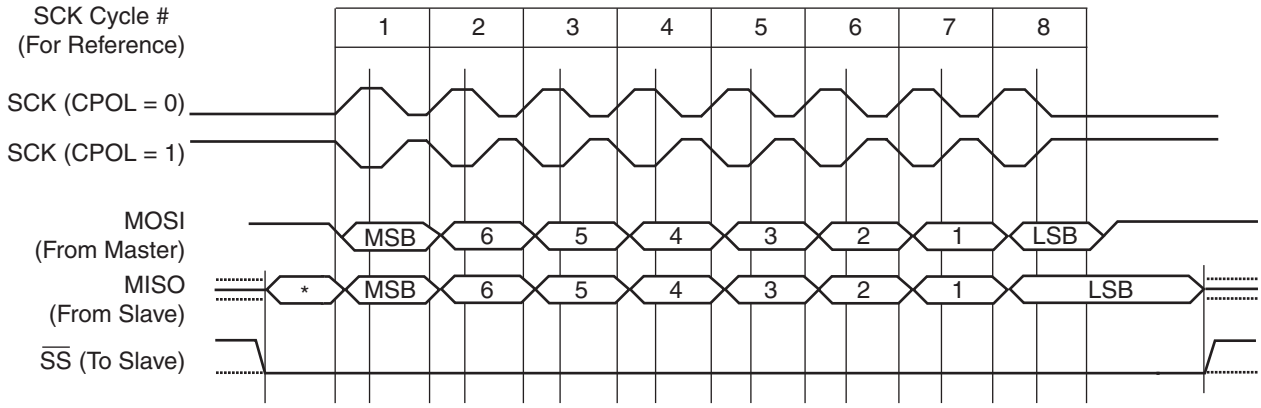
There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 17 and Figure 18.

**Figure 17.** SPI Transfer Format with CPHA = 0 and DORD = 0




Note: * Not defined but normally LSB of character just received.

**Figure 18.** SPI Transfer Format with CPHA = 1 and DORD = 0




Note: * Not defined, but normally LSB of previously transmitted character.

### *SPI Control Register – SPCR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0D ($2D) | **SPIE** | **SPE** | **DORD** | **MSTR** | **CPOL** | **CPHA** | **SPR1** | **SPR0** | **SPCR** |
| 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 – SPIE: SPI Interrupt Enable**

This bit causes the SPI interrupt to be executed if SPIF bit in the SPSR register is set and the global interrupts are enabled.

- **Bit 6 – SPE: SPI Enable**

When the SPE bit is set (one), the SPI is enabled. This bit must be set to enable any SPI operations.

- **Bit 5 – DORD: Data Order**

When the DORD bit is set (one), the LSB of the data word is transmitted first.

When the DORD bit is cleared (zero), the MSB of the data word is transmitted first.

- **Bit 4 – MSTR: Master/Slave Select**

This bit selects Master SPI mode when set (one), and Slave SPI mode when cleared (zero). If SS is configured as an input and is driven low while MSTR is set, MSTR will be cleared, and SPIF in SPSR will become set. The user will then have to set MSTR to re-enable SPI master mode.

- **Bit 3 – CPOL: Clock Polarity**

When this bit is set (one), SCK is high when idle. When CPOL is cleared (zero), SCK is low when idle. Refer to Figure 17 and Figure 18 for additional information.

- **Bit 2 – CPHA: Clock Phase**

Refer to Figure 17 or Figure 18 for the functionality of this bit.

- **Bits 1,0 – SPR1, SPR0: SPI Clock Rate Select 1 and 0**

These two bits control the SCK rate of the device configured as a master. SPR1 and SPR0 have no effect on the slave. The relationship between SCK and the Oscillator Clock frequency $f_{CL}$ is shown in the following table:

**Table 21.** Relationship Between SCK and the Oscillator Frequency

| SPR1 | SPR0 | SCK Frequency |
|---|---|---|
| 0 | 0 | 3 MHz |
| 0 | 1 | 750 kHz |
| 1 | 0 | 187.5 kHz |
| 1 | 1 | 93.75 kHz |

### *SPI Status Register – SPSR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0E ($2E) | **SPIF** | **WCOL** | – | – | – | – | – | – | **SPSR** |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – SPIF: SPI Interrupt Flag**

When a serial transfer is complete, the SPIF bit is set (one) and an interrupt is generated if SPIE in SPCR is set (one) and global interrupts are enabled. If SS is an input and is driven low when the SPI is in master mode, this will also set the SPIF flag. SPIF is cleared by the hardware when executing the corresponding interrupt handling vector. Alternatively, the SPIF bit is cleared by first reading the SPI status register when SPIF is set (one), then accessing the SPI Data Register (SPDR).

- **Bit 6 – WCOL: Write Collision Flag**

The WCOL bit is set if the SPI data register (SPDR) is written during a data transfer. The WCOL bit (and the SPIF bit) are cleared (zero) by first reading the SPI Status Register when WCOL is set (one), and then accessing the SPI Data Register.

- **Bit 5..0 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and will always read as zero.

### *SPI Data Register – SPDR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0F ($2F) | **MSB** | – | – | – | – | – | – | **LSB** | **SPDR** |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | x | x | x | x | x | x | x | x | Undefined |

The SPI Data Register is a read/write register used for data transfer between the register file and the SPI Shift register. Writing to the register initiates data transmission. Reading the register causes the Shift Register Receive buffer to be read.

## UART

The AT43USB320A features a full duplex (separate receive and transmit registers) Universal Asynchronous Receiver and Transmitter (UART). The main features are:

- Baud rate generator that can generate a large number of baud rates (bps)
- High baud rates at low XTAL frequencies
- 8-bit data
- Noise filtering
- Overrun detection
- Framing Error detection
- False Start Bit detection
- Three separate interrupts on TX Complete, TX Data Register Empty and RX Complete

## Data Transmission

A block schematic of the UART transmitter is shown in Figure 19.

Data transmission is initiated by writing the data to be transmitted to the UART I/O Data Register, UDR. Data is transferred from UDR to the Transmit shift register when:

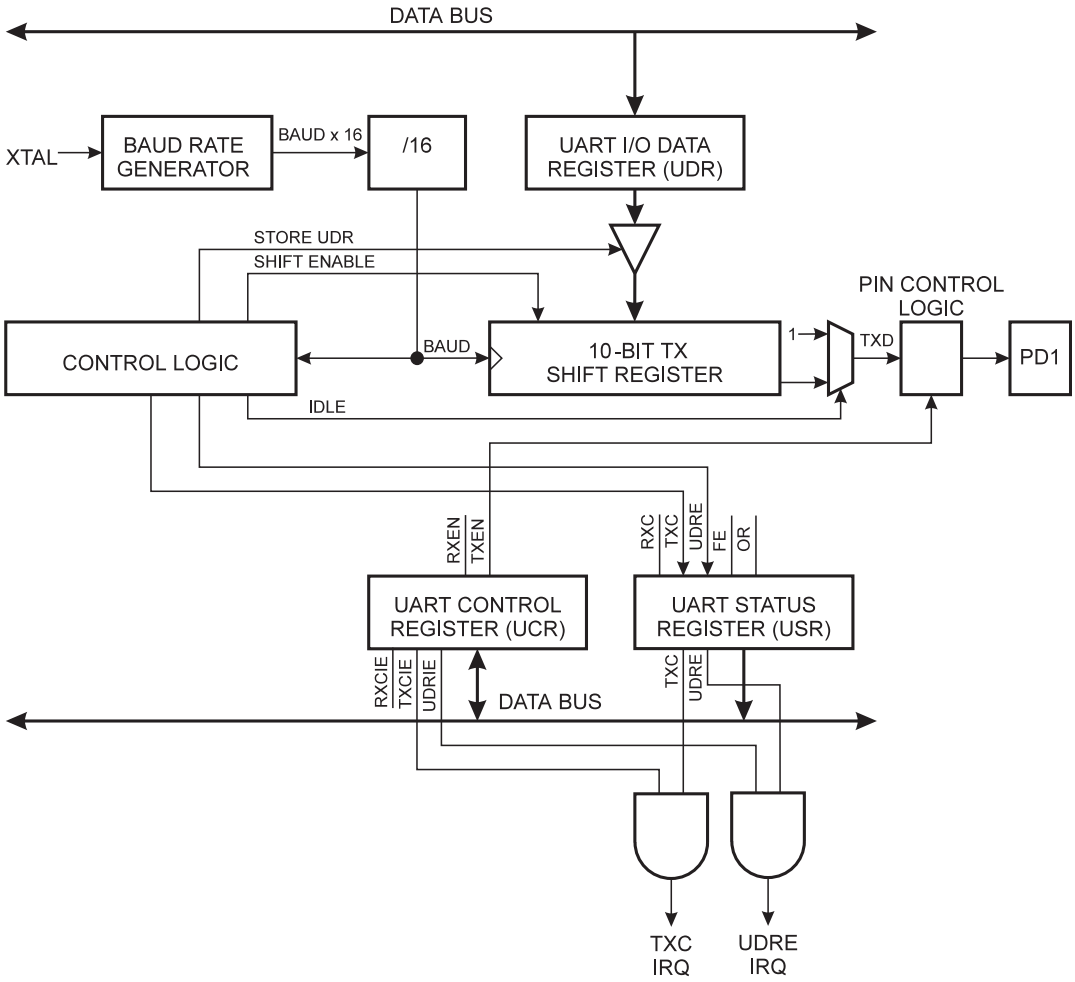
- A new character has been written to UDR after the stop bit from the previous character has been shifted out. The shift register is loaded immediately.
- A new character has been written to UDR before the stop bit from the previous character has been shifted out. The shift register is loaded when the stop bit of the character currently being transmitted has been shifted out.

If the 10-bit Transmitter shift register is empty, data is transferred from UDR to the shift register. At this time the UDR E (UART Data Register Empty) bit in the UART Status Register, USR, is set. When this bit is set (one), the UART is ready to receive the next character. At the same time as the data is transferred from UDR to the 10-bit shift register, bit 0 of the shift register is cleared (start bit) and bit 9 is set (stop bit).

On the baud rate clock following the transfer operation to the shift register, the start bit is shifted out on the TXD pin. The n follows the data, LSB first. When the stop bit has been shifted out, the shift register is loaded if any new data has been written to the UDR during the transmission. During loading, UDRE is set. If there is no new data in the UDR register to send when the stop bit is shifted out, the UDRE flag will remain set until UDR is written again. When no new data has been written and the stop bit has been present on TXD for one bit length, the TX Complete flag (TXC) in USR is set.

The TXEN bit in UCR enables the UART Transmitter when set (one). When this bit is cleared (zero), the PD1 pin can be used for general I/O. When TXEN is set, the UART Transmitter will be connected to PD1, which is forced to be an output pin regardless of the setting of the DDD1 bit in DDRD.

**Figure 19.** UART Transmitter




## Data Reception

Figure 20 shows a block diagram of the UART Receiver.

The receiver front-end logic samples the signal on the RXD pin at a frequency 16 times the baud rate. While the line is idle, one single sample of logical "0" will be interpreted as the falling edge of a start bit and the start bit detection sequence is initiated. Let sample 1 denote the first zero-sample. Following the 1-to-0 transition, the receiver samples the RXD pin at samples 8, 9 and 10. If two or more of these three samples are found to be logical "1"s, the start bit is rejected as a noise spike and the receiver starts looking for the next 1-to-0 transition.

If, however, a valid start bit is detected, sampling of the data bits following the start bit is performed. These bits are also sampled at samples 8, 9 and 10. The logical value found in at least two of the three samples is taken as the bit value. All bits are shifted into the Transmitter Shift register as they are sampled. Sampling of an incoming character is shown in Figure 19.

When the stop bit enters the receiver, the majority of the three samples must be "1" to accept the stop bit. If two or more samples are logical "0's, the Framing Error (FE) flag in the UART Status Register (USR) is set. Before reading the UDR register, the user should always check the FE bit to detect framing errors.

Whether or not a valid stop bit is detected at the end of a character reception cycle, the data is transferred to UDR and the RXC flag in USR is set. UDR is in fact two physically separate reg-

isters, one for transmitted data and one for received data. When UDR is read, the Receive Data register is accessed, and when UDR is written, the Transmit Data register is accessed.

If, after having received a character, the UDR register has not been read since the last receive, the OverRun (OR) flag in UCR is set. This means that the last data byte shifted into the shift register could not be transferred to UDR and has been lost. The OR bit is buffered and is updated when the valid data byte in UDR is read. Thus, the user should always check the OR bit after reading the UDR register in order to detect any overruns if the baud rate is high or CPU load is high.

When the RXEN bit in the UCR register is cleared (zero), the receiver is disabled. This means that the PDO pin can be used as a general I/O pin. When RXEN is set, the UART Receiver will be connected to PDO, which is forced to be an input pin regardless of the setting of the DDDO bit in DDRD. When PDO is forced to input by the UART, the PORTDO bit can still be used to control the pull-up resistor on the pin.

