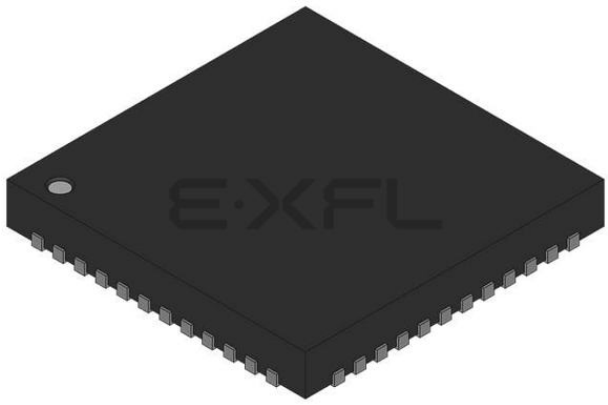


**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | AVR |
| Core Size | 32-Bit Single-Core |
| Speed | 50MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, DMA, POR, PWM, WDT |
| Number of I/O | 36 |
| Program Memory Size | 32KB (32K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 16K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.62V ~ 3.6V |
| Data Converters | A/D 8x12b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 48-VFQFN Exposed Pad |
| Supplier Device Package | 48-QFN (7x7) |
| Purchase URL | https://www.e-xfl.com/product-detail/atmel/at32uc3l032-zaut |

through a dedicated Peripheral Bus address. Some of the general-purpose fuse bits are reserved for special purposes, and should not be used for other functions:

**Table 8-2.** General-purpose Fuses with Special Functions

| General-Purpose fuse number | Name | Usage |
|---|---|---|
| 15:0 | LOCK | Region lock bits. |
| 16 | EPFL | External Privileged Fetch Lock. Used to prevent the CPU from fetching instructions from external memories when in privileged mode. This bit can only be changed when the security bit is cleared. The address range corresponding to external memories is device-specific, and not known to the Flash Controller. This fuse bit is simply routed out of the CPU or bus system, the Flash Controller does not treat this fuse in any special way, except that it can not be altered when the security bit is set.<br>If the security bit is set, only an external JTAG or aWire Chip Erase can clear EPFL. No internal commands can alter EPFL if the security bit is set.<br>When the fuse is erased (i.e. "1"), the CPU can execute instructions fetched from external memories. When the fuse is programmed (i.e. "0"), instructions can not be executed from external memories.<br>This fuse has no effect in devices with no External Memory Interface (EBI). |
| 19:17 | BOOTPROT | Used to select one of eight different bootloader sizes. Pages included in the bootloader area can not be erased or programmed except by a JTAG or aWire chip erase. BOOTPROT can only be changed when the security bit is cleared.<br>If the security bit is set, only an external JTAG or aWire Chip Erase can clear BOOTPROT, and thereby allow the pages protected by BOOTPROT to be programmed. No internal commands can alter BOOTPROT or the pages protected by BOOTPROT if the security bit is set. |
| 21:20 | SECURE | Used to configure secure state and secure state debug capabilities. Decoded into SSE and SSDE signals as shown in Table 8-5. Refer to the AVR32 Architecture Manual and the AVR32UC Technical Reference Manual for more details on SSE and SSDE. |
| 22 | UPROT | If programmed (i.e. “0”), the JTAG USER PROTECTION feature is enabled. If this fuse is programmed some HSB addresses will be accessible by JTAG access even if the flash security fuse is programmed. Refer to the JTAG documentation for more information on this functionality. This bit can only be changed when the security bit is cleared. |

