

Features

- High Performance, Low Power 32-bit AVR[®] Microcontroller
 - Compact Single-Cycle RISC Instruction Set Including DSP Instruction Set
 - Read-Modify-Write Instructions and Atomic Bit Manipulation
 - Performing up to 1.51 DMIPS/MHz
 - Up to 92 DMIPS Running at 66 MHz from Flash (1 Wait-State)
 - Up to 54 DMIPS Running at 36 MHz from Flash (0 Wait-State)
 - Memory Protection Unit
- Multi-Layer Bus System
 - High-Performance Data Transfers on Separate Buses for Increased Performance
 - 8 Peripheral DMA Channels (PDCA) Improves Speed for Peripheral Communication
 - 4 generic DMA Channels for High Bandwidth Data Paths
- Internal High-Speed Flash
 - 256 KBytes, 128 KBytes, 64 KBytes versions
 - Single-Cycle Flash Access up to 36 MHz
 - Prefetch Buffer Optimizing Instruction Execution at Maximum Speed
 - 4 ms Page Programming Time and 8 ms Full-Chip Erase Time
 - 100,000 Write Cycles, 15-year Data Retention Capability
 - Flash Security Locks and User Defined Configuration Area
- Internal High-Speed SRAM
 - 64 KBytes Single-Cycle Access at Full Speed, Connected to CPU Local Bus
 - 64 KBytes (2x32 KBytes with independent access) on the Multi-Layer Bus System
- Interrupt Controller
 - Autovector Low Latency Interrupt Service with Programmable Priority
- System Functions
 - Power and Clock Manager Including Internal RC Clock and One 32 KHz Oscillator
 - Two Multipurpose Oscillators and Two Phase-Lock-Loop (PLL),
 - Watchdog Timer, Real-Time Clock Timer
- External Memories
 - Support SDRAM, SRAM, NANDFlash (1-bit and 4-bit ECC), Compact Flash
 - Up to 66 MHz
- External Storage device support
 - MultiMediaCard (MMC V4.3), Secure-Digital (SD V2.0), SDIO V1.1
 - CE-ATA V1.1, FastSD, SmartMedia, Compact Flash
 - Memory Stick: Standard Format V1.40, PRO Format V1.00, Micro
 - IDE Interface
- One Advanced Encryption System (AES) for AT32UC3A3256S, AT32UC3A3128S, AT32UC3A364S, AT32UC3A4256S, AT32UC3A4128S and AT32UC3A364S
 - 256-, 192-, 128-bit Key Algorithm, Compliant with FIPS PUB 197 Specifications
 - Buffer Encryption/Decryption Capabilities
- Universal Serial Bus (USB)
 - High-Speed USB 2.0 (480 Mbit/s) Device and Embedded Host
 - Flexible End-Point Configuration and Management with Dedicated DMA Channels
 - On-Chip Transceivers Including Pull-Ups
- One 8-channel 10-bit Analog-To-Digital Converter, multiplexed with Digital I/Os.
- Two Three-Channel 16-bit Timer/Counter (TC)
- Four Universal Synchronous/Asynchronous Receiver/Transmitters (USART)
 - Fractional Baudrate Generator



32-bit AVR[®] Microcontroller

AT32UC3A3256S
AT32UC3A3256
AT32UC3A3128S
AT32UC3A3128
AT32UC3A364S
AT32UC3A364
AT32UC3A4256S
AT32UC3A4256
AT32UC3A4128S
AT32UC3A4128
AT32UC3A464S
AT32UC3A464

Summary



- Support for SPI and LIN
 - Optionnal support for IrDA, ISO7816, Hardware Handshaking, RS485 interfaces and Modem Line
- Two Master/Slave Serial Peripheral Interfaces (SPI) with Chip Select Signals
- One Synchronous Serial Protocol Controller
 - Supports I2S and Generic Frame-Based Protocols
- Two Master/Slave Two-Wire Interface (TWI), 400kbit/s I2C-compatible
- 16-bit Stereo Audio Bitstream
 - Sample Rate Up to 50 KHz
- QTouch® Library Support
 - Capacitive Touch Buttons, Sliders, and Wheels
- QTouch® and QMatrix® AcquisitionOn-Chip Debug System (JTAG interface)
 - Nexus Class 2+, Runtime Control, Non-Intrusive Data and Program Trace
- 110 General Purpose Input/Output (GPIOs)
 - Standard or High Speed mode
 - Toggle capability: up to 66MHz
- Packages
 - 144-ball TFBGA, 11x11 mm, pitch 0.8 mm
 - 144-pin LQFP, 22x22 mm, pitch 0.5 mm
 - 100-ball VFBGA, 7x7 mm, pitch 0.65 mm
- Single 3.3V Power Supply

1. Description

The AT32UC3A3/A4 is a complete System-On-Chip microcontroller based on the AVR32 UC RISC processor running at frequencies up to 66MHz. AVR32 UC is a high-performance 32-bit RISC microprocessor core, designed for cost-sensitive embedded applications, with particular emphasis on low power consumption, high code density and high performance.