**Figure 20.** UART Receiver

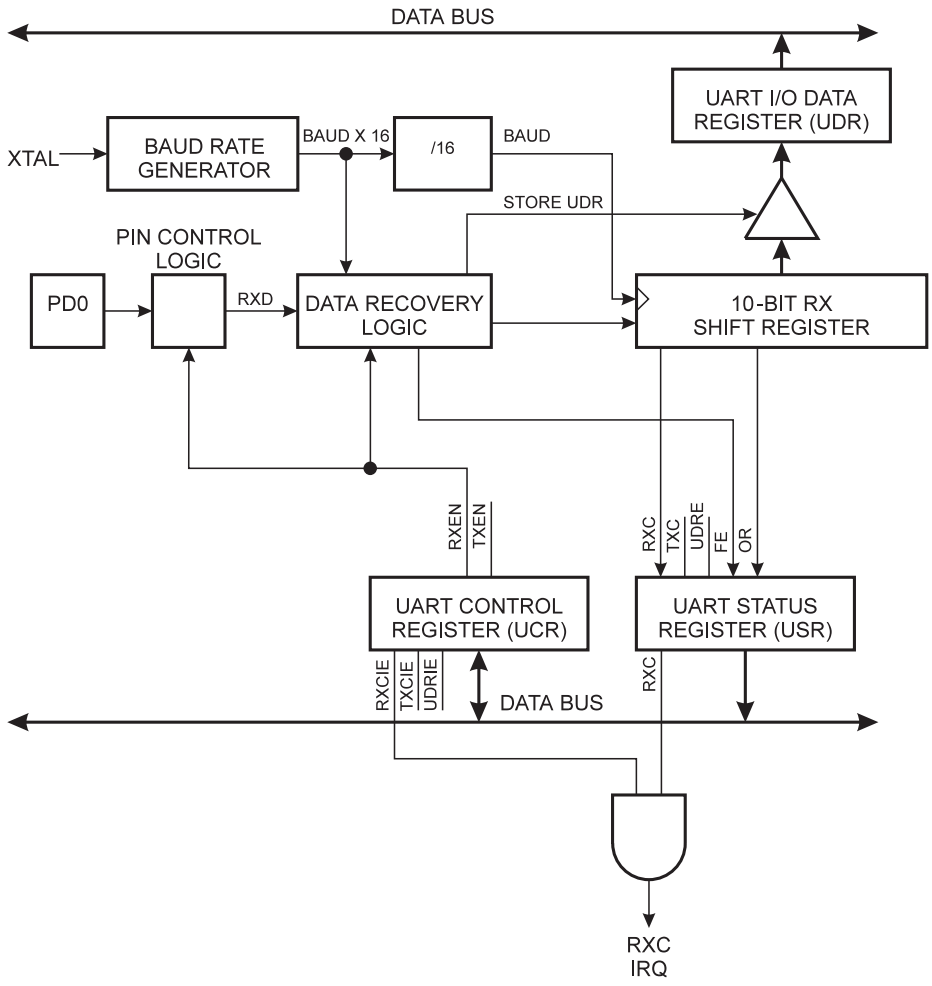
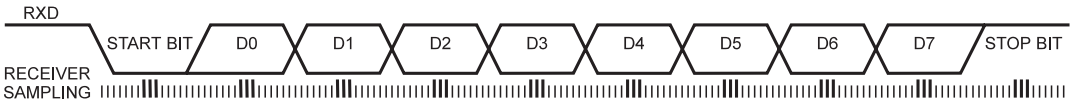



**Figure 21.** Sampling Received Data



## UART Control

### UART I/O Data Register – UDR

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0D ($2C) | **MSB** | – | – | – | – | – | – | **LSB** | UDR |
| 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 UDR register is actually two physically separate registers sharing the same I/O address. When writing to the register, the UART Transmit Data register is written. When reading from UDR, the UART Receive Data register is read.

### UART Status Register – USR

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0D ($2B) | **RXC** | **TXC** | **UDRE** | **FE** | **OR** | – | – | – | USR |
| Read/Write | R/W | R/W | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The USR register is a read-only register providing information on the UART status.

- **Bits 7 – RXC: UART Receive Complete**

This bit is set (one) when a received character is transferred from the Receiver Shift register to UDR. The bit is set regardless of any detected framing errors. When the RXCIE bit in UCR is set, the UART Receive Complete interrupt will be executed when RXC is set (one). RXC is cleared by reading UDR. When interrupt-driven data reception is used, the UART Receive Complete Interrupt routine must read UDR in order to clear RXC, otherwise a new interrupt will occur once the interrupt routine terminates.

- **Bit 6 – TXC: UART Transmit Complete**

This bit is set (one) when the entire character (including the stop bit) in the Transmit Shift register has been shifted out an d no new data has been written to UDR. This flag is especially useful in half-duplex communications interfaces, where a transmitting application must enter receive mode and free the communications bus immediately after completing the transmission.

When the TXCIE bit in UCR is set, setting of TXC causes the UART Transmit Complete interrupt to be executed. TXC is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, the TXC bit is cleared (zero) by writing a logical "1"to the bit.

- **Bit 5 – UDRE: UART Data Register Empty**

This bit is set (one) when a character written to UDR is transferred to the Transmit Shift register. Setting of this bit indicates that the transmitter is ready to receive a new character for transmission.

When the UDRIE bit in UCR is set, the UART Transmit Complete interrupt to be executed as long as UDRE is set. UDRE is cleared by writing UDR. When interrupt-driven data transmittal is used, the UART Data Register Empty Interrupt routine must write UDR in order to clear UDRE, otherwise a new interrupt will occur once the interrupt routine terminates. UDRE is set (one) during reset to indicate that the transmitter is ready.

- **Bit 4 – FE: Framing Error**

This bit is set if a Framing Error condition is detected, i.e., when the stop bit of an incoming character is zero. The FE bit is cleared when the stop bit of received data is one.

- **Bit 3 – OR: Overrun**

This bit is set if an Overrun condition is detected, i.e., when a character already present in the UDR register is not read before the next character has been shifted into the Receiver Shift register. The OR bit is buffered, which means that it will be set once the valid data still in UDRE is read.

The OR bit is cleared (zero) when data is received and transferred to UDR.

- **Bits 2...0 – Res: Reserved Bits**

These bits are reserved bits in the AT43USB320A and will always read as zero.

**UART Control Register – UCR**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $0A ($2A) | **RXCIE** | **TXCIE** | **UDRIE** | **RXEN** | **TXEN** | **-** | **-** | **-** | UCR |
| Read/Write | R/W | R/W | R/W | R/W | R/W | | R | R/W | |
| Initial value | 0 | 0 | 0 | 0 | 0 | | 1 | 0 | |

- **Bit 7 – RXCIE: RX Complete Interrupt Enable**

When this bit is set (one), a setting of the RXC bit in USR will cause the Receive Complete Interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 6 – TXCIE: TX Complete Interrupt Enable**

When this bit is set (one), a setting of the TXC bit in USR will cause the Transmit Complete Interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 5 – UDRIE: UART Data Register Empty Interrupt Enable**

When this bit is set (one), a setting of the UDRE bit in USR will cause the UART Data Register Empty Interrupt routine to be executed provided that global interrupts are enabled.

- **Bit 4 – RXEN: Receiver Enable**

This bit enables the UART receiver when set (one). When the receiver is disabled, the TXC, OR and FE status flags cannot become set. If these flags are set, turning off RXEN does not cause them to be cleared.

- **Bit 3 – TXEN: Transmitter Enable**

This bit enables the UART transmitter when set (one). When disabling the transmitter while transmitting a character, the transmitter is not disabled before the character in the shift register plus any following character in UDR has been completely transmitted.

- **Bit 2 ..0 – RES: Reserved Bits**

These bits are the reserved bits of the AT43USB320A.

## Baud Rate Generator

The baud rate generator is a frequency divider that generates baud rates according to the following equation:

BAUD = SYSCLK/16(UBRR + 1)

- **BAUD = Baud rate**
- **SYSCLK = 16 MHz**
- **UBRR = Contents of the UART Baud Rate register, UBRR (0 – 255)**

For standard crystal frequencies, the most commonly used baud rates can be generated by using the UBRR settings in Table 22. UBRR values that yield an actual baud rate differing less than 2% from the target baud rate are boldface in the table. However, using baud rates that have more than 1% error is not recommended. High error ratings give less noise immunity.

**Table 22.** UBRR Settings

| Baud Rate | UBRR | % Error |
|---|---|---|
| 2400 | 416 | 0.08 |
| 4800 | 207 | 0.16 |
| 9600 | 103 | 0.16 |
| 14400 | 68 | 0.64 |
| 19200 | 51 | 0.16 |
| 28800 | 34 | 0.79 |
| 38400 | 25 | 0.16 |
| 57600 | 16 | 2.12 |
| 76800 | 12 | 0.16 |
| 115200 | 8 | 3.55 |

### UART BAUD Rate Register – UBRR

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $09 ($29) | **MSB** | | | | | | | **LSB** | UBRR |
| 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 UBRR register is an 8-bit read/write register that specifies the UART Baud Rate according to the equation on the previous page.

## I/O-Ports

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies for changing drive value if configured as output or enabling/disabling of pull-up resistors if configured as input.

### Port A

Port A is an 8-bit bi-directional I/O port. The Port A output buffers can sink or source 4 mA

Three I/O memory address locations are allocated for the Port A, one each for the Data Register PORTA, $1B($3B), Data Direction Register (DDRA), $1A($3A) and the Port A Input Pins

(PINA) $19($39). The Port A Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

The port pins have no selectable pull-up resistors.

***Port A Data Register – PORTA***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1B ($3B) | PORTA7 | PORTA6 | PORTA5 | PORTA4 | PORTA3 | PORTA2 | PORTA1 | PORTA0 | PORTA |
| 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 | |

***Port A Data Direction Register – DDRA***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1A ($3A) | DDA7 | DDA6 | DDA5 | DDA4 | DDA3 | DDA2 | DDA1 | DDA0 | DDRA |
| 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 | |

***Port A Input Pins Address – PINA***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $19 ($39) | PINA7 | PINA6 | PINA5 | PINA4 | PINA3 | PINA2 | PINA1 | PINA0 | PINA |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | |

The Port A Input Pins address (PINA) is not a register, and this address enables access to the physical value on each Port A pin. When reading PORTA the Port A Data Latch is read, and when reading PINA, the logical values present on the pins are read.

**Port A as General Digital I/O**

All 8 pins in Port A have equal functionality when used as digital I/O pins.

**PAn, General I/O Pin:** The DDAn bit in the DDRA register selects the direction of this pin, if DDAn is set (one), PAn is configured as an output pin. If DDAn is cleared (zero), PAn is configured as an input pin. If PORTAn is set (one) when the pin is configured as an input pin, the MOS pull-up resistor is activated. To switch the pull-up resistor off, the PORTAn has to be cleared (zero) or the pin has to configured as an output pin. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not active.

**Table 23.** DDAn Effects on Port A Pins

| DDAn | PORTAn | I/O | Comment |
|---|---|---|---|
| 0 | 0 | Input | Tri-state (Hi-Z) |
| 0 | 1 | Input | Tri-state (Hi-Z) |
| 1 | 0 | Output | Push-Pull Zero Output |
| 1 | 1 | Output | Push-Pull One Output |

Note: n: 7,6...0, pin number.

**Port B**

Port B is an 8-bit bi-directional I/O port. The Port B output buffers can sink or source 4 mA.

Three I/O memory address locations are allocated for the Port B, one each for the Data Register - PORTB, $18($38), Data Direction Register (DDRB), $17($37) and the Port B Input Pins (PINB), $16($36). The Port B Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

The port pins have no selectable pull-up resistors.

The Port B pins with alternate functions are shown in the following table:

**Table 24.** Port B Pins Alternate Functions

| Port Pin | Alternate Functions |
|---|---|
| PB0 | T0 (Timer/Counter 0 External Counter Input) |
| PB1 | T1 (Timer/Counter 1 External Counter Input) |
| PB4 | SS (SPI Slave Select Input) |
| PB5 | MOSI (SPI Bus Master Output/Slave Input) |
| PB6 | MISO (SPI Bus Master Input/Slave Output) |
| PB7 | SCK (SPI Bus Serial Clock) |

When the pins are used for the alternate function the DDRB and PORTB register has to be set according to the alternate function description.

### *Port B Data Register – PORTB*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $18 ($38) | PORTB7 | PORTB6 | PORTB5 | PORTB4 | PORTB3 | PORTB2 | PORTB1 | PORTB0 | PORTB |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

### *Port B Data Direction Register – DDRB*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $17 ($37) | DDB7 | DDB6 | DDB5 | DDB4 | DDB3 | DDB2 | DDB1 | DDB0 | DDRB |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

### *Port B Input Pins Address – PINB*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $16 ($36) | PINB7 | PINB6 | PINB5 | PINB4 | PINB3 | PINB2 | PINB1 | PINB0 | PINB |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | |

The Port B Input Pins address (PINB) is not a register, and this address enables access to the physical value on each Port B pin. When reading PORTB, the Port B Data Latch is read, and when reading PINB, the logical values present on the pins are read.

**Port B as General Digital I/O**

All 8 pins in port B have equal functionality when used as digital I/O pins.

**PBn, General I/O Pin:** The DDBn bit in the DDRB register selects the direction of this pin, if DDBn is set (one), PBn is con-figured as an output pin. If DDBn is cleared (zero), PBn is configured as an input pin. If PORTBn is set (one) when the pin is configured as an input pin, the MOS pull-up resistor is activated. To switch the pull-up resistor off, the PORTBn has to be cleared (zero) or the pin has to configured as an output pin. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not active.

**Table 25.** DDBn Effects on Port B Pins

| DDBn | PORTBn | I/O | Comment |
|---|---|---|---|
| 0 | 0 | Input | Tri-state (Hi-Z) |
| 0 | 1 | Input | Tri-state (Hi-Z) |
| 1 | 0 | Output | Push-Pull Zero Output |
| 1 | 1 | Output | Push-Pull One Output |

Note: n: 7, 6...0, pin number.