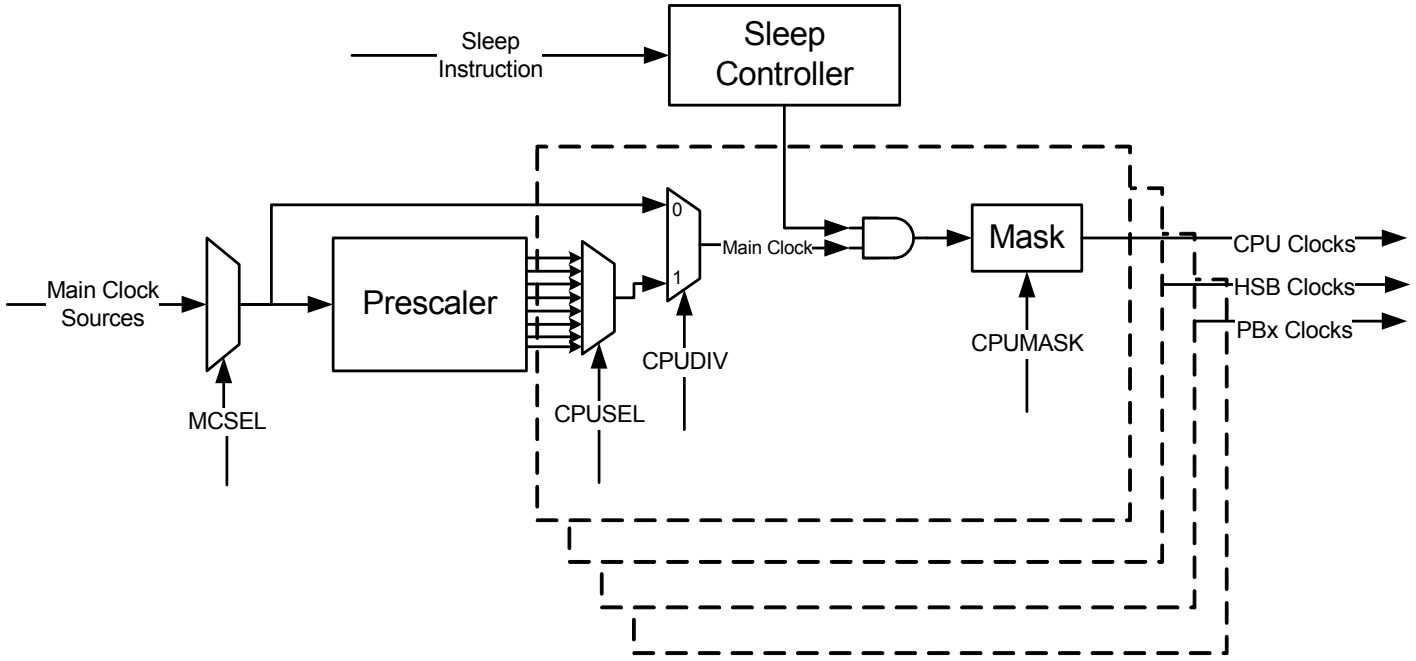
The BOOTPROT fuses protects the following address space for the Boot Loader:

## 12.6 Functional Description

### 12.6.1 Synchronous Clocks

The System RC Oscillator (RCSYS) and a selection of other clock sources can provide the source for the main clock, which is the origin for the synchronous CPU/HSB and PBx module clocks. For details about the other main clock sources, please refer to the Main Clock Control (MCCTRL) register description. The synchronous clocks can run of the main clock and all the 8-bit prescaler settings as long as $f_{CPU} \geq f_{PBx}$. The synchronous clock source can be changed on-the fly, according to variations in application load. The clock domains can be shut down in sleep mode, as described in Section 12.6.3. The module clocks in every synchronous clock domain can be individually masked to minimize power consumption in inactive modules.

**Figure 12-2.** Synchronous Clock Generation




#### 12.6.1.1 *Selecting the main clock source*

The common main clock can be connected to RCSYS or a selection of other clock sources. For details about the other main clock sources, please refer to the MCCTRL register description. By default, the main clock will be connected to RCSYS. The user can connect the main clock to another source by writing to the Main Clock Select (MCCTRL.MCSEL) field. The user must first assure that the source is enabled and ready in order to avoid a deadlock. Care should also be taken so that the new synchronous clock frequencies do not exceed the maximum frequency for each clock domain.

#### 12.6.1.2 *Selecting synchronous clock division ratio*

The main clock feeds an 8-bit prescaler, which can be used to generate the synchronous clocks. By default, the synchronous clocks run on the undivided main clock. The user can select a prescaler division for the CPU clock by writing a one to the CPU Division bit in the CPU Clock Select register (CPUSEL.CPUDIV), and a value to the CPU Clock Select field (CPUSEL.CPUSEL), resulting in a CPU clock frequency:

$$f_{CPU} = f_{main} / 2^{(CPUSEL+1)}$$

### 13.6.25 Oscillator 0 Version Register

**Name:** OSC0VERSION

**Access Type:** Read-only

**Offset:** 0x03C8

**Reset Value:** -

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | VARIANT | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | VERSION[11:8] | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VERSION[7:0] | | | | | | | |

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

### 16.7.2 Interrupt Disable Register

**Name:** IDR

**Access Type:** Write-only

**Offset:** 0x004

**Reset Value:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | INT30 | INT29 | INT28 | INT27 | INT26 | INT25 | INT24 |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| INT23 | INT22 | INT21 | INT20 | INT19 | INT18 | INT17 | INT16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| INT15 | INT14 | INT13 | INT12 | INT11 | INT10 | INT9 | INT8 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INT7 | INT6 | INT5 | INT4 | INT3 | INT2 | INT1 | NMI |

- **INTn: External Interrupt n**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.
  Please refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  Writing a zero to this bit has no effect.
  Writing a one to this bit will clear the corresponding bit in IMR.

