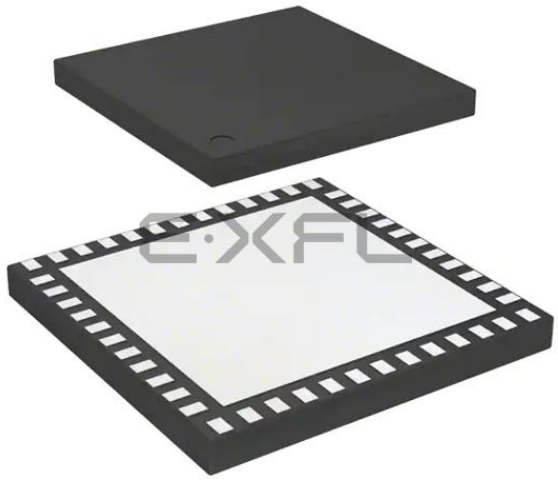


**Welcome to E-XFL.COM**

## 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, USB |
| Peripherals | Brown-out Detect/Reset, DMA, POR, PWM, WDT |
| Number of I/O | 36 |
| Program Memory Size | 64KB (64K 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-UFLGA Exposed Pad |
| Supplier Device Package | 48-TLLGA (5.5x5.5) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atuc64l4u-d3ht |

address exception. Doubleword-sized accesses with word-aligned pointers will automatically be performed as two word-sized accesses.

The following table shows the instructions with support for unaligned addresses. All other instructions require aligned addresses.

**Table 4-1.** Instructions with Unaligned Reference Support

| Instruction | Supported Alignment |
|---|---|
| ld.d | Word |
| st.d | Word |

#### 4.3.2.5 *Unimplemented Instructions*

The following instructions are unimplemented in AVR32UC, and will cause an Unimplemented Instruction Exception if executed:

- All SIMD instructions
- All coprocessor instructions if no coprocessors are present
- retj, incjosp, popjc, pushjc
- tlbr, tlbs, tlbw
- cache

#### 4.3.2.6 *CPU and Architecture Revision*

Three major revisions of the AVR32UC CPU currently exist. The device described in this datasheet uses CPU revision 3.

The Architecture Revision field in the CONFIG0 system register identifies which architecture revision is implemented in a specific device.

AVR32UC CPU revision 3 is fully backward-compatible with revisions 1 and 2, ie. code compiled for revision 1 or 2 is binary-compatible with revision 3 CPUs.

**Table 4-4.** Priority and Handler Addresses for Events

| Priority | Handler Address | Name | Event source | Stored Return Address |
|---|---|---|---|---|
| 1 | 0x80000000 | Reset | External input | Undefined |
| 2 | Provided by OCD system | OCD Stop CPU | OCD system | First non-completed instruction |
| 3 | EVBA+0x00 | Unrecoverable exception | Internal | PC of offending instruction |
| 4 | EVBA+0x04 | TLB multiple hit | MPU | PC of offending instruction |
| 5 | EVBA+0x08 | Bus error data fetch | Data bus | First non-completed instruction |
| 6 | EVBA+0x0C | Bus error instruction fetch | Data bus | First non-completed instruction |
| 7 | EVBA+0x10 | NMI | External input | First non-completed instruction |
| 8 | Autovectored | Interrupt 3 request | External input | First non-completed instruction |
| 9 | Autovectored | Interrupt 2 request | External input | First non-completed instruction |
| 10 | Autovectored | Interrupt 1 request | External input | First non-completed instruction |
| 11 | Autovectored | Interrupt 0 request | External input | First non-completed instruction |
| 12 | EVBA+0x14 | Instruction Address | CPU | PC of offending instruction |
| 13 | EVBA+0x50 | ITLB Miss | MPU | PC of offending instruction |
| 14 | EVBA+0x18 | ITLB Protection | MPU | PC of offending instruction |
| 15 | EVBA+0x1C | Breakpoint | OCD system | First non-completed instruction |
| 16 | EVBA+0x20 | Illegal Opcode | Instruction | PC of offending instruction |
| 17 | EVBA+0x24 | Unimplemented instruction | Instruction | PC of offending instruction |
| 18 | EVBA+0x28 | Privilege violation | Instruction | PC of offending instruction |
| 19 | EVBA+0x2C | Floating-point | UNUSED | |
| 20 | EVBA+0x30 | Coprocessor absent | Instruction | PC of offending instruction |
| 21 | EVBA+0x100 | Supervisor call | Instruction | PC(Supervisor Call) +2 |
| 22 | EVBA+0x34 | Data Address (Read) | CPU | PC of offending instruction |
| 23 | EVBA+0x38 | Data Address (Write) | CPU | PC of offending instruction |
| 24 | EVBA+0x60 | DTLB Miss (Read) | MPU | PC of offending instruction |
| 25 | EVBA+0x70 | DTLB Miss (Write) | MPU | PC of offending instruction |
| 26 | EVBA+0x3C | DTLB Protection (Read) | MPU | PC of offending instruction |
| 27 | EVBA+0x40 | DTLB Protection (Write) | MPU | PC of offending instruction |
| 28 | EVBA+0x44 | DTLB Modified | UNUSED | |