## Port C

Port C is an 8-bit bi-directional I/O port with push-pull outputs. The Port C output buffers can sink 4 mA

Three I/O memory address locations are allocated for the Port C, one each for the Data Register – PORTC, $15($35), Data Direction Register – DDRC, $14($34) and the Port C Input Pins – PINC, $13($33). The Port C Input Pins address is read only, while the Data Register and the Data Direction Register are read/write.

***Port C Data Register – PORTC***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $15 ($35) | PORTC7 | PORTC6 | PORTC5 | PORTC4 | PORTC3 | PORTC2 | PORTC1 | PORTC0 | PORTC |
| 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 | |

***Port C Data Direction Register – DDRC***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $14 ($34) | DDC7 | DDC6 | DDC5 | DDC4 | DDC3 | DDC2 | DDC1 | DDC0 | DDRC |
| 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 | |

***Port C Input Pins Address – PINC***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $13 ($33) | PINC7 | PINC6 | PINC5 | PINC4 | PINC3 | PINC2 | PINC1 | PINC0 | PINC |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial value | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | |

The Port C Input Pins address PINC is not a register, and this address enables access to the physical value on each Port C pin. When reading PORTC, the Port C Data Latch is read, and when reading PINC, the logical values present on the pins are read.

### Port C as General Digital I/O

All 8 pins in Port C have equal functionality when used as digital I/O pins.

PCn, General I/O pin: The DDCn bit in the DDRC register selects the direction of this pin, if DDCn is set (one), PCn is configured as an output pin. If DDCn is cleared (zero), PCn is configured as an input pin. The value of PORTCn has no meaning in this mode. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not active.

**Table 26.** DDCn Effects on Port C Pins

| DDCn | PORTCn | I/O | Comment |
|---|---|---|---|
| 0 | 0 | Input | Tri-state (Hi-Z) |
| 0 | 1 | Input | Tri-state (Hi-Z) |
| 1 | 0 | Output | Push-pull Zero Output |
| 1 | 1 | Output | Push-pull One Output |

Note: n: 7…0, pin number

## Port D

Port D is an 8-bit bi-directional I/O port. Its output buffers can sink or source 2 mA.

Three I/O memory address locations are allocated for the Port D, one each for the Data Register - PORTD, $12($32), Data Direction Register (DDRD), $11($31) and the Port D Input Pins (PIND), $10($30). The Port D Input Pins' address is read only, while the Data Register and the Data Direction Register are read/write.

The port pins have no selectable pull-up resistors.

Some Port D pins have alternate functions as shown in Table 27.

**Table 27.** Port D Alternate Functions

| Port Pin | Alternate Function |
|---|---|
| PD0 | RXD (UART Input Line) |
| PD1 | TXD (UART Output Line) |
| PD2 | INT0, External Interrupt 0 |
| PD3 | INT1, External Interrupt 1 |
| PD5 | OC1A Timer/Counter1 Output Compare A |

When the pins are used for the alternate function the DDRD and PORTD register has to be set according to the alternate function description.

***Port D Data Register – PORTD***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $12 ($32) | PORTD7 | PORTD6 | PORTD5 | PORTD4 | PORTD3 | PORTD2 | PORTD1 | PORTD0 | PORTD |
| 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 | |

***Port D Data Direction Register – DDRD***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $11 ($31) | DDD7 | DDD6 | DDD5 | DDD4 | DDD3 | DDD2 | DDD1 | DDD0 | DDRD |
| 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 | |

***Port D Input Pins Address – PIND***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $10 ($30) | PIND7 | PIND6 | PIND5 | PIND4 | PIND3 | PIND2 | PIND1 | PIND0 | PIND |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A | |

The Port D Input Pins address (PIND) is not a register, and this address enables access to the physical value on each Port D pin. When reading PORTD, the Port D Data Latch is read, and when reading PIND, the logical values present on the pins are read.

### Port D as General Digital I/O

**PDn, General I/O Pin:** The DDDn bit in the DDRD register selects the direction of this pin. If DDDn is set (one), PDn is con-figured as an output pin. If DDDn is cleared (zero), PDn is configured as an input pin. If PORTDn is set (one) when the pin is configured as an input pin, the MOS pull-up resistor is activated. To switch the pull-up resistor off, the PORTDn has to be cleared (zero) or the pin has to configured as an output pin. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not active.

**Table 28.** DDDn Bits on Port D Pins

| DDDn | PORTDn | I/O | Comment |
|---|---|---|---|
| 0 | 0 | Input | Tri-state (Hi-Z) |
| 0 | 1 | Input | Tri-state (Hi-Z) |
| 1 | 0 | Output | Push-Pull Zero Output |
| 1 | 1 | Output | Push-Pull One Output |

Note: n: 7, 6...0, pin number.

## Programming the USB Module

The USB hardware consists of two devices, hub and function, each with their own device address and endpoints. Its operation is controlled through a set of memory mapped registers. The exact configuration of the USB device is defined by the software and it can be programmed to operate as a compound device, or as a hub only or as a function only. The hub has the required control and interrupt endpoints. The number of external downstream ports is programmable from 0 to 4. The DP and DM pins of the unused port(s) must be connected to ground. The USB function has one control endpoint and 2 programmable endpoints. All the endpoints have their own 8-byte FIFO. If the hub is disabled, one extra endpoint becomes available to the function.

## The USB Function

The USB function hardware is designed to operate in the single packet mode and to manage the USB protocol layer. It consists of a Serial Interface Engine (SIE), endpoint FIFOs and a Function Interface Unit (FIU). The SIE performs the following tasks: USB signaling detection/generation, data serialization/de-serialization, data encoding/decoding, bit stuffing and unstuffing, clock/data separation, and CRC generation/checking. It also decodes and manages all packet data types and packet fields.

The endpoint FIFO buffers the data to be sent out or data received. The FIU manages the flow of data between the SIE, FIFO and the internal microcontroller bus. It controls the FIFO and monitors the status of the transactions and interfaces to the CPU. It initiates interrupts and acts upon commands sent by the firmware.

The USB function hardware of the AT43USB320A makes the physical interface and the protocol layer transparent to the user. To start the process, the firmware must first enable the endpoints and which place them in receive mode by default. The device address by default is address 0. The USB function hardware then waits for a setup token from the host. When a valid the setup token is received, it automatically stores the data packet in endpoint 0 FIFO and responds with an ACK. It then notifies the microcontroller through an interrupt. The microcontroller reads the FIFO and parses the request.

Transactions for the non-control endpoints are even simpler. Once the endpoint is enabled, it waits for an IN or an OUT token depending whether it is programmed as an IN or OUT endpoint. For example, if it is an IN endpoint, the microcontroller simply loads the data into the endpoint's FIFO and sets a bit in the control and status register. The USB hardware will assemble the data in a USB packet and waits for an IN token. When it receives one, it automatically responds by transmitting the data packet and completes the transaction by waiting for the host's ACK. When one is received, the USB hardware will signal the microcontroller

that the transaction has been completed successfully. Retries and data toggles are performed automatically by the USB hardware. When the IN endpoint is not ready to send data, in the case where the microcontroller has not filled the FIFO, it will automatically respond with a NAK.

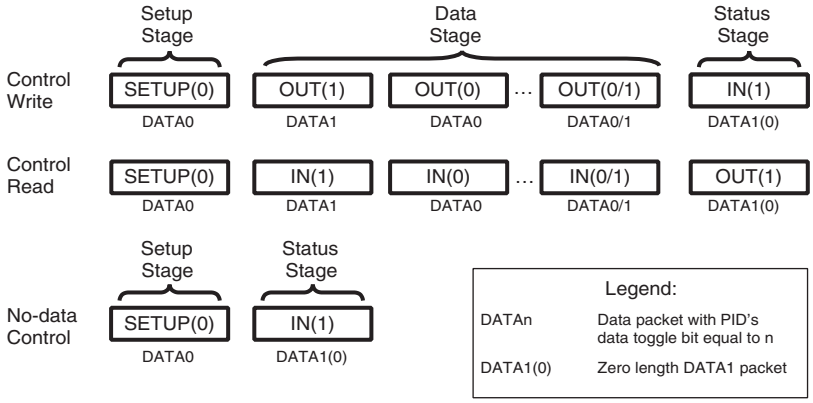
Similarly, an OUT endpoint will wait for an OUT token. When one is received, it will store the data in the FIFO, completes the transaction and interrupt the microcontroller, which then reads the FIFO and enables the endpoint for the next packet. If the FIFO is not cleared, the USB hardware will responds with a NAK.

A detailed description of how USB transactions are handled is described in the following sections. First for a control endpoint and then for non-control endpoints.

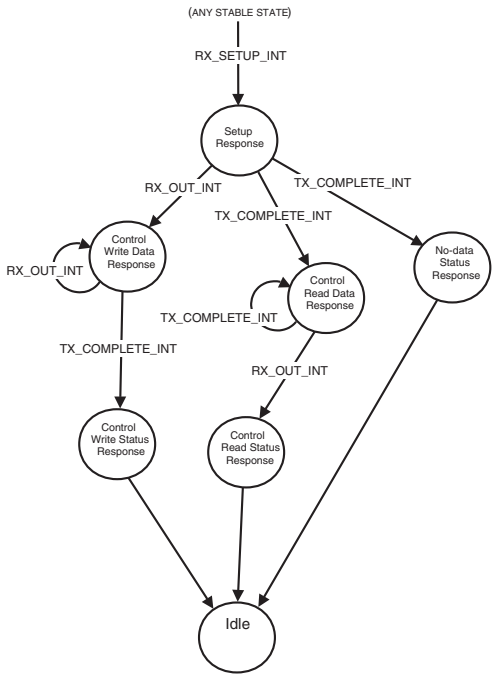
### Control Transfers at Control Endpoint EP0

The description given below is for the function control endpoint, but applies to the hub control endpoint as well if the proper registers are used.

The following illustration describes the three possible types of control transfers – Control Write, Control Read and No-data control:



The following state diagram shows how the various state transitions are triggered. Additional decision making may take place within the response states to determine the next expected state. Unmarked arcs represent transitions that trigger immediately following completion of the response state processing. Stable states, those requiring an interrupt to exit having no unmarked arcs as exit paths, are shown in bold.



The following information describes how the AT43USB320A's USB hardware and firmware operates during a control transfer between the host and the hub's or function's control endpoint.

Legend: DATA1/DATA0 = Data packet with DATA1 or DATA2 PID

DATA1(0) = Zero length DATA1 packet

**Idle State**

*This is the default state from power-up.*

**Setup Response State**

*The Function Interface Unit (FIU) receives a SETUP token with 8 bytes of data from the Host. The FIU stores the data in the FIFO, sends an ACK back to the host and asserts an RX_SETUP interrupt.*

| Hardware | Firmware |
|---|---|
| 1. SETUP token, Data from Host | |
| 2. ACK to Host | |
| 3. Store data in FIFO | |
| 4. Set RX SETUP → INT | |
| | 5. Read UISR |
| | 6. Read CSR0 |
| | 7. Read Byte Count |
| | 8. Read FIFO |
| | 9. Parse command data |
| | 10. Write to H/FCAR0:<br>a. If Control Read: set DIR, clear RX SETUP, fill FIFO, set TX Packet Ready in CAR0<br>b. If Control Write: clear DIR in CAR0<br>c. If no Data Stage: set Data End, clear DIR, set Force STALL in CAR0 |
| | 11. Set UIAR[EP0 INTACK] to clear the interrupt source |

**No-data Status Response State**

*The Function Interface Unit receives an IN token from the Host. The FIU responds with a zero length DATA1 packet until receiving an ACK from the host, then asserts a TX_COMPLETE interrupt.*

| Hardware | Firmware |
|---|---|
| 1. IN token from Host | |
| 2. Send DATA1(0) | |
| 3. ACK from Host | |
| 4. Set TX COMPLETE → INT | |
| | 5. Read UISR |
| | 6. Read CSR0 |
| | 7. If SET ADDRESS, program the new Address, set ADD_EN bit |
| | 8. Clear TX_COMPLETE, clear Data End, set Force STALL in CAR0 |
| | 9. Set UIAR[EP0 INTACK] |

**Control Read Data Response State**

*The Function Interface Unit receives an IN token from the Host. The FIU responds with NAKs until TX_PACKET_READY is set. The FIU then sends the data in the FIFO upstream, retrying until it successfully receives an ACK from the host. Finally, the FIU clears the TX_PACKET_READY bit and asserts a TX_COMPLETE interrupt.*

| Hardware | Firmware |
|---|---|
| 1. IN token from Host | |
| 2. a. If TX Packet Ready = 1, send DATA0/DATA1<br>b. If TX Packet Ready = 0, send NAK | |
| 3. ACK from Host | |
| 4. Clear TX Packet Ready<br>Set TX Complete → INT | |
| | 5. Read UISR |
| | 6. Read CSR0 |
| | 7. Clear TX COMPLETE in CAR0:<br>a. If more data: fill FIFO, set TX Packet Ready, set DIR in CAR0<br>b. If no more data: set Force STALL, set DATA END in CAR0 |
| | 8. Set UIAR[EP0 INTACK] to clear interrupt source |
| Repeat steps 1 through 8 | |