The processor implements a Memory Protection Unit (MPU) and a fast and flexible interrupt controller for supporting modern operating systems and real-time operating systems. Higher computation capabilities are achievable using a rich set of DSP instructions.

The AT32UC3A3/A4 incorporates on-chip Flash and SRAM memories for secure and fast access. 64 KBytes of SRAM are directly coupled to the AVR32 UC for performances optimization. Two blocks of 32 Kbytes SRAM are independently attached to the High Speed Bus Matrix, allowing real ping-pong management.

The Peripheral Direct Memory Access Controller (PDCA) enables data transfers between peripherals and memories without processor involvement. The PDCA drastically reduces processing overhead when transferring continuous and large data streams.

The Power Manager improves design flexibility and security: the on-chip Brown-Out Detector monitors the power supply, the CPU runs from the on-chip RC oscillator or from one of external oscillator sources, a Real-Time Clock and its associated timer keeps track of the time.

The device includes two sets of three identical 16-bit Timer/Counter (TC) channels. Each channel can be independently programmed to perform frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation. 16-bit channels are combined to operate as 32-bit channels.

The AT32UC3A3/A4 also features many communication interfaces for communication intensive applications like UART, SPI or TWI. The USART supports different communication modes, like SPI Mode and LIN Mode. Additionally, a flexible Synchronous Serial Controller (SSC) is available. The SSC provides easy access to serial communication protocols and audio standards like I2S.

The AT32UC3A3/A4 includes a powerful External Bus Interface to interface all standard memory device like SRAM, SDRAM, NAND Flash or parallel interfaces like LCD Module.

The peripheral set includes a High Speed MCI for SDIO/SD/MMC and a hardware encryption module based on AES algorithm.

The device embeds a 10-bit ADC and a Digital Audio bistream DAC.

The Direct Memory Access controller (DMACA) allows high bandwidth data flows between high speed peripherals (USB, External Memories, MMC, SDIO, ...) and through high speed internal features (AES, internal memories).

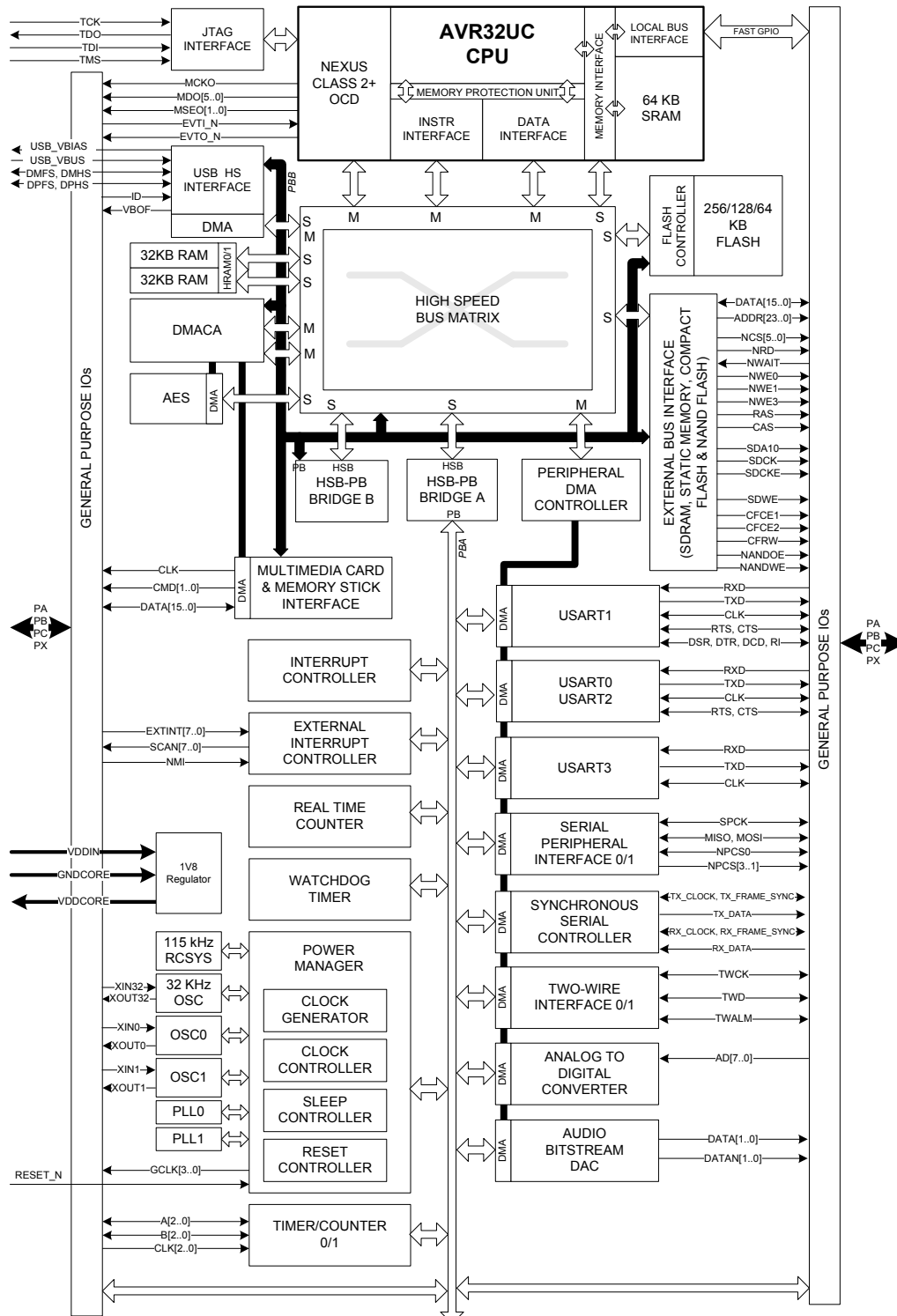
The High-Speed (480MBit/s) USB 2.0 Device and Host interface supports several USB Classes at the same time thanks to the rich Endpoint configuration. The Embedded Host interface allows device like a USB Flash disk or a USB printer to be directly connected to the processor. This peripheral has its own dedicated DMA and is perfect for Mass Storage application.