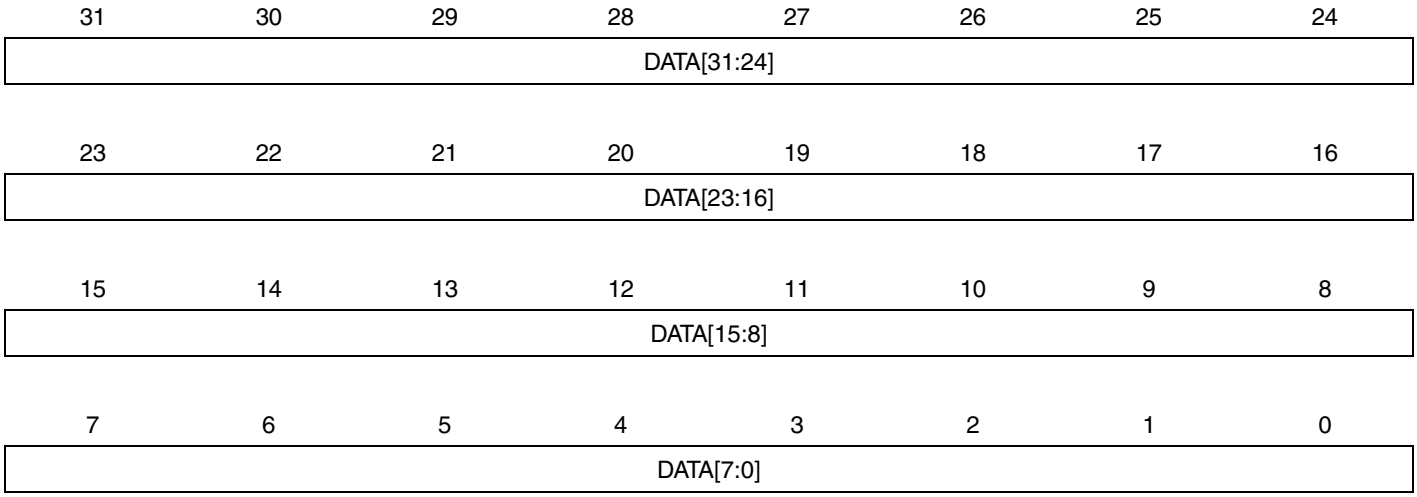
### 7.7.21 Performance Channel 0 Write Data Cycles

**Name:** PWDATA0

**Access Type:** Read-only

**Offset:** 0x810

**Reset Value:** 0x00000000




- **DATA: Data Cycles Counted Since Last Reset**
  Clock cycles are counted using the CLK_PDCA_HSB clock

*8.7.1.6 Features Register*

**Register Name:** UFEATURES

**Access Type:** Read-Only

**Offset:** 0x081C

**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 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | EPTNBRMAX | | | |

- **EPTNBRMAX: Maximal Number of pipes/endpoints**
  This field indicates the number of hardware-implemented pipes/endpoints:

### 11.5.5 Special Function Registers

**Name:** SFR0...SFR15

**Access Type:** Read/Write

**Offset:** 0x110 - 0x14C

**Reset Value:** -

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

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

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

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

- **SFR: Special Function Register Fields**

  Those registers are not a HMATRIX specific register. The field of those will be defined where they are used.

1: Power-on Reset 3.3V is masked.

- **POR33STATUS: Power-on Reset 3.3V Status**
  0: Power-on Reset is disabled.
  1: Power-on Reset is enabled.
  This bit is read-only. Writing to this bit has no effect.
- **POR33EN: Power-on Reset 3.3V Enable**
  0: Writing a zero to this bit disables the POR33 detector.
  1: Writing a one to this bit enables the POR33 detector.
- **DEEPDIS: Disable Regulator Deep Mode**
  0: Regulator will enter deep mode in low-power sleep modes for lower power consumption.
  1: Regulator will stay in full-power mode in all sleep modes for shorter start-up time.
- **FCD: Flash Calibration Done**
  0: The flash calibration will be redone after any reset.
  1: The flash calibration will only be redone after a Power-on Reset.
  This bit is cleared after a Power-on Reset.
  This bit is set when the CALIB field has been updated by flash calibration after a reset.
- **CALIB: Calibration Value**
  Calibration value for Voltage Regulator. This is calibrated during production and should not be changed.
- **ON: Voltage Regulator On Status**
  0: The voltage regulator is currently disabled.
  1: The voltage regulator is currently enabled.
  This bit is read-only. Writing to this bit has no effect.
- **VREGOK: Voltage Regulator OK Status**
  0: The voltage regulator is disabled or has not yet reached a stable output voltage.
  1: The voltage regulator has reached the output voltage threshold level after being enabled.
  This bit is read-only. Writing to this bit has no effect.
- **EN: Enable**
  0: The voltage regulator is disabled.
  1: The voltage regulator is enabled.
  **Note**: This bit is set after a Power-on Reset (POR).
- **SELVDD: Select VDD**
  Output voltage of the Voltage Regulator. The default value of this bit corresponds to an output voltage of 1.8V.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