**Control Read Status Response State**

*The Function Interface Unit receives an OUT token from the Host with a zero length DATA1 packet. The FIU responds with a NAK until TX_COMPLETE is cleared. The FIU will then ACK the retried OUT token from the Host and assert an RX_OUT interrupt.*

| Hardware | Firmware |
|---|---|
| 1. OUT token from Host | |
| 2. DATA1(0) from Host | |
| 3. TX Complete = 0 ?<br>a. If yes, ACK to Host<br>Set RX OUT → INT<br>b. If no, NAK to Host | |
| | 4. Read UISR |
| | 5. Read CSR0 |
| | 6. Clear RX OUT, set Data End, set Force Stall in H/FCAR0.<br>Note: A SETUP token will clear Data End, therefore, it is not cleared by FW in case Host retries. |
| | 7. Set UIAR[EP0 INTACK] to clear interrupt source |

**Control Write Data Response State**

*The Function Interface Unit receives an OUT token from the Host with a DATA packet. The FIU places the incoming data into the FIFO, issues an ACK to the host, and asserts an RX_OUT interrupt.*

| Hardware | Firmware |
|---|---|
| 1. OUT token from Host | |
| 2. Put DATA0/DATA1 into FIFO | |
| 3. ACK to Host | |
| 4. Set RX OUT → INT | |
| | 5. Read UISR |
| | 6. Read CSR0 |
| | 7. Read FIFO |
| | 8. Clear RX OUT<br>If last data packet, set Force STALL, set DATA END. |
| | 9. Set UIAR[EP0 INTACK] to clear the interrupt source |

Repeat steps 1 through 9 until last DATA PACKET:

**Control Write Status Response State**

*The Function Interface Unit receives an IN token from the Host. The FIU responds with a zero length DATA1 packet, retrying until it receives an ACK back from the Host. The FIU then asserts a TX_COMPLETE interrupt.*

| Hardware | Firmware |
|---|---|
| 1. IN token from Host | |
| 2. Send Data1(0) | |
| 3. ACK from Host | |
| 4. Set TX Complete → INT | |
| | 5. Read UISR |
| | 6. Read CSR0 |
| | 7. Clear TX COMPLETE, clear Data End, set Force STALL in CAR0 |
| | 8. Set UIAR[EP0 INTACK] to clear the interrupt source |

**Interrupt/Bulk IN Transfers at Function Endpoint**

The firmware must first condition the endpoint through the Endpoint Control Register, FENDP1/2_CNTR:

Set endpoint direction: set EPDIR

Set interrupt or bulk: EPTYPE = 11 or 10

Enable endpoint: set EPEN

*The Function Interface Unit receives an IN token from the Host. The FIU responds with NAKs until TX_PACKET_READY is set. The FIU then sends the data in the FIFO upstream, retrying until it successfully receives an ACK from the host. Finally, the FIU clears the TX_PACKET_READY bit and asserts a TX_COMPLETE interrupt.*

1. Read UISR
2. Read FCSR1/2
3. Clear TX_COMPLETE

   If more data: fill FIFO, set TX Packet Ready

   Wait for TX_COMPLETE interrupt

   If no more data: set DATA END in FCAR1/2
4. Set UIAR[FEP1/2 INTACK] to clear the interrupt source

**Interrupt/Bulk OUT Transfers at Function Endpoint EP1 and 2**

The firmware must first condition the endpoint through the Endpoint Control Register, FENDP1/2_CNTR:

Set endpoint direction: clear EPDIR

Set interrupt or bulk: EPTYPE = 11 or 10

Enable endpoint: set EPEN

*The Function Interface Unit receives an OUT token from the Host with a DATA packet. The FIU places the incoming data into the FIFO, issues an ACK to the host, and asserts an RX_OUT interrupt.*

1. Read UISR
2. Read FCSR1/2
3. Read FIFO
4. Clear RX_OUT

   If more data:

   Wait for RX_OUT interrupt

   If no more data: set DATA END
5. Set UIAR[FEP1/2 INTACK] to clear the interrupt source

## USB Registers

The following sections describe the registers of the AT43USB320A's USB hub and function units.

Reading a bit for which the microcontroller does not have read access will yield a zero value result. Writing to a bit for which the microcontroller does not have write access has no effect.

### Hub Address Register – HADDR

The USB hub contains an address register that contains the hub address assigned by the host. This Hub Address Register must be programmed by the microcontroller once it has received a SET_ADDRESS request from the host. The USB hardware uses the new address only after the status phase of the transaction is completed when the microcontroller has enabled the new address by setting bit 0 of the Global State Register. After power-up or reset, this register will contain the value of 0x00.

***Hub Address Register – HADDR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FEF | **SAEN** | **HADD6** | **HADD5** | **HADD4** | **HADD3** | **HADD2** | **HADD1** | **HADD0** | **HADDR** |
| 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 – SAEN: Single Address Enable**

The Single Address Enable bit allows the microcontroller to configure the AT43USB320A into a single address or a composite device. Once this capability is enabled, the hub endpoint 0 (HEP0) is converted from a control endpoint to a programmable function endpoint FEP3; all the endpoints would then operate on the single address.

- **Bit 6..0 – HADD6...0: Hub Address[6:0]**

### Function Address Register – FADDR

The USB function contains an address register that contains the function address assigned by the host. This Function Address Register must be programmed by the microcontroller once it has received a SET_ADDRESS request from the host and completed the status phase of the transaction. After power up or reset, this register will contain the value of 0x00.

***Function Address Register – FADDR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FEE | **FEN** | **FADD6** | **FADD5** | **FADD4** | **FADD3** | **FADD2** | **FADD1** | **FADD0** | **FADDR** |
| 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 – FEN: Function Enable**

The Function Enable bit (FEN) allows the firmware to enable or disable the function endpoints. The firmware will set this bit after receipt of a reset through the hub, SetPortFeature[PORT_RESET]. Once this bit is set, the USB hardware passes to and from the host.

When the Single Address bit is set, the condition of FEN is ignored.

- **Bit 6..0 – FADD6...0: Function Address[6:0]**

## Endpoint Registers

***Hub Endpoint 0 Control Register – HENDP0_CR***

***Function Endpoint 0 Control Register – FENDP0_CR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FE7 | **EPEN** | **–** | **–** | **–** | **DTGLE** | **EPDIR** | **EPTYPE1** | **EPTYPE0** | **HENDP0_CR** |
| $24 ($44) | **EPEN** | **–** | **–** | **–** | **DTGLE** | **EPDIR** | **EPTYPE1** | **EPTYPE0** | **FENDP0_CR** |
| Read/Write | R/W | R | R | R | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – EPEN: Endpoint Enable**

0 = Disable endpoint

1 = Enable endpoint

- **Bit 6..4 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 3 – DTGLE: Data Toggle**

Identifies DATA0 or DATA1 packets. This bit will automatically toggle and requires clearing by the firmware only in certain special circumstances.

- **Bit 2 – EPDIR: Endpoint Direction**

0 = Out

1 = In

- **Bit 1, 0 – EPTYPE: Endpoint Type**

These bits must be programmed as 0, 0.

### *Function Endpoint 1, 2 Control Register – FENDP1,2_CR*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FE4 | **EPEN** | **–** | **–** | **–** | **DTGLE** | **EPDIR** | **EPTYPE1** | **EPTYPE0** | **FENDP1_CR** |
| $1FE3 | **EPEN** | **–** | **–** | **–** | **DTGLE** | **EPDIR** | **EPTYPE1** | **EPTYPE0** | **FENDP2_CR** |
| Read/Write | R/W | R | R | R | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – EPEN: Endpoint Enable**

0 = Disable endpoint

1 = Enable endpoint

- **Bit 6..4 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 3 – DTGLE: Data Toggle**

Identifies DATA0 or DATA1 packets. This bit will automatically toggle and requires clearing by the firmware only in certain special circumstances.

- **Bit 2 – EPDIR: Endpoint Direction**

0 = Out

1 = In

- **Bit 1, 0 – EPTYPE: Endpoint Type**

These bits programs the type of endpoint.

| Bit1 | Bit0 | Type |
|---|---|---|
| 0 | 1 | Isochronous |
| 1 | 0 | Bulk |
| 1 | 1 | Interrupt |

### *Hub Endpoint 0 Data Register – HDR0*

### *Function Endpoint 0..2 Data Register – FDR0..2*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FD7 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 | HDR0 |
| $1FD5 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 | FDR0 |
| $1FD4 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 | FDR1 |
| $1FD3 | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 | FDR2 |
| 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 | |

This register is used to read data from or to write data to the Hub Endpoint 0 FIFO.

- **Bit 7..0 – FDAT7..0: FIFO Data**

Hub endpoint 1 has a single byte data register instead of a FIFO. This data register contains the hub and port status change bitmap. This data register is automatically updated by the USB hardware and is not accessible by the firmware. The bits in this register when read by the host will be:

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $ | – | – | P5 SC | P4 SC | P3 SC | P2 SC | P1 SC | H SC | HDR1 |
| 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 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 5 – P5 SC: Port 5 Status Change**
- **Bit 4 – P4 SC: Port 4Status Change**
- **Bit 3 – P3 SC: Port 3 Status Change**
- **Bit 2 – P2 SC: Port 2 Status Change**
- **Bit 1 – P1 SC: Port 1 Status Change**
- **Bit 0 – H SC: Hub Status Change**

### *Hub Endpoint 0 Byte Count Register – HBYTE_CNT0*

### *Function Endpoint 0..2 Byte Count Register – FBYTE_CNT0..2*

The contents of these registers stores the number of bytes to be sent or that was received by USB Hub and Function endpoints. This count includes the 16-bit CRC. To get the actual byte count of the data, subtract the count in the register by 2. Hub endpoint 1 has no byte count register.

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Hub EP0 $1FCF | – | – | – | **BYTCT4** | **BYTCT3** | **BYTCT2** | **BYTCT1** | **BYTCT0** | **HBYTE_CNT0** |
| Function EP0 $1FCD | – | – | – | **BYTCT4** | **BYTCT3** | **BYTCT2** | **BYTCT1** | **BYTCT0** | **FBYTE_CNT0** |
| Function EP1 $1FCC | – | – | – | **BYTCT4** | **BYTCT3** | **BYTCT2** | **BYTCT1** | **BYTCT0** | **FBYTE_CNT1** |
| Function EP2 $1FCB | – | – | – | **BYTCT4** | **BYTCT3** | **BYTCT2** | **BYTCT1** | **BYTCT0** | **FBYTE_CNT2** |
| Read/Write | R | R | R | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..5 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 4..0 – BYTCT4..0: Byte Count – Length of Endpoint Data Packet**

### *Hub Endpoint 0 Service Routine Register – HCSR0*

### *Function Endpoint 0 Service Routine Register – FCSR0*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Function EP0 $1FDF | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX COMPLETE | HCSR0 |
| Function EP0 $1FDD | – | – | – | – | STALL SENT | RX SETUP | RX OUT PACKET | TX COMPLETE | FCSR0 |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..4 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 3 – STALL SENT**

The USB hardware sets this bit after a STALL has been sent to the host. The firmware uses this bit when responding to a Get Status[Endpoint] request. It is a read only bit and that is cleared indirectly by writing a one to the STALL_SENT_ACK bit of the Control and Acknowledge Register.

- **Bit 2 – RX SETUP: Setup Packet Received**

This bit is used by control endpoints only to signal to the microcontroller that the USB hardware has received a valid SETUP packet and that the data portion of the packet is stored in the FIFO. The hardware will clear all other bits in this register while setting RX SETUP. If interrupt is enabled, the microcontroller will be interrupted when RX SETUP is set. After the completion of reading the data from the FIFO, firmware should clear this bit by writing a one to the RX_SETUP_ACK bit of the Control and Acknowledge Register.

- **Bit 1 – RX OUT PACKET**

The USB hardware sets this bit after it has stored the data of an OUT transaction in the FIFO. While this bit is set, the hardware will NAK all OUT tokens. The USB hardware will not overwrite the data in the FIFO except for an early set-up. RX OUT Packet is used for the following operations:

1. Control write transactions by a control endpoint.
2. OUT transaction with DATA1 PID to complete the status phase of a control endpoint.

Setting this bit causes an interrupt to the microcontroller if the interrupt is enabled. FW clears this bit after the FIFO contents have been read by writing a one to the RX_OUT_PACKET_ACK bit of the Control and Acknowledge Register.

- **Bit 0 – TX COMPL: Transmit Completed**

This bit is used by a control endpoint hardware to signal to the microcontroller that it has successfully completed certain transactions. TX Complete is set at the completion of a:

1. Control read data stage.
2. Status stage without data stage.
3. Status stage after a control write transaction.

This bit is read only and is cleared indirectly by writing a one to the TX_COMPLETE_ACK bit of the Control and Acknowledge Register.

### *Hub Endpoint 0 Control and Acknowledge Register – HCAR0*

### *Function Endpoint 0 Control and Acknowledge Register – FCAR0*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Hub EP0 $1FA7 | DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_ SENT_ ACK | RX_ SETUP_ ACK | RX_OUT_ PACKET_ ACK | TX_ COMPLETE_ ACK | HCAR0 |
| Function EP0 $1FDD | DIR | DATA END | FORCE STALL | TX PACKET READY | STALL_ SENT_ ACK | RX_ SETUP_ ACK | RX_OUT_ PACKET_ ACK | TX_ COMPLETE_ ACK | FCAR0 |
| 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 – DIR: Control transfer direction**

It is set by the microcontroller firmware to indicate the direction of a control transfer to the USB hardware. The FW writes to this bit location after it receives an RX SETUP interrupt. The hardware uses this bit to determine the status phase of a control transfer.