### 16.7.3 Interrupt Mask Register

**Name:** IMR

**Access Type:** Read-only

**Offset:** 0x008

**Reset Value:** 0x00000000

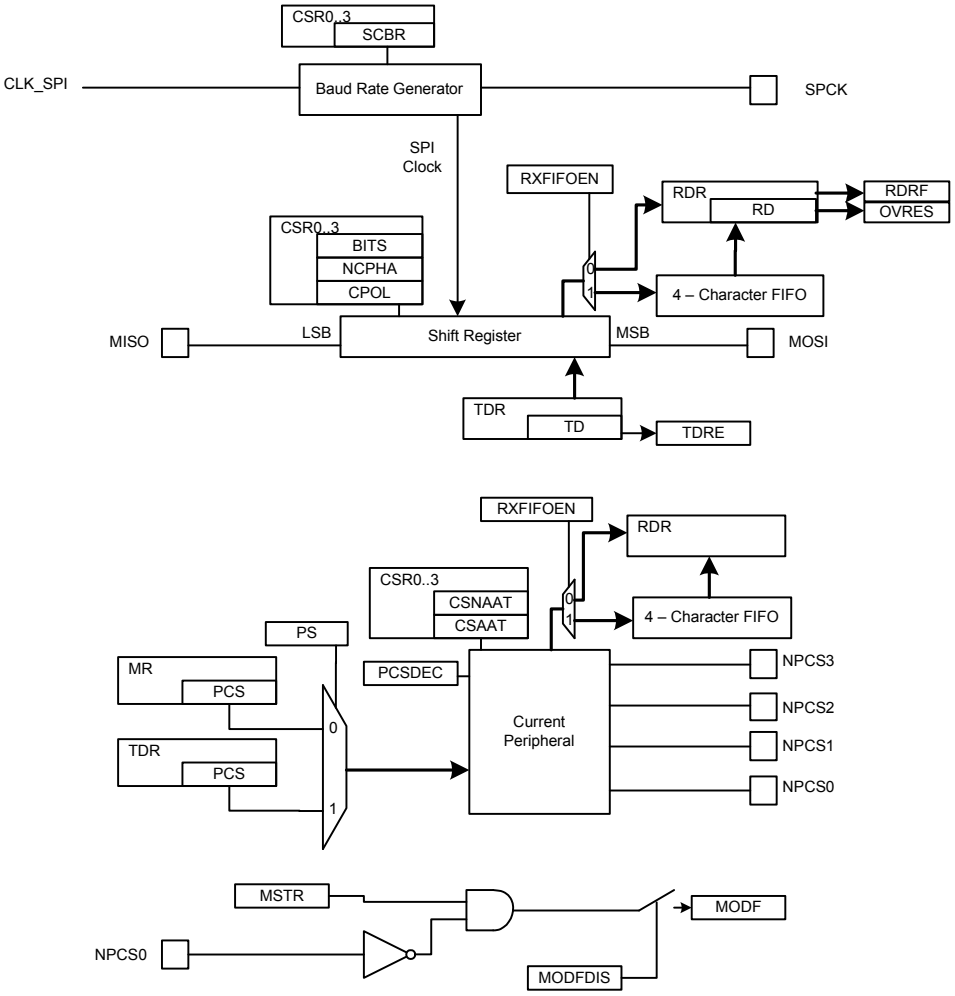
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | INT30 | INT29 | INT28 | INT27 | INT26 | INT25 | INT24 |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| INT23 | INT22 | INT21 | INT20 | INT19 | INT18 | INT17 | INT16 |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| INT15 | INT14 | INT13 | INT12 | INT11 | INT10 | INT9 | INT8 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INT7 | INT6 | INT5 | INT4 | INT3 | INT2 | INT1 | NMI |

- **INTn: External Interrupt n**
  0: The corresponding interrupt is disabled.
  1: The corresponding interrupt is enabled.
  This bit is cleared when the corresponding bit in IDR is written to one.
  This bit is set when the corresponding bit in IER is written to one.
  Please refer to the Module Configuration section for the number of external interrupts.
- **NMI: Non-Maskable Interrupt**
  0: The Non-Maskable Interrupt is disabled.
  1: The Non-Maskable Interrupt is enabled.
  This bit is cleared when the corresponding bit in IDR is written to one.
  This bit is set when the corresponding bit in IER is written to one.

Figure 20-5 on page 419shows a block diagram of the SPI when operating in master mode. Figure 20-6 on page 420 shows a flow chart describing how transfers are handled.