### 14.6.21 120MHz RC Oscillator Configuration Register

**Name:** RC120MCR

**Access Type:** Read/Write

**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 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | EN |

- **EN: RC120M Enable**
  0: The 120 MHz RC oscillator is disabled.
  1: The 120 MHz RC oscillator is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

**Figure 20-28.** Master Node Configuration, NACT=SUBSCRIBE

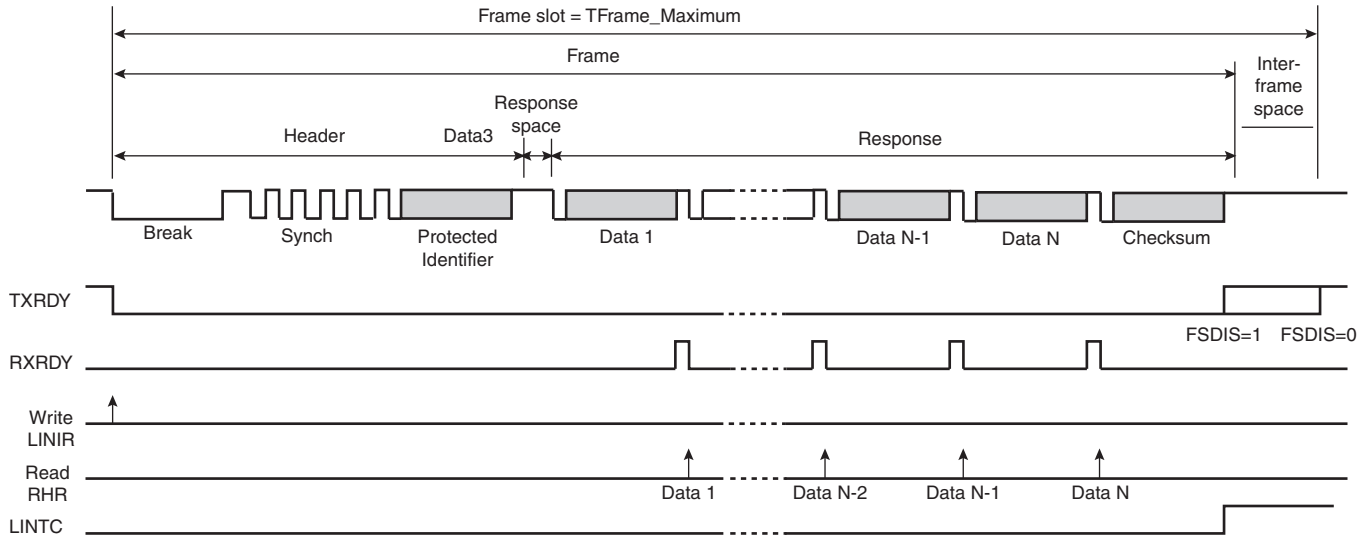
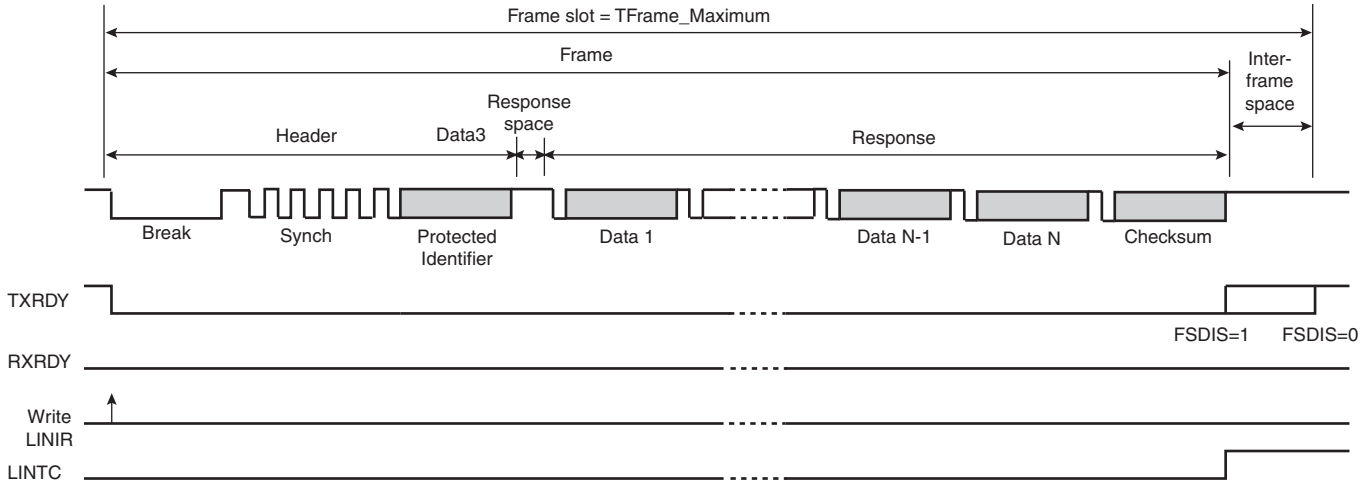