0 = control write or no data stage

1 = control read

- **Bit 6 – DATA END**

When set to 1 by firmware, this bit indicate that the microcontroller has either placed the last data packet in FIFO, or that the microcontroller has processed the last data packet it expects from the Host. This bit is used by control endpoints only together with bit 4 (TX Packet Ready) to signal the USB hardware to go to the STATUS phase after the packet currently residing in the FIFO is transmitted. After the hardware completes the STATUS phase it will interrupt the microcontroller without clearing this bit.

- **Bit 5 – FORCE STALL**

This bit is set by the microcontroller to indicate a stalled endpoint. The hardware will send a STALL handshake as a response to the next IN or OUT token, or whenever there is a control transfer without a Data Stage.

The microcontroller sets this bit if it wants to force a STALL. A STALL is sent if any of the following condition is encountered:

1. An unsupported request is received.
2. The host continues to ask for data after the data is exhausted.
3. The control transfer has no data stage.

- **Bit 4 – TX PACKET READY: Transmit Packet Ready**

When set by the firmware, this bit indicates that the microcontroller has loaded the FIFO with a packet of data. This bit is cleared by the hardware after the USB Host acknowledges the packet. For ISO endpoints, this bit is cleared unconditionally after the data is sent.

This bit is used for the following operations:

1. Control read transactions by a control endpoint.
2. IN transactions with DATA1 PID to complete the status phase for a control endpoint, when this bit is zero but Data End set high (bit 4).
3. By a BULK IN or ISO IN or INT IN endpoint.

The microcontroller should write into the FIFO only if this bit is cleared. After it has completed writing the data, it should set this bit. This data can be of zero length.

Hardware clears this bit after it receives an ACK. If the interrupt is enabled and if the TX Complete bit is set, clearing the TX Packet Ready bit by the hardware causes an interrupt to the microcontroller.

- **Bit 3 – STALL_SENT_ACK: Acknowledge Stall Sent Interrupt**

Firmware sets this bit to clear STALL SENT, CSR bit 3. The 1 written in the CSRACK3 bit is not actually stored and thus does not have to be cleared.

- **Bit 2 – RX_SETUP_ACK: Acknowledge RX SETUP Interrupt**

Firmware sets this bit to clear RX SETUP, CSR bit2. The 1 written in the CSRACK2 bit is not actually stored and thus does not have to be cleared.

- **Bit 1 – RX_OUT_PACKET_ACK: Acknowledge RX OUT PACKET Interrupt**

Firmware sets this bit to clear RX OUT PACKET, CSR bit1. The 1 written in the CSRACK1 bit is not actually stored and thus does not have to be cleared.

- **Bit 0 – TX_COMPLETE_ACK: Acknowledge TX COMPLETE Interrupt**

Firmware sets this bit to clear TX COMPLETE, CSR bit0. The 1 written in the CSRACK0 bit is not actually stored and thus does not have to be cleared.

### *Function Endpoint 1, 2 Service Routine Register – FCSR1, 2*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Function EP1 $1FDC | – | – | – | – | STALL SENT | – | RX OUT PACKET | TX COMPLETE | FCSR1 |
| Function EP2 $1FDB | – | – | – | – | STALL SENT | – | RX OUT PACKET | TX COMPLETE | FCSR2 |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..4 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 3 – STALL SENT**

The USB hardware sets this bit after a STALL has been sent to the host. The firmware uses this bit when responding to a Get Status[Endpoint] request. It is a read only bit and that is cleared indirectly by writing a one to the STALL_SENT_ACK bit of the Control and Acknowledge Register.

- **Bit 2 – Reserved**

This bit is reserved in the AT43USB320A and will read as zero.

- **Bit 1 – RX OUT PACKET**

The USB hardware sets this bit after it has stored the data of an OUT transaction in the FIFO. While this bit is set, the hardware will NAK all OUT tokens. The USB hardware will not overwrite the data in the FIFO except for an early set-up. RX OUT Packet is used by a BULK OUT or ISO OUT or INT OUT endpoint.

Setting this bit causes an interrupt to the microcontroller if the interrupt is enabled. FW clears this bit after the FIFO contents have been read by writing a one to the RX_SETUP_ACK bit of the Control and Acknowledge Register.

- **Bit 0 – TX COMPLETE: Transmit Completed**

This bit is used by the endpoint hardware to signal to the microcontroller that the IN transaction was completed successfully. This bit is read only and is cleared indirectly by writing a one to the TX_COMPLETE_ACK bit of the Control and Acknowledge Register.

### *Function Endpoint 1, 2 Control and Acknowledge Register – FCAR1, 2*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Function EP1 $1FA4 | – | DATA END | FORCE STALL | TX PACKET RDY | STALL_SENT-ACK | – | RX_OUT_PACKET_ACK | TX_COMPLETE_ACK | FCAR1 |
| Function EP2 $1FA3 | – | DATA END | FORCE STALL | TX PACKET RDY | STALL_SENT-ACK | – | RX_OUT_PACKET_ACK | TX_COMPLETE-ACK | FCAR2 |
| 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 7 – Reserved**

This bit is reserved in the AT43USB320A and will read as zero.

- **Bit 6 – DATA END**

When set to 1 by firmware, this bit indicate that the microcontroller has either placed the last data packet in FIFO, or that the microcontroller has processed the last data packet it expects from the Host.

- **Bit 5 – FORCE STALL**

This bit is set by the microcontroller to indicate a stalled endpoint. The hardware will send a STALL handshake as a response to the next IN or OUT token. The microcontroller sets this bit if it wants to force a STALL. A STALL is send if the host continues to ask for data after the data is exhausted.

- **Bit 4 – TX PACKET RDY: Transmit Packet Ready**

When set by the firmware, this bit indicates that the microcontroller has loaded the FIFO with a packet of data. This bit is cleared by the hardware after the USB Host acknowledges the packet. For ISO endpoints, this bit is cleared unconditionally after the data is sent.

The microcontroller should write into the FIFO only if this bit is cleared. After it has completed writing the data, it should set this bit. This data can be of zero length.

The hardware clears this bit after it receives an ACK. If the interrupt is enabled and if the TX Complete bit is set, clearing the TX Packet Ready bit by the hardware causes an interrupt to the microcontroller.

- **Bit 3 – STALL_SENT_ACK: Acknowledge Stall Sent Interrupt**

Firmware sets this bit to clear STALL SENT, CSR bit 3. The 1 written in the CSRACK3 bit is not actually stored and thus does not have to be cleared.

- **Bit 2 – Reserved**

This bit is reserved in the AT43USB320A and will read as zero.

- **Bit 1 – RX_OUT_PACKET_ACK: Acknowledge RX OUT PACKET Interrupt**

Firmware sets this bit to clear RX OUT PACKET, CSR bit1. The 1 written in the CSRACK1 bit is not actually stored and thus does not have to be cleared.

- **Bit 0 – TX_COMPLETE_ACK: Acknowledge TX COMPLETE Interrupt**

Firmware sets this bit to clear TX COMPLETE, CSR bit0. The 1 written in the CSRACK0 bit is not actually stored and thus does not have to be cleared.

## USB Hub

The hub in a USB system provides for the electrical interface between USB devices and the host. The major functions that the hub must supports are:

- Connectivity
- Power management
- Device connect and disconnect
- Bus fault detection and recovery
- Full speed and low speed device support

A hub consists of two major components: a hub repeater and a hub controller. The hub repeater is responsible for:

- Providing upstream connectivity between the selected device and the Host
- Managing connectivity setup and tear-down
- Handling bus fault detection and recovery
- Detecting connect/disconnect on each port

The Hub Controller is responsible for:

- Hub enumeration
- Providing configuration information to the host
- Providing status of each port to the host
- Controlling each port per host command

The first two tasks of the hub are similar to that of a USB function and are described in detail in the following section. The descriptions will cover the features of the AT43USB320A's hub and how to program it to make a USB-compliant hub.

Control transactions for the hub control endpoint proceed exactly the same way as those described for the embedded function. The operation of the hub's endpoint 1 is fully implemented in the hardware and does not need any firmware support. Any status changes within the hub will automatically update hub endpoint 1, which will be sent to the host at the next IN token that is addressed to it. If no change has occurred, the interrupt endpoint will respond with a NAK.

**Hub General Registers** ***Global State Register – GLB_STATE***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FFB | – | – | – | SUSP FLG | RESUME FLG | RMWUPE | CONFG | HADD EN | GLB_STATE |
| Read/Write | R | R | R | R | R | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7...5 – Reserved Bits**

These bits are reserved in the AT43USB320A and will read as zeros.

- **Bit 4 – SUSP FLG: Suspend Flag**

This bit is set to 1 while the USB hardware is in the suspended state. This bit is a firmware read only bit. It is set and cleared by the USB hardware.

- **Bit 3 – RESUME FLGL Resume Flag**

When the USB hardware receives a resume signal from the upstream device it sets this bit. This bit will stay set until the USB hardware completes the downstream resume signaling. This bit is a firmware read only bit. It is set and cleared by the USB hardware.

- **Bit 2 – RMWUPE: Remote Wakeup Enable**

This bit is set if the host enables the hub's remote wakeup feature.

- **Bit 1 – CONFG: Configured**

This bit is set by firmware after a valid SET_CONFIGURATION request is received. It is cleared by a reset or by a SET_CONFIGURATION with a value of 0.

- **Bit 0 – HADD EN: Hub Address Enabled**

This bit is set by firmware after the status phase of a SET_ADDRESS request transaction so the hub will use the new address starting at the next transaction.

**Hub Status Register**

In the AT43USB320A overcurrent detection and port power switch control output processing is done in firmware. The hardware is designed so that various types of hubs are possible just through firmware modifications.

1. Hub local power status, bits 0 and 2, are optional features and apply to hubs that report on a global basis. If this feature is not used, both these bits should be programmed to 0. To use this feature, the firmware needs to know the status of the local power supply, which requires an input pin and extra internal or external circuitry.
2. Hub overcurrent status, bits 1 and 3, apply to self powered hubs with bus powered SIE only, or hubs that are programmable as self/bus powered. The firmware should clear these two bits to 0.

The firmware uses bits 1 and 3 to generate bit 0 of the Hub and Port Status Change Bitmap which is transmitted through the Hub Endpoint1 Data Register. Bit 0 of this register is a 1 whenever bit 1 or 3 of HSTATR is a 1.

***Hub Status Register – HSTR***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FC7 | – | – | – | – | OVLSC | LPSC | OVI | LPS | HSTR |
| Read/Write | R | R | R | R | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7..4 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 3 – OVLSC: Overcurrent Status Change**

0 = No change has occurred on Overcurrent Indicator

1 = Overcurrent Indicator has changed

- **Bit 2 – LPSC: Hub Local Power Status Change**

0 = No change has occurred on Local Power Status

1 = Local Power Status has changed

- **Bit 1 – OVI: Overcurrent Indicator**

0 = All power operations normal

1 = An overcurrent exist on a hub wide basis

- **Bit 0 – LPS: Hub Local Power Status**

0 = Local power supply is good

1 = Local power supply is lost (inactive)

### *Hub Port Control Register – HPCON*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| $1FC5 | – | HPCON2 | HPCON1 | HPCON0 | – | HPADD2 | HPADD1 | HPADD0 | HPCON |
| Read/Write | R | R/W | R/W | R/W | R | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – Reserved**

This bits is reserved in the AT43USB320A and will read as zero.

- **Bit 6..4 – HPCON2..0: Hub Port Control Command**

These bits are written by firmware to control the port states upon receipt of a Host request.

| Bit6 | Bit5 | Bit4 | Action |
|---|---|---|---|
| 0 | 0 | 0 | Disable port |
| 0 | 0 | 1 | Enable port |
| 0 | 1 | 0 | Reset and enable port |
| 0 | 1 | 1 | Suspend port |
| 1 | 0 | 0 | Resume port |

**Disable Port** = ClearPortFeature(PORT_ENABLE)

**Action:** USB hardware places addressed port in disabled state. Port 1 is placed in disabled state by firmware.

**Enable Port** = SetPort Feature(PORT_ENABLE)

**Action:** USB hardware places addressed port in enabled state. Firmware is responsible for placing Port 1 in enabled state.

**Reset and Enable Port** = SetPort Feature(PORT_RESET)

**Action:** USB hardware drives reset signaling through addressed port. USB hardware and firmware resets their embedded function registers to the default state.

**Suspend Port** = SetPortFeature(PORT_SUSPEND)

**Action:** USB hardware places port in idle state and stops propagating traffic through the addressed port. Firmware places Port 1 in suspend state by disabling its endpoints and placing the peripheral function in its low power state.

**Resume Port** = ClearPortFeature(PORT_SUSPEND)

**Action:** USB hardware sends resume signaling to addressed port and then enables port. Firmware takes the embedded function out of the suspend state and enables Port 1's endpoints.

- **Bit 3 – Reserved**

This bits is reserved in the AT43USB320A and will read as zero.

- **Bit 2..0 – HPCON2..0: Hub Port Address**

These bits define which port is being addressed for the command defined by bits [2:0].

| Bit2 | Bit1 | Bit0 | Port addresses |
|---|---|---|---|
| 1 | 0 | 1 | Port 5 |
| 1 | 0 | 0 | Port 4 |
| 0 | 1 | 1 | Port 3 |
| 0 | 1 | 0 | Port 2 |
| 0 | 0 | 1 | Port 1 |

**Selective Suspend and Resume**

The host can selectively suspend and resume a port through the Set Port Feature (PORT_SUSPEND) and Clear Port Feature (PORT_SUSPEND).