### *20.7.3.1 Master mode block diagram*

**Figure 20-5.** Master Mode Block Diagram



### 20.8.1 Control Register

**Name:** CR

**Access Type:** Write-only

**Offset:** 0x00

**Reset Value:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | LASTXFER |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | - | - | - | - | - |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | FLUSHFIFO |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SWRST | - | - | - | - | - | SPIDIS | SPIEN |

- **LASTXFER: Last Transfer**
  1: The current NPCS will be deasserted after the character written in TD has been transferred. When CSRn.CSAAT is one, this allows to close the communication with the current serial peripheral by raising the corresponding NPCS line as soon as TD transfer has completed.
  0: Writing a zero to this bit has no effect.
- **FLUSHFIFO: Flush Fifo Command**
  1: If The FIFO Mode is enabled (MR.FIFOEN written to one) and if an overrun error has been detected, this command allows to empty the FIFO.
  0: Writing a zero to this bit has no effect.
- **SWRST: SPI Software Reset**
  1: Writing a one to this bit will reset the SPI. A software-triggered hardware reset of the SPI interface is performed. The SPI is in slave mode after software reset. Peripheral DMA Controller channels are not affected by software reset.
  0: Writing a zero to this bit has no effect.
- **SPIDIS: SPI Disable**
  1: Writing a one to this bit will disable the SPI. As soon as SPIDIS is written to one, the SPI finishes its transfer, all pins are set in input mode and no data is received or transmitted. If a transfer is in progress, the transfer is finished before the SPI is disabled. If both SPIEN and SPIDIS are equal to one when the CR register is written, the SPI is disabled.
  0: Writing a zero to this bit has no effect.
- **SPIEN: SPI Enable**
  1: Writing a one to this bit will enable the SPI to transfer and receive data.
  0: Writing a zero to this bit has no effect.

- **BITS: Bits Per Transfer**
  The BITS field determines the number of data bits transferred. Reserved values should not be used.

| BITS | Bits Per Transfer |
|---|---|
| 0000 | 8 |
| 0001 | 9 |
| 0010 | 10 |
| 0011 | 11 |
| 0100 | 12 |
| 0101 | 13 |
| 0110 | 14 |
| 0111 | 15 |
| 1000 | 16 |
| 1001 | 4 |
| 1010 | 5 |
| 1011 | 6 |
| 1100 | 7 |
| 1101 | Reserved |
| 1110 | Reserved |
| 1111 | Reserved |

- **CSAAT: Chip Select Active After Transfer**
  1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.
  0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**
  0: The Peripheral Chip Select does not rise between two transfers if the TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
  1: The Peripheral Chip Select rises systematically between each transfer performed on the same slave for a minimal duration of:

  $\frac{DLYBCS}{CLKSPI}$ (if DLYBCT field is different from 0)

  $\frac{DLYBCS + 1}{CLK\overline{SPI}}$ (if DLYBCT field equals 0)
- **NCPHA: Clock Phase**
  1: Data is captured after the leading (inactive-to-active) edge of SPCK and changed on the trailing (active-to-inactive) edge of SPCK.
  0: Data is changed on the leading (inactive-to-active) edge of SPCK and captured after the trailing (active-to-inactive) edge of SPCK.
  NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.
- **CPOL: Clock Polarity**
  1: The inactive state value of SPCK is logic level one.
  0: The inactive state value of SPCK is logic level zero.

### 22.9.12 Parameter Register

**Name:** PR

**Access Type:** Read-only

**Offset**: 0x2C

**Reset Value:** -

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

### 23.7.3 Interlinked Multiple Value Duty Register

**Name:** IMDUTY

**Access Type:** Write-only

**Offset**: 0x08

**Reset Value:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| DUTY3 | | | | | | | |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| DUTY2 | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| DUTY1 | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DUTY0 | | | | | | | |

- **DUTYn: Duty Cycle**

  The value written to DUTY field *n* will be written to the PWMA channel selected by the corresponding SEL field in the IMCHSEL register.

  If the value zero is written to DUTY all affected channels will be disabled. In this state the output waveform will be zero all the time.

- **BCPC: RC Compare Effect on TIOB**

| BCPC | Effect |
|---|---|
| 0 | none |
| 1 | set |
| 2 | clear |
| 3 | toggle |

- **BCPB: RB Compare Effect on TIOB**

| BCPB | Effect |
|---|---|
| 0 | none |
| 1 | set |
| 2 | clear |
| 3 | toggle |

- **ASWTRG: Software Trigger Effect on TIOA**

| ASWTRG | Effect |
|---|---|
| 0 | none |
| 1 | set |
| 2 | clear |
| 3 | toggle |

- **AEEVT: External Event Effect on TIOA**

| AEEVT | Effect |
|---|---|
| 0 | none |
| 1 | set |
| 2 | clear |
| 3 | toggle |

- **ACPC: RC Compare Effect on TIOA**

| ACPC | Effect |
|---|---|
| 0 | none |
| 1 | set |
| 2 | clear |
| 3 | toggle |

### 24.7.4 Channel Counter Value Register

**Name:** CV

**Access Type:** Read-only

**Offset:** 0x10 + n * 0x40

**Reset Value:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| CV[15:8] | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CV[7:0] | | | | | | | |

- **CV: Counter Value**
  CV contains the counter value in real time.