**Figure 20-29.** Master Node Configuration, NACT=IGNORE




### *20.6.7.2 Slave Node Configuration*

This is identical to the master node configuration above, except for:

- LIN mode selected in MR.MODE is slave
- When the baud rate is configured, wait until CSR.LINID is a one, then;
- Check for LINISFE and LINPE errors, clear errors and LINIDby writing a one to RSTSTA
- Read IDCHR
- Configure the frame transfer by writing to NACT, PARDIS, CHKDIS, CHKTYPE, DLCM, and DLC in LINMR

**IMPORTANT**: if NACT=PUBLISH, and this field is already correct, the LINMR register must still be written with this value in order to set TXRDY, and to request the corresponding Peripheral DMA Controller write transfer.

The different NACT settings result in the same procedure as for the master node, see page 455.

# 21. Serial Peripheral Interface (SPI)

Rev: 2.1.1.3

## 21.1 Features

- **Compatible with an embedded 32-bit microcontroller**
- **Supports communication with serial external devices**
  - **Four chip selects with external decoder support allow communication with up to 15 peripherals**
  - **Serial memories, such as DataFlash and 3-wire EEPROMs**
  - **Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and Sensors**
  - **External co-processors**
- **Master or Slave Serial Peripheral Bus Interface**
  - **4 - to 16-bit programmable data length per chip select**
  - **Programmable phase and polarity per chip select**
  - **Programmable transfer delays between consecutive transfers and between clock and data per chip select**
  - **Programmable delay between consecutive transfers**
  - **Selectable mode fault detection**
- **Connection to Peripheral DMA Controller channel capabilities optimizes data transfers**
  - **One channel for the receiver, one channel for the transmitter**
  - **Next buffer support**
  - **Four character FIFO in reception**

## 21.2 Overview

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master"' which controls the data flow, while the other devices act as “slaves'' which have data shifted into and out by the master. Different CPUs can take turn being masters (Multiple Master Protocol opposite to Single Master Protocol where one CPU is always the master while all of the others are always slaves) and one master may simultaneously shift data into multiple slaves. However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): this data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO): this data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK): this control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SPCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): this control line allows slaves to be turned on and off by hardware.

### 21.8.9 Chip Select Register 0

**Name:** CSR0

**Access Type:** Read/Write

**Offset:** 0x30

**Reset Value:** 0x00000000

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

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

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

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BITS | | | | CSAAT | CSNAAT | NCPHA | CPOL |

- **DLYBCT: Delay Between Consecutive Transfers**

  This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

  When DLYBCT equals zero, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

  Otherwise, the following equation determines the delay:

$$\text{Delay Between Consecutive Transfers} = \frac{32 \times DLYBCT}{CLKSPI}$$

- **DLYBS: Delay Before SPCK**

  This field defines the delay from NPCS valid to the first valid SPCK transition.

  When DLYBS equals zero, the NPCS valid to SPCK transition is 1/2 the SPCK clock period.

  Otherwise, the following equations determine the delay:

$$\text{Delay Before SPCK} = \frac{DLYBS}{CLKSPI}$$

- **SCBR: Serial Clock Baud Rate**

  In Master Mode, the SPI Interface uses a modulus counter to derive the SPCK baud rate from the CLK_SPI. The Baud rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK baud rate:

$$\text{SPCK Baudrate} = \frac{CLKSPI}{SCBR}$$

  Writing the SCBR field to zero is forbidden. Triggering a transfer while SCBR is zero can lead to unpredictable results.