A port enters the suspend state after the microcontroller interprets the suspend request and sets the appropriate bits of the Hub Port Control Register, HPCON. From this point on he hub repeater hardware is responsible for proper actions in placing Ports [1:4] in the suspend mode. For Port 5, the embedded function port, the hardware will stop responding to any normal bus traffic, but the microcontroller firmware must place all external circuitry associated with the function in the low-power state.

A port exits from the suspend state when the hub receives a Clear Port Feature (PORT_SUSPEND) or Set Port Feature (PORT_RESET). If the Clear Port Feature (PORT_SUSPEND) is directed towards Ports [1:4], the USB hardware drives a "K" downstream for at least 20 ms followed by a low speed EOP. It then places the port in the enabled state. A Clear Port Feature (PORT_SUSPEND) to Port 1 (the embedded function) causes the firmware to wait 20 ms, take the embedded function out of the suspended state and then enable the port.

The ports can also exit from the suspended state through a remote wakeup if this feature is enabled. For Ports [1:4], this means detection of a connect/disconnect or an upstream directed J to K signaling. Remote wakeup for the embedded function is initiated through an external interrupt at INT0.

**Hub Port Status Register**

The bits in this register are used by the microcontroller firmware when reporting a port's status through the Port Status Field, *wPortStatus*. Bits 3 (POCI) and 5 (PPSTAT) are used by the USB hardware and are the only two bits that the firmware should set or clear. All other bits should not be modified by the firmware.

***Hub Port Status Register – HPSTAT1:5***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Port1 $1FB8 | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT | HPSTAT1 |
| Port2 $1FB9 | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT | HPSTAT2 |
| Port3 $1FBA | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT | HPSTAT3 |
| Port4 $1FBB | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT | HPSTAT4 |
| Port5 $1FBC | – | LSP | PPSTAT | PRSTAT | POCI | PSSTAT | PESTAT | PCSTAT | HPSTAT5 |
| Read/Write | R | R | R/W | R | R/W | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

- **Bit 7 – Reserved**

This bit is reserved in the AT43USB320A and will read as zero.

- **Bit 6 – LSP: Low-speed Device Attached**

0 = Full-speed device attached to this port

1 = Slow-speed device attached to this port

Set to 0 for Port 1 (full-speed only). Set and cleared by the hardware upon detection of device at EOF2.

- **Bit 5 – PPSTAT: Port Power Status**

0 = Port is powered OFF

1 = Port is powered ON

Set to 1 for Port 1. Set and cleared based on present status of port power.

- **Bit 4 – PRSTAT: Port Reset Status**

0 = Reset signaling not asserted

1 = Reset signaling asserted

Set and cleared by the hardware as a result of initiating a port reset by Port Control Register.

- **Bit 3 – POCI: Port Overcurrent Indicator**

0 = Power normal

1 = Overcurrent exist on port

Set to 0 for Port 1. Set and cleared by firmware upon detection of an overcurrent or removal of an overcurrent.

- **Bit 2 – PSSTAT: Port Suspend Status**

0 = Port not suspended

1 = Port suspended

Set and cleared by the hardware as controlled through Port Control Register.

- **Bit 1 – PESTAT: Port Enable Status**

0 = Port is disabled

1 = Port is enabled

Set and cleared by the hardware as controlled through Port Control register.

- **Bit 0 – PCSTAT: Port Connect Status**

0 = No device on this port

1 = Device present on this port

Set to 1 for Port 1. Set and cleared by the hardware after sampling of connect status at EOF2.

***Hub Port State Register – HPSTAT1:5***

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Port1 $1FA8 | – | – | – | – | – | – | **DPSTATE** | **DMSTATE** | **PSTATE1** |
| Port2 $1FA9 | – | – | – | – | – | – | **DPSTATE** | **DMSTATE** | **PSTATE2** |
| Port3 $1FAA | – | – | – | – | – | – | **DPSTATE** | **DMSTATE** | **PSTATE3** |
| Port4 $1FAB | – | – | – | – | – | – | **DPSTATE** | **DMSTATE** | **PSTATE4** |
| Port5 $1FAC | – | – | – | – | – | – | **DPSTATE** | **DMSTATE** | **PSTATE5** |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

These registers contain the state of the ports' DP and DM pins, which will be sent to the host upon receipt of a GetBusState request.

- **Bit 7..2 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 1 – DPSTATE: DPlus State**

Value of DP at last EOF. Set and cleared by hardware at EOF2.

Set to 1 for Port 1.

- **Bit 0 – DMSTATE: DMinus State**

Value of DM at last EOF. Set and cleared by hardware at EOF2.

Set to 0 for Port 1.

### *Hub Port Status Change Register – PSCR1..3*

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| Port1 $1FB0 | – | – | – | **RSTSC** | **POCIC** | **PSSC** | **PESC** | **PCSC** | **PSCR1** |
| Port2 $1FB1 | – | – | – | **RSTSC** | **POCIC** | **PSSC** | **PESC** | **PCSC** | **PSCR2** |
| Port3 $1FB2 | – | – | – | **RSTSC** | **POCIC** | **PSSC** | **PESC** | **PCSC** | **PSCR3** |
| Port4 $1FB3 | – | – | – | **RSTSC** | **POCIC** | **PSSC** | **PESC** | **PCSC** | **PSCR4** |
| Port5 $1FB4 | – | – | – | **RSTSC** | **POCIC** | **PSSC** | **PESC** | **PCSC** | **PSCR5** |
| Read/Write | R | R | R | R | R | R | R | R | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The microcontroller firmware uses the bits in this register to monitor when a port status change has occurred, which then gets reported to the host through the Port Change Field *wPortChange*.

Except for bit 3, the Port Overcurrent Indicator Change, the bits in this register are set by the USB hardware. Otherwise, the firmware should only clear these bits.

- **Bit 7..5 – Reserved**

These bits are reserved in the AT43USB320A and will read as zero.

- **Bit 4 – RSTSC: Port Reset Status Change**

0 = No change

1 = Reset complete

This bit is set by the USB hardware after it completes RESET signaling which is initiated when the Reset and Enable Port command is detected at the Port Control Register, HPCON. The firmware sends this command when it decodes a SetPortFeature(PORT_RESET) request from the host.

At EOF2 after the hardware completes the port reset, the hardware sets the Port Enable Status bit and clears the Port Reset Status bit of the Hub Port Status Register, HPSTAT. Cleared by firmware, ClearPortFeature(PORT_RESET).

- **Bit 3 – POCIC: Port Overcurrent Indicator Change**

0 = No change has occurred on Overcurrent Indicator

1 = Overcurrent Indicator has changed

This bit is relevant to hubs with individual overcurrent reporting only. The firmware sets this bit as a result of detecting overcurrent at the ports OVC# pin. The firmware clears bit through ClearPortFeature(PORT_OVER_CURRENT). For Port 1, this bit is always cleared.

- **Bit 2 – PSSC: Port Suspend Status Change**

0 = No change

1 = Resume completed

Port 1:4 is set by hardware upon completion of firmware initiated resume process. Port 5 is set by firmware 20 ms after the next EOF2 after completion of resume process. RESUME signaling is initiated through global resume, selective resume and remote wakeup. Cleared by firmware via host request ClearPortFeature(PORT_SUSPEND).

- **Bit 1 – PESC: Port Enable/Disable Status Change**

0 = No change has occurred on Port Enable/Disable Status

1 = Port Enable/Disable status has changed

Set by hardware due to babble, physical disconnect or overcurrent except for Port 5 in which case it is set by hardware at EOF2 due to hardware events. Cleared by firmware via Host request ClearPortFeature(PORT_ENABLE).

- **Bit 0 – PCSC: Port Connect Status Change**

0 = No change has occurred on Current Connect Status

1 = Current Connect Status has changed

This bit is set by hardware at EOF2 after it detects a connect or disconnect at a port, except for Port 5. Hardware sets this bit for Port 5 after a hub reset. Cleared by firmware via Host request ClearPortFeature(PORT_CONNECTION).

**Hub and Port Power Management**

For the utmost flexibility, the USB hardware of the AT43USB320A is designed to accommodate hubs of various capacitance. Management of the downstream port power is also defined by the firmware: per port or global overcurrent sensing, individual or gang power switching. While the interface to the external power supply monitoring and switching is achieved through the microcontroller's GPIO pins, the USB hardware of the AT43USB320A contains the circuitry to handle all the possible combinations port power management tasks.

**Overcurrent sensing**

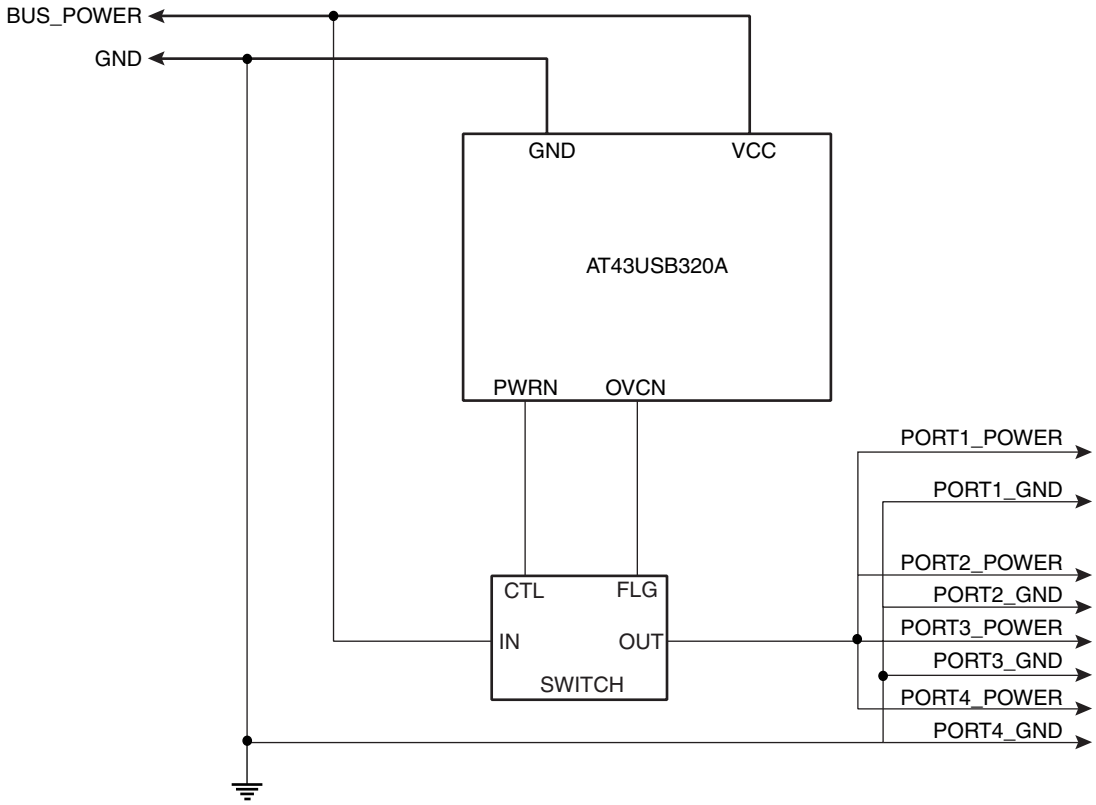
1. **Global Overcurrent Protection** – In this mode, the Port Overcurrent Indicator and Port Overcurrent Indicator Change should be set to 0's. For the AT43USB320A an external solid state switch, such as the Micrel MIC2545-2, is required to switch power to the external USB ports. The FLG# output of the switch should be connected to PD0. When an overcurrent occurs, FLG is asserted and the firmware should set the Hub Overcurrent Indicator and Hub Overcurrent Indicator Change and switch off power to the hub.
2. **Individual Port Over-current Protection** – The Hub Overcurrent Indicator and Hub Overcurrent Indicator Change bits should be set to 0's. One MIC2026-2 is required for each two USB ports. Each of the FLG# outputs of the MIC2026-2 should be connected to an unused microcontroller port. An overcurrent is indicated by assertion of FLG#. The firmware sets the corresponding port's Overcurrent Indicator and the Overcurrent Indicator Change bits and switches off power to the port. At the next IN token from the Host, the AT43USB320A reports the status change.

**Port Power Switching**

1. **Gang Power Switching** – One of the microcontroller I/O port pins must be programmed as an output to control the external switch, PWRN. Switch ON is requested by the USB Host through the SetPortFeature(PORT_POWER) request. Switch OFF is executed upon receipt of a ClearPortFeature(PORT_POWER) or upon detecting an overcurrent condition. The firmware clears the Power Control Bit. Only if all of the Power Control Bits of ports 1 through 4 are cleared should the firmware de-assert the PWRN pin.
2. **Individual Power Switching** – One microcontroller I/O port pin must be assigned for each USB port to control the external switch, PWRxN, where x = 1, 2, 3, 4. Each of the Power Control Bits controls one PWRxN.
3. **Multiple Ganged Overcurrent Protection** – Overcurrent sensing is grouped physically into one or more gangs, but reported individually.

Figure 22 shows a simplified diagram of a power management circuit of an AT43USB320A based hub design with global overcurrent protection and ganged power switching.

**Figure 22.** Port Power Management




## Suspend and Resume

The AT43USB320A enters suspend only when requested by the USB host through bus inactivity for at least 3 ms. The USB hardware would detect this request, sets the GLB_SUSP bit of SPRSR, Suspend/Resume Register, and interrupts the microcontroller if the interrupt is enabled. The microcontroller should shut down any peripheral activity and enter the Power Down mode by setting the SE and SM bits of MCUCR and then executes the SLEEP instruction. The USB hardware shuts off the oscillator and PLL.