### 24.7.10 Channel Interrupt Disable Register

**Name:** IDR

**Access Type:** Write-only

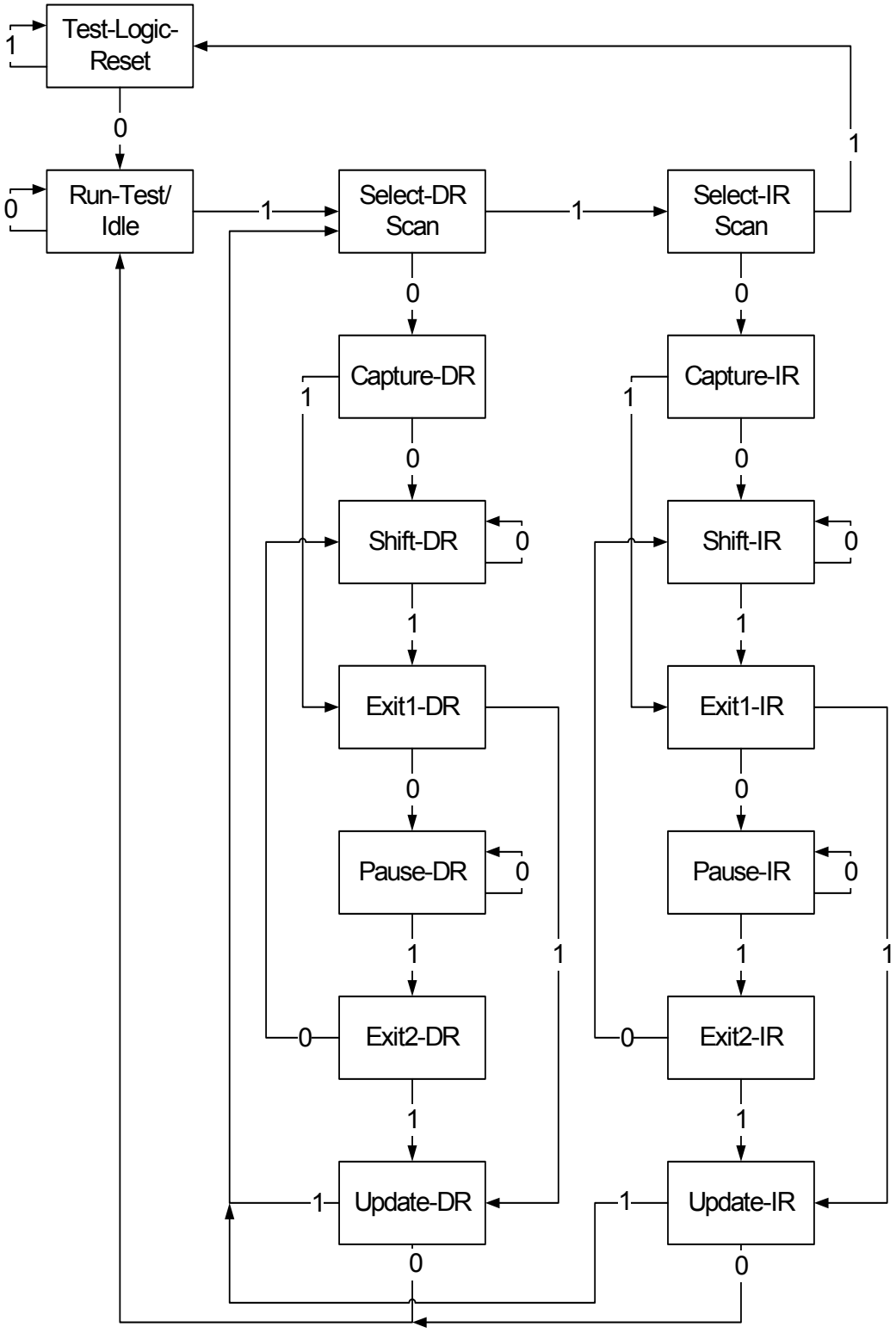
**Offset:** 0x28 + n * 0x40

**Reset Value:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| - | - | - | - | - | - | - | - |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| - | - | - | - | - | - | - | - |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

**Figure 31-6.** TAP Controller State Diagram



continue shifting the same instruction until the busy bit clears, or start shifting data. If shifting data, you must be prepared that the data shift may also report busy.

- During Shift-DR of an address: The new address is ignored. The SAB stays in address mode, so no data must be shifted. Repeat the address until the busy bit clears.
- During Shift-DR of read data: The read data is invalid. The SAB stays in data mode. Repeat scanning until the busy bit clears.
- During Shift-DR of write data: The write data is ignored. The SAB stays in data mode. Repeat scanning until the busy bit clears.