  At reset, SCBR is zero and the user has to write it to a valid value before performing the first transfer.

  If a clock divider (SCBRn) field is set to one and the other SCBR fields differ from one, access on CSn is correct but no correct access will be possible on other CS.

## 24.9 Module configuration

The specific configuration for each IISC instance is listed in the following tables. The module bus clocks listed here are connected to the system bus clocks. Please refer to the Power Manager chapter for details.

**Table 24-7.** IISC Clocks

| Clock Name | Description |
|---|---|
| CLK_IISC | Clock for the IISC bus interface |
| GCLK | The generic clock used for the IISC is GCLK6 |

**Table 24-8.** Register Reset Values

| Register | Reset Value |
|---|---|
| VERSION | 0x00000100 |

### 30.5.4 Interrupts

The ACIFB interrupt request line is connected to the interrupt controller. Using the ACIFB interrupt requires the interrupt controller to be programmed first.

### 30.5.5 Peripheral Events

The ACIFB peripheral events are connected via the Peripheral Event System. Refer to the Peripheral Event System chapter for details.

### 30.5.6 Debug Operation

When an external debugger forces the CPU into debug mode, the ACIFB continues normal operation. If the ACIFB is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

## 30.6 Functional Description

The ACIFB is enabled by writing a one to the Control Register Enable bit (CTRL.EN). Additionally, the comparators must be individually enabled by programming the MODE field in the AC Configuration Register (CONFn.MODE).

The results from the individual comparators can either be used directly (normal mode), or the results from two comparators can be grouped to generate a comparison window (window mode). All comparators need not be in the same mode, some comparators may be in normal mode, while others are in window mode. There are restrictions on which AC channels that can be grouped together in a window pair, see Section 30.6.5.

### 30.6.1 Analog Comparator Operation

Each AC channel can be in one of four different modes, determined by CONFn.MODE:

- Off
- Continuous Measurement Mode (CM)
- User Triggered Single Measurement Mode (UT)
- Event Triggered Single Measurement Mode (ET)

After being enabled, a startup time defined in CTRL.SUT is required before the result of the comparison is ready. The GCLK is used for measuring the startup time of a comparator,

During the startup time the AC output is not available. When the ACn Ready bit in the Status Register (SR.ACRDYn) is one, the output of ACn is ready. In window mode the result is available when both the comparator outputs are ready (SR.ACRDYn=1 and SR.ACRDYn+1=1).

#### *30.6.1.1 Continuous Measurement Mode*

In CM, the Analog Comparator is continuously enabled and performing comparisons. This ensures that the result of the latest comparison is always available in the ACn Current Comparison Status bit in the Status Register (SR.ACCSn). Comparisons are done on every positive edge of GCLK.

CM is enabled by writing CONFn.MODE to 1. After the startup time has passed, a comparison is done and SR is updated. Appropriate peripheral events and interrupts are also generated. New comparisons are performed continuously until the CONFn.MODE field is written to 0.

## 30.9 User Interface

**Table 30-4.** ACIFB Register Memory Map

| Offset | Register | Register Name | Access | Reset |
|---|---|---|---|---|
| 0x00 | Control Register | CTRL | Read/Write | 0x00000000 |
| 0x04 | Status Register | SR | Read-only | 0x00000000 |
| 0x10 | Interrupt Enable Register | IER | Write-only | 0x00000000 |
| 0x14 | Interrupt Disable Register | IDR | Write-only | 0x00000000 |
| 0x18 | Interrupt Mask Register | IMR | Read-only | 0x00000000 |
| 0x1C | Interrupt Status Register | ISR | Read-only | 0x00000000 |
| 0x20 | Interrupt Status Clear Register | ICR | Write-only | 0x00000000 |
| 0x24 | Test Register | TR | Read/Write | 0x00000000 |
| 0x30 | Parameter Register | PARAMETER | Read-only | -(1) |
| 0x34 | Version Register | VERSION | Read-only | -(1) |
| 0x80 | Window0 Configuration Register | CONFW0 | Read/Write | 0x00000000 |
| 0x84 | Window1 Configuration Register | CONFW1 | Read/Write | 0x00000000 |
| 0x88 | Window2 Configuration Register | CONFW2 | Read/Write | 0x00000000 |
| 0x8C | Window3 Configuration Register | CONFW3 | Read/Write | 0x00000000 |
| 0xD0 | AC0 Configuration Register | CONF0 | Read/Write | 0x00000000 |
| 0xD4 | AC1 Configuration Register | CONF1 | Read/Write | 0x00000000 |
| 0xD8 | AC2 Configuration Register | CONF2 | Read/Write | 0x00000000 |
| 0xDC | AC3 Configuration Register | CONF3 | Read/Write | 0x00000000 |
| 0xE0 | AC4 Configuration Register | CONF4 | Read/Write | 0x00000000 |
| 0xE4 | AC5 Configuration Register | CONF5 | Read/Write | 0x00000000 |
| 0xE8 | AC6 Configuration Register | CONF6 | Read/Write | 0x00000000 |
| 0xEC | AC7 Configuration Register | CONF7 | Read/Write | 0x00000000 |

Note: 1. The reset values for these registers are device specific. Please refer to the Module Configuration section at the end of this chapter.