### Global Resume

Global resume is signaled by a J to K state change on Port0. The USB hardware enables the oscillator/PLL, propagates the RESUME signaling, and sets the RSM bit of the SPRSR, which generates an interrupt. The microcontroller starts executing where it left off and services the interrupt. As part of the ISR, the firmware clears the GLB_SUSP bit.

### Remote Wakeup

While the AT43USB320A is in global suspend, resume signaling is also possible through remote wakeup if the remote wakeup feature is enabled. Remote wakeup is defined as a port connect, port disconnect or resume signaling received at a downstream port or, in case of the embedded function, through an external interrupt.

A remote wakeup initiated at a downstream port is similar in many respects to a global resume. The USB hardware enables the oscillator/PLL, propagates the RESUME signaling, and sets the RSM bit of the SPRSR which generates an interrupt. The microcontroller starts executing where it left off and services the interrupt. As part of the ISR, the firmware clears the GLB_SUSP bit.

A remote wakeup from the embedded function is initiated through INT0 or the external interrupt, INT1, which enables the oscillator/PLL and the USB hardware. The USB hardware drives RESUME signaling and sets the FRMWUP and RSM bits of SPRSR which generates an inter-

rupt to the microcontroller. The microcontroller starts executing where it left off and services the interrupt. As part of the ISR, the firmware clears the GLB SUSP bit.

At completion of RESUME signaling, the USB hardware sets the Port Suspend Status Change bits of the Hub Port Status Change Registers.

**Selective Suspend and Resume**

See “Hub Port Control Register – HPCON” on page 89.

**Suspend and Resume Process**

*Global Suspend*

*The Host stops sending packets, the hardware detects this as global suspend signaling and stops all downstream signaling. Finally, the hardware asserts the GLB_SUSP interrupt.*

| Hardware | Firmware |
|---|---|
| | 1.Host stops sending packets |
| 2. Global suspend signaling detected | |
| 3. Stop downstream signaling | |
| 4. Set GBL SUS bit → interrupt | |
| | 5. Shut down any peripheral activity |
| | 6. Set Sleep Enable and Sleep Mode bits of MCUCR |
| | 7. Set GPIO to low power state if required |
| | 8. Set UOVCER bit 2 |
| | 9. Execute SLEEP instruction |
| 10. SLEEP bit detected | |
| 11. Shut off oscillator | |

*Global Resume*

*The Host resumes signaling, the hardware detects this as global resume and propagates this signaling to all downstream ports. Finally, the hardware enables the oscillator and asserts the RSM interrupt.*

| Hardware | Firmware |
|---|---|
| | 1.Host resumes signaling |
| 2. Resume signaling detected | |
| 3. Propagate signaling downstream | |
| 4. Enable oscillator | |
| 5. Set RSM bit → interrupt | |
| | 6. Reset RSM and GBL SUSP bits |
| | 7. Restore GPIO states if required |
| | 8. Clear UOVCER bit 2 |
| | 9. Enable peripheral activity |

*Remote Wake-up, Downstream Ports*

*The hardware detects a connect/disconnect/port resume and propagates resume signaling upstream. Finally, the hardware enables the oscillator and asserts the RSM interrupt.*

| Hardware | Firmware |
|---|---|
| 1. Connect/disconnect/port resume detected | |
| 2. Propagate resume signaling | |
| 3. Enable Oscillator | |
| 4. Set RSM bit → interrupt | |
| | 5. Reset RSM and GBL SUSP bits |
| | 6. Restore GPIO states if required |
| | 7. Clear UOVCER bit 2 |
| | 8. Enable peripheral activity |

*Remote Wake-up, Embedded Function*

*The hardware detects an INT0/INT1 and propagates resume signaling upstream. Finally, the hardware enables the oscillator and asserts the RSM and FRWUP interrupts.*

| Hardware | Firmware |
|---|---|
| | 1.External event activates INT0/INT1 |
| 2. Propagate resume signaling | |
| 3. Enable Oscillator | |
| 4. Set RSM and FRMWUP bits → interrupt | |
| | 5. Clear GLB SUSP, RSM, FRMWUP bits |
| | 6. Restore GPIO states if required |
| | 7. Clear UOVCER bit 2 |
| | 8. Enable peripheral activity |

*Selective Suspend, Downstream Ports*

| Hardware | Firmware |
|---|---|
| | 1. Set or Clear Port Feature PORT_SUSPEND decoded |
| | 2. Write HPCON[2:0] and HPADD[2:0] bits |
| 3. Suspend or resume port per command | |

*Selective Suspend, Embedded Function*

| Hardware | Firmware |
|---|---|
| | 1. Set Port Feature PORT_SUSPEND decoded<br>2. Disable Port 5's endpoints<br>3. Set GPIO to low power state if required |

*Selective Resume, Embedded Function*

| Hardware | Firmware |
|---|---|
| | 1. Clear Port Feature PORT_SUSPEND decoded<br>2. Clear Port 5 suspend status bit<br>3. Restore GPIO states if required<br>4. Wait 23 ms, then set enable status bit and suspend change bit<br>5. Enable Port 5 endpoints |
| 6. Send updated port status at next IN to endpoint1 | |

# Electrical Specification

## Absolute Maximum Ratings

Stresses beyond those listed below 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.

**Table 29.** Absolute Maximum Ratings

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC5}$ | 5V Power Supply | | | 5.5 | V |
| VI | DC input voltage | | -0.3V | VCEXT+0.3 4.6 max | V |
| VO | DC output voltage | | -0.3 | VCEXT+0.3 4.6 max | V |
| TO | Operating temperature | | -40 | +125 | °C |
| TS | Storage temperature | | -65 | +150 | °C |

## DC Characteristics

The values shown in this table are valid for TA = 0°C to 85°C, VCC = 4.4 to 5.25V, unless otherwise noted.

**Table 30.** Power Supply

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | 5V Power Supply | | 4.4 | 5.25 | V |
| $I_{CC}$ | 5V Supply Current | | | 40 | mA |
| $I_{CCS}$ | Suspended Device Current | | | 250 | uA |

**Table 31.** USB Signals: DPx, DMx

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{IH}$ | Input Level High (driven) | | 2.0 | | V |
| $V_{IHZ}$ | Input Level High (floating) | | 2.7 | | V |
| $V_{IL}$ | Input Level Low | | | 0.8 | V |
| $V_{DI}$ | Differential Input Sensitivity | DPx and DMx | 0.2 | | V |
| $V_{CM}$ | Differential Common Mode Range | | 0.8 | 2.5 | V |
| $V_{OL1}$ | Static Output Low | RL of 1.5 kΩ to 3.6V | | 0.3 | V |
| $V_{OH1}$ | Static Output High | RL of 15 kΩ to GND | 2.8 | 3.6 | V |
| $V_{CRS}$ | Output Signal Crossover | | 1.3 | 2.0 | V |
| $V_{IN}$ | Input Capacitance | | | 20 | pF |

**Table 32.** PA, PB, PC, PD

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{OL2}$ | Output Low Level | IOL = 4 mA | | 0.5 | V |
| $V_{OH2}$ | Output High Level | IOH = 4 mA | VCEXT - 0.4 | | V |
| $V_{IL2}$ | Input Low Level | | -0.3 | 0.3 VCEXT | V |
| $V_{IH2}$ | Input High Level | | 0.7 VCEXT | VCEXT + 0.3 | V |
| C | Input/Output capacitance | 1 MHz | | 10 | pF |

**Table 33.** Oscillator Signals: XTAL1, XTAL2

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{LH}$ | OSC1 switching level | | 0.47 | 1.20 | V |
| $V_{HL}$ | OSC1 switching level | | 0.67 | 1.44 | V |
| CX1 | Input capacitance, XTAL1 | | | 16 | pF |
| CX2 | Output capacitance, XTAL2 | | | 16 | pF |
| C12 | OSC1/2 capacitance | | | 8 | pF |
| $t_{SU}$ | Start-up time | 6 MHz, fundamental | | 2 | ms |
| DL | Drive level | | | 150 | µW |

Note: XTAL2 must not be used to drive other circuitry.

## AC Characteristics

**Table 34.** USB Driver Characteristics, Full Speed Operation

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| TR | Rise time | $C_L$ = 50 pF | 4 | 20 | ns |
| TF | Fall time | $C_L$ = 50 pF | 4 | 20 | ns |
| TRFM | TR/TF matching | | 90 | 110 | % |
| ZDRV | Driver output resistance(1) | Steady state drive | 28 | 44 | Ω |

Note: 1. With external 27Ω series resistor.

**Figure 23.** Full-speed Load

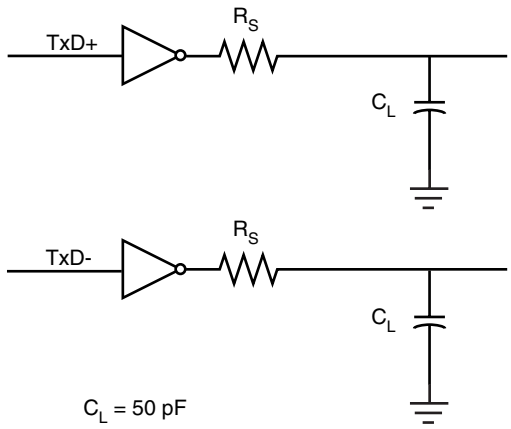



**Table 35.** USB Driver Characteristics, Low-speed Operation

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| TR | Rise time | CL = 200 - 600 pF | 75 | 300 | ns |
| TF | Fall time | CL = 200 - 600 pF | 75 | 300 | ns |
| TRFM | TR/TF matching | | 80 | 125 | % |

**Figure 24.** Low-speed Downstream Port Load

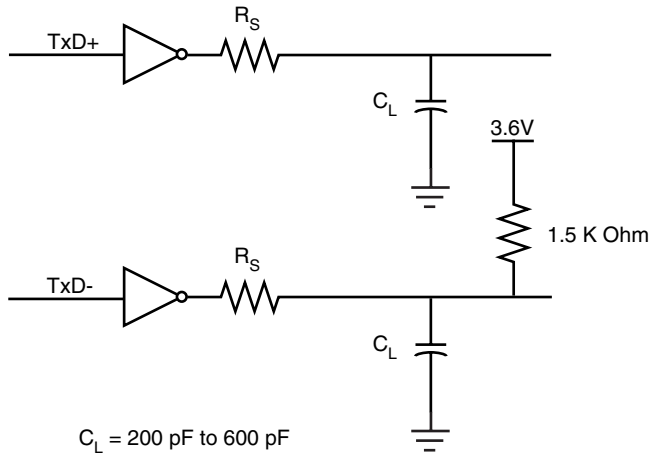



**Table 36.** USB Source Timings, Full-speed Operation

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| TDRATE | Full Speed Data Rate(1) | Average Bit Rate | 11.97 | 12.03 | Mb/s |
| TFRAME | Frame Interval(1) | | 0.9995 | 1.0005 | ms |
| TRFI | Consecutive Frame Interval Jitter(1) | No clock adjustment | | 42 | ns |
| TRFIADJ | Consecutive Frame Interval Jitter(1) | With clock adjustment | | 126 | ns |
| TDJ1<br>TDJ2 | Source Diff Driver Jitter<br>To Next Transition<br>For Paired Transitions | | -2<br>-1 | 2<br>1 | ns |
| TFDEOP | Source Jitter for Differential Transition to SEO Transitions | | -2 | 5 | ns |
| TDEOP | Differential to EOP Transition Skew | | -2 | 5 | ns |
| TJR1<br>TJR2 | Recvr Data Jitter Tolerance<br>To Next Transition<br>For Paired Transitions | | -18.5<br>-9 | 18.5<br>9 | ns |
| TFEOPT | Source SEO interval of EOP | | 160 | 175 | ns |
| TFEOPR | Receiver SEO interval of EOP | | 82 | | ns |
| TFST | Width of SEO interval during differential transition | | | 14 | ns |

Note: 1. With 6.000 MHz, 100 ppm crystal.