#### 31.4.11.5 *Error Reporting*

The Service Access Bus may not be able to complete all accesses as requested. This may be because the address is invalid, the addressed area is read-only or cannot handle byte/halfword accesses, or because the chip is set in a protected mode where only limited accesses are allowed.

The error bit is updated when an access completes, and is cleared when a new access starts.

What to do if the error bit is set:

- During Shift-IR: The new instruction is selected. The last operation performed using the old instruction did not complete successfully.
- During Shift-DR of an address: The previous operation failed. The new address is accepted. If the read bit is set, a read operation is started.
- During Shift-DR of read data: The read operation failed, and the read data is invalid.
- During Shift-DR of write data: The previous write operation failed. The new data is accepted and a write operation started. This should only occur during block writes or stream writes. No error can occur between scanning a write address and the following write data.
- While polling with CANCEL_ACCESS: The previous access was cancelled. It may or may not have actually completed.
- After power-up: The error bit is set after power up, but there has been no previous SAB instruction so this error can be discarded.

#### 31.4.11.6 *Protected Reporting*

A protected status may be reported during Shift-IR or Shift-DR. This indicates that the security bit in the Flash Controller is set and that the chip is locked for access, according to Section 31.5.1.

The protected state is reported when:

- The Flash Controller is under reset. This can be due to the AVR_RESET command or the RESET_N line.
- The Flash Controller has not read the security bit from the flash yet (This will take a a few ms). Happens after the Flash Controller reset has been released.
- The security bit in the Flash Controller is set.

What to do if the protected bit is set:

- Release all active AVR_RESET domains, if any.
- Release the RESET_N line.
- Wait a few ms for the security bit to clear. It can be set temporarily due to a reset.

### 31.6.8 aWire Response Summary

The implemented aWire responses are shown in the table below.

**Table 31-49.** aWire Response Summary

| RESPONSE | Instruction | Description |
|---|---|---|
| 0x40 | ACK | Acknowledge. |
| 0x41 | NACK | Not acknowledge. Sent after CRC errors and after unknown commands. |
| 0xC0 | IDCODE | The JTAG idcode. |
| 0xC1 | MEMDATA | Values read from memory. |
| 0xC2 | MEMORY_READWRITE_STATUS | Status after a MEMORY_WRITE or a MEMORY_READ command. OK, busy, error. |
| 0xC3 | BAUD_RATE | The current baudrate. |
| 0xC4 | STATUS_INFO | Status information. |
| 0xC5 | MEMORY_SPEED | SAB to aWire speed information. |

#### *31.6.8.1 ACK*

The AW has received the command successfully and performed the operation.

**Table 31-50.** ACK Details

| Response | Details |
|---|---|
| Response value | 0x40 |
| Additional data | N/A |

#### *31.6.8.2 NACK*

The AW has received the command, but got a CRC mismatch.

**Table 31-51.** NACK Details

| Response | Details |
|---|---|
| Response value | 0x41 |
| Additional data | N/A |

#### *31.6.8.3 IDCODE*

The JTAG idcode for this device.

**Table 31-52.** IDCODE Details

| Response | Details |
|---|---|
| Response value | 0xC0 |
| Additional data | JTAG idcode |

#### *31.6.8.4 MEMDATA*

The data read from the address specified by the MEMORY_READ command. The last 3 bytes are status bytes from the read. The first status byte is the status of the command described in the table below. The last 2 bytes are the number of remaining data bytes to be sent in the data field of the packet when the error occurred. If the read was not successful all data bytes after the failure are undefined. A successful word read (4 bytes) will look like this:

## 32.5 I/O Pin Characteristics

**Table 32-7.** Normal I/O Pin Characteristics[1]

| Symbol | Parameter | Condition | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| $R_{PULLUP}$ | Pull-up resistance | | 75 | 100 | 145 | kOhm |
| $V_{IL}$ | Input low-level voltage | $V_{VDD}$ = 3.0V | -0.3 | | 0.3*$V_{VDD}$ | V |
| | | $V_{VDD}$ = 1.62V | -0.3 | | 0.3*$V_{VDD}$ | |
| $V_{IH}$ | Input high-level voltage | $V_{VDD}$ = 3.6V | 0.7*$V_{VDD}$ | | $V_{VDD}$ + 0.3 | V |
| | | $V_{VDD}$ = 1.98V | 0.7*$V_{VDD}$ | | $V_{VDD}$ + 0.3 | |
| $V_{OL}$ | Output low-level voltage | $V_{VDD}$ = 3.0V, $I_{OL}$ = 3mA | | | 0.4 | V |
| | | $V_{VDD}$ = 1.62V, $I_{OL}$ = 2mA | | | 0.4 | |
| $V_{OH}$ | Output high-level voltage | $V_{VDD}$ = 3.0V, $I_{OH}$ = 3mA | $V_{VDD}$ - 0.4 | | | V |
| | | $V_{VDD}$ = 1.62V, $I_{OH}$ = 2mA | $V_{VDD}$ - 0.4 | | | |
| $f_{MAX}$ | Output frequency[2] | $V_{VDD}$ = 3.0V, load = 10pF | | | 45 | MHz |
| | | $V_{VDD}$ = 3.0V, load = 30pF | | | 23 | |
| $t_{RISE}$ | Rise time[2] | $V_{VDD}$ = 3.0V, load = 10pF | | | 4.7 | ns |
| | | $V_{VDD}$ = 3.0V, load = 30pF | | | 11.5 | |
| $t_{FALL}$ | Fall time[2] | $V_{VDD}$ = 3.0V, load = 10pF | | | 4.8 | |
| | | $V_{VDD}$ = 3.0V, load = 30pF | | | 12 | |
| $I_{LEAK}$ | Input leakage current | Pull-up resistors disabled | | | 1 | µA |
| $C_{IN}$ | Input capacitance, all normal I/O pins except PA05, PA07, PA17, PA20, PA21, PB04, PB05 | TQFP48 package | | 1.4 | | pF |
| | | QFN48 package | | 1.1 | | |
| | | TLLGA 48 package | | 1.1 | | |
| $C_{IN}$ | Input capacitance, PA20 | TQFP48 package | | 2.7 | | |
| | | QFN48 package | | 2.4 | | |
| | | TLLGA 48 package | | 2.4 | | |
| $C_{IN}$ | Input capacitance, PA05, PA07, PA17, PA21, PB04, PB05 | TQFP48 package | | 3.8 | | |
| | | QFN48 package | | 3.5 | | |
| | | TLLGA 48 package | | 3.5 | | |

Notes: 1. $V_{VDD}$ corresponds to either $V_{VDDIN}$ or $V_{VDDIO}$, depending on the supply for the pin. Refer to Section 3.2 on page 9 for details.
2. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

**Table 32-8.** High-drive I/O Pin Characteristics[1]

| Symbol | Parameter | Condition | Min | Typ | Max | Units |
|---|---|---|---|---|---|---|
| $R_{PULLUP}$ | Pull-up resistance | PA06 | 30 | 50 | 110 | kOhm |
| | | PA02, PB01, RESET | 75 | 100 | 145 | |
| | | PA08, PA09 | 10 | 20 | 45 | |

**Figure 32-16.** SPI Slave Mode With (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

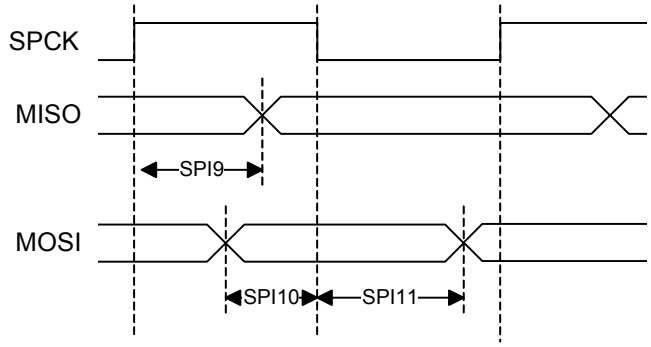



**Figure 32-17.** SPI Slave Mode NPCS Timing

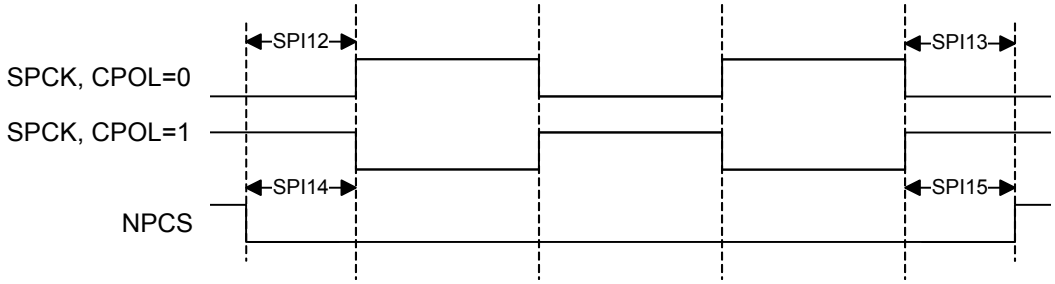



**Table 32-41.** SPI Timing, Slave Mode[1]

| Symbol | Parameter | Conditions | Min | Max | Units |
|---|---|---|---|---|---|
| SPI6 | SPCK falling to MISO delay | $V_{VDDIO}$ from 3.0V to 3.6V, maximum external capacitor = 40pF | | 30.8 | ns |
| SPI7 | MOSI setup time before SPCK rises | | 0 | | |
| SPI8 | MOSI hold time after SPCK rises | | 4.1 | | |
| SPI9 | SPCK rising to MISO delay | | | 29.9 | |
| SPI10 | MOSI setup time before SPCK falls | | 0 | | |
| SPI11 | MOSI hold time after SPCK falls | | 3.5 | | |
| SPI12 | NPCS setup time before SPCK rises | | 1.9 | | |
| SPI13 | NPCS hold time after SPCK falls | | 0.2 | | |
| SPI14 | NPCS setup time before SPCK falls | | 2.2 | | |
| SPI15 | NPCS hold time after SPCK rises | | 0 | | |