# 32. Glue Logic Controller (GLOC)

Rev: 1.0.0.0

## 32.1 Features

- **Glue logic for general purpose PCB design**
- **Programmable lookup table**
- **Up to four inputs supported per lookup table**
- **Optional filtering of output**

## 32.2 Overview

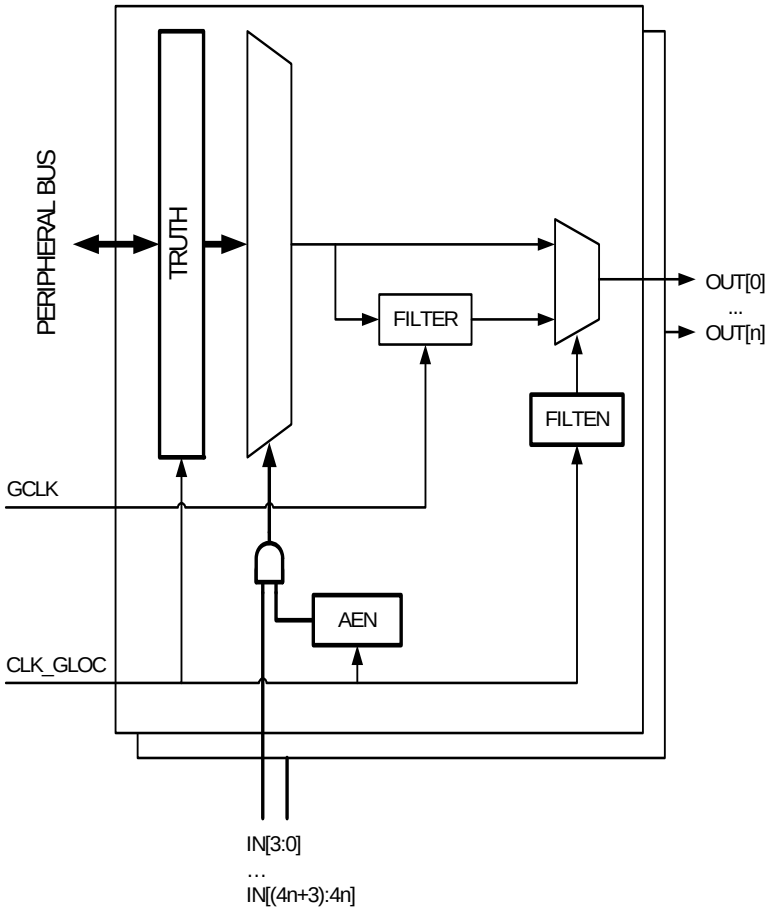
The Glue Logic Controller (GLOC) contains programmable logic which can be connected to the device pins. This allows the user to eliminate logic gates for simple glue logic functions on the PCB.

The GLOC consists of a number of lookup table (LUT) units. Each LUT can generate an output as a user programmable logic expression with four inputs. Inputs can be individually masked.

The output can be combinatorially generated from the inputs, or filtered to remove spikes.