**Figure 25.** Differential Data Jitter

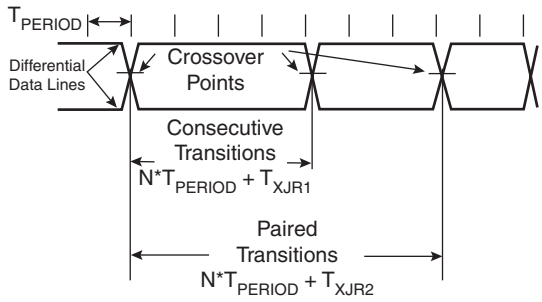



**Figure 26.** Differential-to-EOP Transition Skew and EOP Width

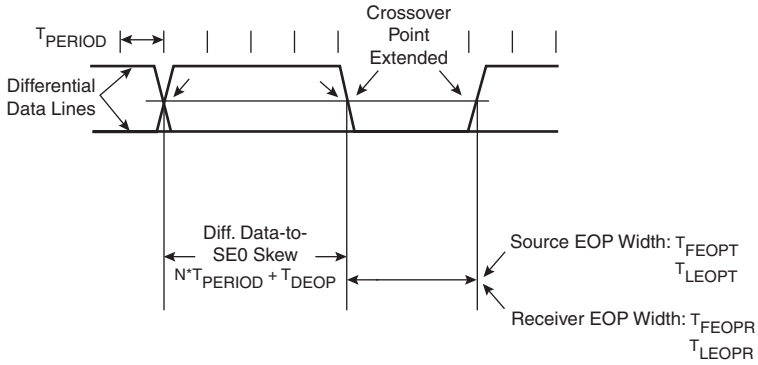



**Figure 27.** Receiver Jitter Tolerance

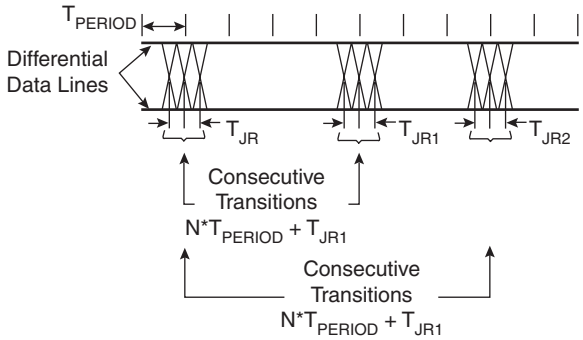


**Table 37.** Hub Timings, Full-speed Operation

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| THDD2 | Hub Differential Data Delay without cable | | | 44 | ns |
| THDJ1<br>THDJ2 | Hub Diff Driver Jitter to Next Transition for Paired Transitions | | -3<br>-1 | 3<br>1 | ns |
| TFSOP | Data Bit Width Distortion after SOP | | -5 | 5 | ns |
| TFEOPD | Hub EOP Delay Relative to THDD | | 0 | 15 | ns |
| TFHESK | Hub EOP Output Width Skew | | -15 | 15 | ns |

**Table 38.** Hub Timings, Low-speed Operation

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| TLHDD | Hub Differential Data Delay | | | 300 | ns |
| TLHDJ1<br>TLHDJ2<br>TLUHJ1<br>TLUHJ2 | Downstr Hub Diff Driver Jitter to Next Transition, downst for Paired Transitions, downst to Next Transition, upstr for Paired Transitions, upstr | | -45<br>-15<br>-45<br>-45 | 45<br>15<br>45<br>45 | ns |
| TSOP | Data Bit Width Distortion after SOP | | -60 | 60 | ns |
| TLEOPD | Hub EOP Delay Relative to THDD | | 0 | 200 | ns |
| TLHESK | Hub EOP Output Width Skew | | -300 | 300 | ns |

**Table 39.** Hub Event Timings

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| TDCNN | Time to detect a downstream port connect event | | 2.5 | 2000 | µs |
| TDDIS | Time to detect a disconnect event on downstream port<br>Awake Hub<br>Suspended Hub | | 2.5<br>2.5 | 2000<br>12000 | µs |
| TURSM | Time from detecting downstream resume to rebroadcast | | | 100 | µs |
| TDRST | Duration of driving reset to a downstream device | Only for a SetPortFeature (PORT_RESET) request | 10 | 20 | µs |
| TDSPDEV | Time to evaluate device speed after reset | | 2.5 | 1000 | µs |
| TURLK | Time to detect a long K from upstream | | 2.5 | 5.5 | µs |
| TURLSEO | Time to detect a long SEO from upstream | | 2.5 | 5.5 | µs |
| TURPSEO | Duration of repeating SEO upstream | | | 23 | FS bits |
| TUDEOP | Duration of sending SEO upstream after EOF1 | | | 2 | FS bits |

**Figure 28.** Hub Differential Delay, Differential Jitter and SOP Distortion

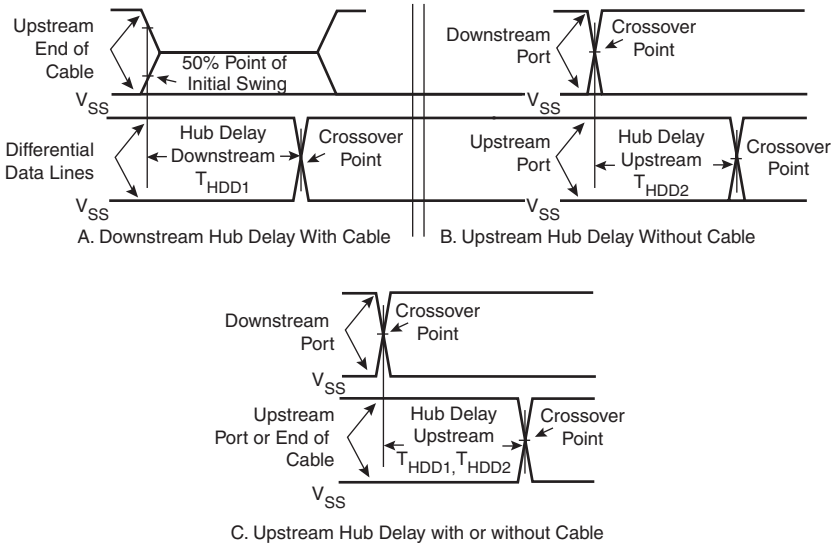



**Figure 29.** Hub EOP Delay and EOP Skew

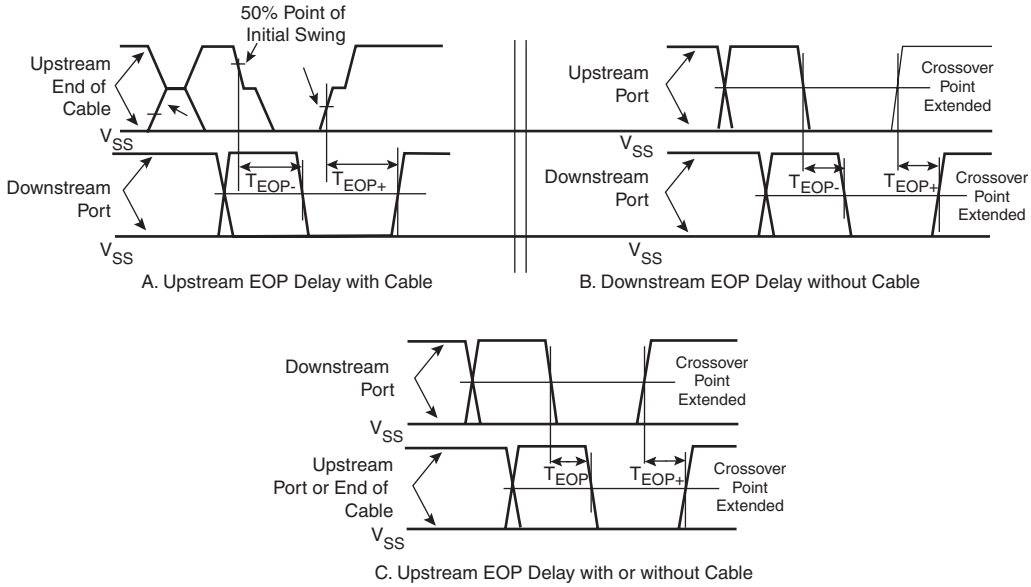
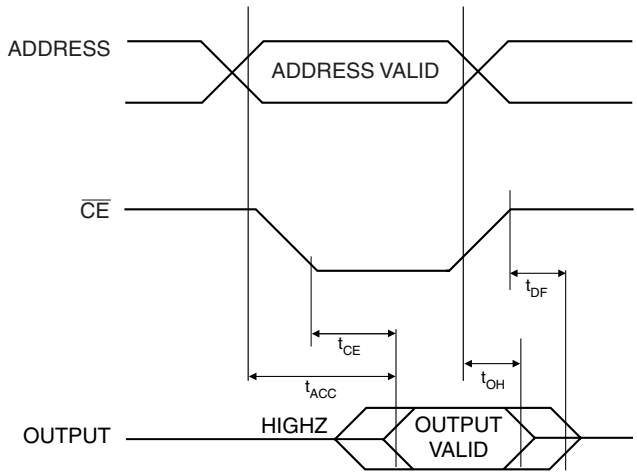


**Table 40.** External Program Memory Read Timing

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $t_{ACC}$ | Address to Output Delay | | | 55 | ns |
| $t_{CEN}$ | CEN to Output Delay | | | 55 | ns |
| $t_{DF}$ | CEN to Output Float | | 0 | | ns |
| $t_{OH}$ | Output Hold from CEN or Address, whichever occurred first | | 0 | | ns |

**Figure 30.** External Program Memory Read Timing Diagram

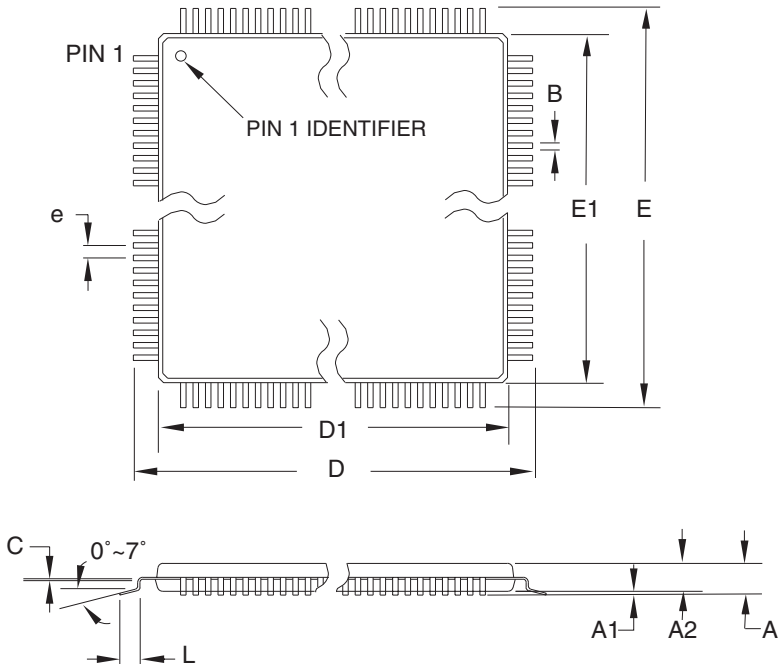


## Ordering Information

| Ordering Code | Package | Operation Range |
|---|---|---|
| AT43USB320A-AC | 100 LQFP | Commercial<br>(0°C to 70°C) |

## Packaging Information

### 100AA – LQFP




**COMMON DIMENSIONS**
(Unit of Measure = mm)

| SYMBOL | MIN | NOM | MAX | NOTE |
|---|---|---|---|---|
| A | – | – | 1.60 | |
| A1 | 0.05 | – | 0.15 | |
| A2 | 1.35 | 1.40 | 1.45 | |
| D | 15.75 | 16.00 | 16.25 | |
| D1 | 13.90 | 14.00 | 14.10 | Note 2 |
| E | 15.75 | 16.00 | 16.25 | |
| E1 | 13.90 | 14.00 | 14.10 | Note 2 |
| B | 0.17 | – | 0.27 | |
| C | 0.09 | – | 0.20 | |
| L | 0.45 | – | 0.75 | |
| e | 0.50 TYP | | | |

Notes:
1. This package conforms to JEDEC reference MS-026, Variation AED.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.08 mm maximum.

04/29/2002

| | TITLE | DRAWING NO. | REV. |
|---|---|---|---|
| 2325 Orchard Parkway<br>San Jose, CA 95131 | **100AA,** 100-lead, 14 x 14 mm Body Size, 1.4 mm Body Thickness, 0.5 mm Lead Pitch, Low Profile Quad Flat Pack (LQFP) | 100AA | C |

## Errata Sheet

Errata (All Date Codes): Missed Watchdog Timer Reset

## Problem

There is a synchronization problem between the watchdog clock and the AVR clock. Even though the clock inputs to both the watchdog timer and the AVR core are generated through the same crystal, the two clock sources are not going through the same PLL. The AVR is clocked at 12 MHz and the watchdog timer is clocked at 1MHz. The WDR (Watchdog Reset) instruction is a one-clock-cycle instruction. As such, when a watchdog timer reset occurs due to a WDR, the watchdog timer may miss the reset. This happens frequently if the AVR is clocked much faster than the watchdog timer.

## Fix/Workaround

A workaround is to invoke the WDR repetitively to ensure that the watchdog timer will be able to receive the reset signal. If the AVR runs at 12 MHz, the WDR command must be invoked fourteen times back to back.

The following is the sample code for resetting and arming the watchdog timer, assuming the AVR is running at 12 MHz:

```
asm ( "ldi r16,15\n WDR\n WDR\n WDR\n WDR\n WDR\n WDR\n WDR\n WDR\n
WDR\n WDR\n WDR\n WDR\n out 0x21,r16 " );
```

To disarm and disable the watchdog, do the following:

```
asm ( "ldi r16,0x18\nldi r17,0x10\n\n out 0x21,r16\n out 0x21,r17 " );
```

Please note that if the AVR runs at 24 MHz, the WDR should be invoked twenty-six times.

## Change Log

| Doc. Rev. | Comments |
|---|---|
| 1443E | • **Data Correction:** timeout period data in Table 19 on page 51.<br>• **Information Change:** UART does not support a 9-bit data mode. Changes were made to “Data Reception” on page 59, to Figure 19 on page 59 and Figure 20 on page 60. The “UART Control Register – UCR” on page 62.<br>• **Additions:** Added an “Errata Sheet” on page 111 and a “Change Log” on page 112. |

## Table of Contents

## Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

## Regional Headquarters

*Europe*
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

*Asia*
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

*Japan*
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

## Atmel Operations

*Memory*
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

*Microcontrollers*
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

*ASIC/ASSP/Smart Cards*
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

*RF/Automotive*
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

*Biometrics/Imaging/Hi-Rel MPU/*
*High Speed Converters/RF Datacom*
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80


*Literature Requests*
www.atmel.com/literature




1443E–USB–4/04 /xM