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

**Maximum SPI Frequency, Slave Input Mode**

The CAT module does not terminate a QTouch burst when the detection voltage is reached on the sense capacitor. This can cause the sense capacitor to be charged more than necessary. Depending on the dielectric absorption characteristics of the capacitor, this can lead to unstable measurements.
**Fix/Workaround**
Use the minimum possible value for the MAX field in the ATCFG1, TG0CFG1, and TG1CFG1 registers.

4. **Autonomous CAT acquisition must be longer than AST source clock period**
When using the AST to trigger CAT autonomous touch acquisition in sleep modes where the CAT bus clock is turned off, the CAT will start several acquisitions if the period of the AST source clock is larger than one CAT acquisition. One AST clock period after the AST trigger, the CAT clock will automatically stop and the CAT acquisition can be stopped prematurely, ruining the result.
**Fix/Workaround**
Always ensure that the ATCFG1.max field is set so that the duration of the autonomous touch acquisition is greater than one clock period of the AST source clock.

### 35.1.13 aWire

1. **aWire CPU clock speed robustness**
The aWire memory speed request command counter warps at clock speeds below approximately 5kHz.
**Fix/Workaround**
None.

2. **The aWire debug interface is reset after leaving Shutdown mode**
If the aWire debug mode is used as debug interface and the program enters Shutdown mode, the aWire interface will be reset when the part receives a wakeup either from the WAKE_N pin or the AST.
**Fix/Workaround**
None.

### 35.1.14 CHIP

1. **Increased Power Consumption in VDDIO in sleep modes**
If OSC0 is enabled in crystal mode when entering a sleep mode where the OSC0 is disabled, this will lead to an increased power consumption in VDDIO.
**Fix/Workaround**
Solution 1: Disable OSC0 by writing a zero to the Oscillator Enable bit in the System Control Interface (SCIF) Oscillator Control Register (SCIF.OSC0CTRL.OSCEN) before going to a sleep mode where OSC0 is disabled.
Solution 2: Pull down or up XIN0 or XOUT0 with 1MOhm resistor.

### 35.1.15 I/O Pins

1. **PA17 has low ESD tolerance**
PA17 only tolerates 500V ESD pulses (Human Body Model).
**Fix/Workaround**
Care must be taken during manufacturing and PCB design.

2. **If a reset happens during the last SAB write, the aWire will stall**
   If a reset happens during the last word, halfword or byte write the aWire will wait forever for an acknowledge from the SAB.
   **Fix/Workaround**
   Reset the aWire by keeping the RESET_N line low for 100ms.

3. **aWire enable does not work in Static mode**
   aWire enable does not work in Static mode.
   **Fix/Workaround**
   None.

4. **aWire CPU clock speed robustness**
   The aWire memory speed request command counter warps at clock speeds below approximately 5kHz.
   **Fix/Workaround**
   None.

5. **The aWire debug interface is reset after leaving Shutdown mode**
   If the aWire debug mode is used as debug interface and the program enters Shutdown mode, the aWire interface will be reset when the part receives a wakeup either from the WAKE_N pin or the AST.
   **Fix/Workaround**
   None.

6. **aWire PB mapping and PB clock mask number**
   The aWire PB has a different PB address and PB clock mask number.
   **Fix/Workaround**
   Use aWire PB address 0xFFFF6C00 and PB clock (PBAMASK) 24.

7. **VERSION register reads 0x200**
   The VERSION register reads 0x200 instead of 0x210.
   **Fix/Workaround**
   None.

### 35.4.22 Chip

1. **WAKE_N pin can only wake up the chip from Shutdown mode**
   It is not possible to wake up the chip from any other sleep mode than Shutdown using the WAKE_N pin. If the WAKE_N pin is asserted during a sleep mode other than Shutdown, nothing will happen.
   **Fix/Workaround**
   Use an EIC pin to wake up from sleep modes higher than Shutdown.

2. **Power consumption in static mode is too high**
   Power consumption in static mode is too high when PA21 is high.
   **Fix/Workaround**
   Ensure PA21 is low.

3. **Shutdown mode is not functional**
   Do not enter Shutdown mode.
   **Fix/Workaround**
   None.

4. **VDDIN current consumption increase above 1.8V**