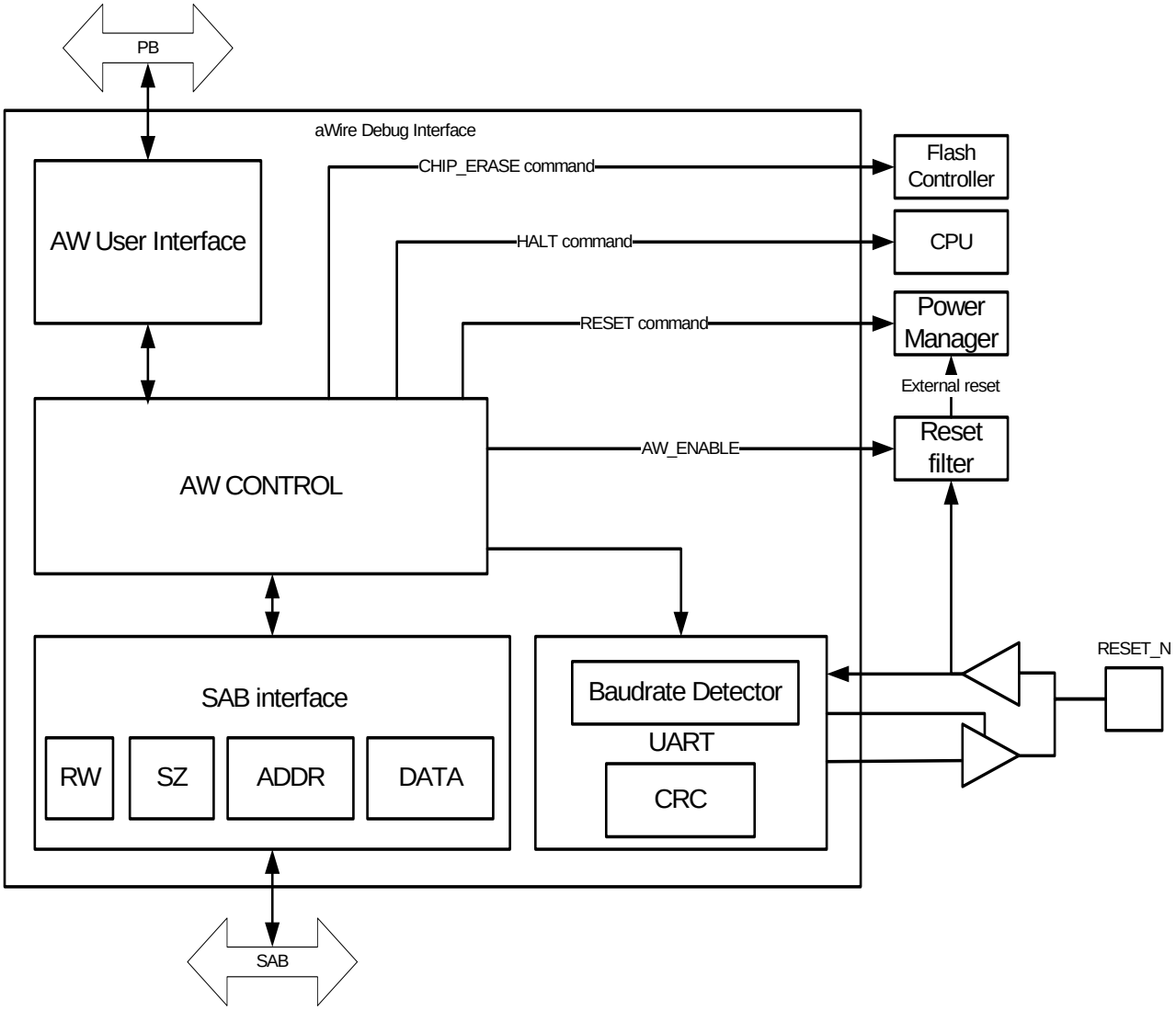
## 32.3 Block Diagram

**Figure 32-1.** GLOC Block Diagram



### 34.6.3 Block Diagram

**Figure 34-8.** aWire Debug Interface Block Diagram




### 34.6.4 I/O Lines Description

**Table 34-29.** I/O Lines Description

| Name | Description | Type |
|---|---|---|
| DATA | aWire data multiplexed with the RESET_N pin. | Input/Output |
| DATAOUT | aWire data output in 2-pin mode. | Output |

### 34.6.5 Product Dependencies

In order to use this module, other parts of the system must be configured correctly, as described below.

### 34.6.7 aWire Command Summary

The implemented aWire commands are shown in the table below. The responses from the AW are listed in Section 34.6.8.

**Table 34-31.** aWire Command Summary

| COMMAND | Instruction | Description |
|---|---|---|
| 0x01 | AYA | “Are you alive”. |
| 0x02 | JTAG_ID | Asks AW to return the JTAG IDCODE. |
| 0x03 | STATUS_REQUEST | Request a status message from the AW. |
| 0x04 | TUNE | Tell the AW to report the current baud rate. |
| 0x05 | MEMORY_SPEED_REQUEST | Reports the speed difference between the aWire control and the SAB clock domains. |
| 0x06 | CHIP_ERASE | Erases the flash and all volatile memories. |
| 0x07 | DISABLE | Disables the AW. |
| 0x08 | 2_PIN_MODE | Enables the DATAOUT pin and puts the aWire in 2-pin mode, where all responses are sent on the DATAOUT pin. |
| 0x80 | MEMORY_WRITE | Writes words, halfwords, or bytes to the SAB. |
| 0x81 | MEMORY_READ | Reads words, halfwords, or bytes from the SAB. |
| 0x82 | HALT | Issues a halt command to the device. |
| 0x83 | RESET | Issues a reset to the Reset Controller. |
| 0x84 | SET_GUARD_TIME | Sets the guard time for the AW. |

All aWire commands are described below, with a summary in table form.

**Table 34-32.** Command/Response Description Notation

| Command/Response | Description |
|---|---|
| Command/Response value | Shows the command/response value to put into the command/response field of the packet. |
| Additional data | Shows the format of the optional data field if applicable. |
| Possible responses | Shows the possible responses for this command. |

#### *34.6.7.1 AYA*

This command asks the AW: “Are you alive”, where the AW should respond with an acknowledge.

**Table 34-33.** AYA Details

| Command | Details |
|---|---|
| Command value | 0x01 |
| Additional data | N/A |
| Possible responses | 0x40: ACK (Section 34.6.8.1)<br>0x41: NACK (Section 34.6.8.2) |

1. The size of the data field: 7 (size and starting address + read length indicator) in the length field.
2. The size of the transfer: Words, halfwords, or bytes.
3. The starting address of the transfer.
4. The number of **bytes** to read (max 65532).

The 4 MSB of the 36 bit SAB address are submitted together with the size field (2 bits). The 4 remaining address bytes are submitted before the number of bytes to read. The size of the transfer is specified using the values from the following table:

**Table 34-43.** Size Field Decoding

| Size field | Description |
|---|---|
| 00 | Byte transfer |
| 01 | Halfword transfer |
| 10 | Word transfer |
| 11 | Reserved |

Below is an example read command:

1. 0x55 (sync)
2. 0x81 (command)
3. 0x00 (length MSB)
4. 0x07 (length LSB)
5. 0x25 (size and address MSB, the two MSB of this byte are unused and set to zero)
6. 0x00
7. 0x00
8. 0x00
9. 0x04 (address LSB)
10. 0x00
11. 0x04
12. 0xXX (CRC MSB)
13. 0xXX (CRC LSB)

The length field is set to 0x0007 because there are 7 bytes of additional data: 5 bytes of address and size and 2 bytes with the number of bytes to read. The address and size field indicates one word (four bytes) should be read from address 0x500000004.

**Table 34-44.** MEMORY_READ Details

| Command | Details |
|---|---|
| Command value | 0x81 |
| Additional data | Size, Address and Length |
| Possible responses | 0xC1: MEMDATA (Section 34.6.8.4)<br>0xC2: MEMORY_READWRITE_STATUS (Section 34.6.8.5)<br>0x41: NACK (Section 34.6.8.2) |

# 35. Electrical Characteristics

## 35.1 Absolute Maximum Ratings*

**Table 35-1.** Absolute Maximum Ratings

| |
|---|
| Operating temperature..... -40°C to +85°C |
| Storage temperature..... -60°C to +150°C |
| Voltage on input pins (except for 5V pins) with respect to ground .....-0.3V to $V_{VDD}$[2]+0.3V |
| Voltage on 5V tolerant[1] pins with respect to ground ..... .....-0.3V to 5.5V |
| Total DC output current on all I/O pins - VDDIO, 64-pin package ..... .....141 mA |
| Total DC output current on all I/O pins - VDDIN, 64-pin package ..... 42 mA |
| Total DC output current on all I/O pins - VDDIO, 48-pin package ..... .....120mA |
| Total DC output current on all I/O pins - VDDIN, 48-pin package ..... 39 mA |
| Maximum operating voltage VDDCORE..... 1.98V |
| Maximum operating voltage VDDIO, VDDIN ..... 3.6V |

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or 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.

Notes: 1. 5V tolerant pins, see Section "Peripheral Multiplexing on I/O lines" on page 10
2. $V_{VDD}$ corresponds to either $V_{VDDIN}$ or $V_{VDDIO}$, depending on the supply for the pin. Refer to Section on page 10 for details.

## 35.2 Supply Characteristics

The following characteristics are applicable to the operating temperature range: $T_A$ = -40°C to 85°C, unless otherwise specified and are valid for a junction temperature up to $T_J$ = 100°C. Please refer to Section 6. "Supply and Startup Considerations" on page 39.

**Table 35-2.** Supply Characteristics

| Symbol | Parameter | Voltage | | |
|---|---|---|---|---|
| | | Min | Max | Unit |
| $V_{VDDIO}$ | DC supply peripheral I/Os | 1.62 | 3.6 | V |
| $V_{VDDIN}$ | DC supply peripheral I/Os, 1.8V single supply mode | 1.62 | 1.98 | V |
| | DC supply peripheral I/Os and internal regulator, 3.3V supply mode | 1.98 | 3.6 | V |
| $V_{VDDCORE}$ | DC supply core | 1.62 | 1.98 | V |
| $V_{VDDANA}$ | Analog supply voltage | 1.62 | 1.98 | V |

# Table of Contents