AT32UC3A3/A4 integrates a class 2+ Nexus 2.0 On-Chip Debug (OCD) System, with non-intrusive real-time trace, full-speed read/write memory access in addition to basic runtime control.

2. Overview

2.1 Block Diagram

Figure 2-1. Block Diagram



2.2 Configuration Summary

The table below lists all AT32UC3A3/A4 memory and package configurations:

Table 2-1. Configuration Summary

| Feature | AT32UC3A3256/128/64 | AT32UC3A4256/128/64 |
|-------------------------|---|------------------------------|
| Flash | 256/128/64 KB | |
| SRAM | 64 KB | |
| HSB RAM | 64 KB | |
| EBI | Full | Nand flash only |
| GPIO | 110 | 70 |
| External Interrupts | 8 | |
| TWI | 2 | |
| USART | 4 | |
| Peripheral DMA Channels | 8 | |
| Generic DMA Channels | 4 | |
| SPI | 2 | |
| MCI slots | 2 MMC/SD slots | 1 MMC/SD slot + 1 SD slot |
| High Speed USB | 1 | |
| AES (S option) | 1 | |
| SSC | 1 | |
| Audio Bitstream DAC | 1 | |
| Timer/Counter Channels | 6 | |
| Watchdog Timer | 1 | |
| Real-Time Clock Timer | 1 | |
| Power Manager | 1 | |
| Oscillators | PLL 80-240 MHz (PLL0/PLL1) Crystal Oscillators 0.4-20 MHz (OSC0/OSC1) Crystal Oscillator 32 KHz (OSC32K) RC Oscillator 115 kHz (RCSYS) | |
| 10-bit ADC | 1 | |
| number of channels | 8 | |
| JTAG | 1 | |
| Max Frequency | 66 MHz | |
| Package | LQFP144, TFBGA144 | VFBGA100 |

3. Package and Pinout

3.1 Package

The device pins are multiplexed with peripheral functions as described in the Peripheral Multiplexing on I/O Line section.

Figure 3-1. TFBGA144 Pinout (top view)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|------|-------|-------|------|-------|-------|-------|-------|-----------|---------|---------|----------|
| A | PX40 | PB00 | PA28 | PA27 | PB03 | PA29 | PC02 | PC04 | PC05 | DPHS | DMHS | USB_VBUS |
| B | PX10 | PB11 | PA31 | PB02 | VDDIO | PB04 | PC03 | VDDIO | USB_VBIAS | DMFS | GNDPLL | PA09 |
| C | PX09 | PX35 | GNDIO | PB01 | PX16 | PX13 | PA30 | PB08 | DPFS | GNDCORE | PA08 | PA10 |
| D | PX08 | PX37 | PX36 | PX47 | PX19 | PX12 | PB10 | PA02 | PA26 | PA11 | PB07 | PB06 |
| E | PX38 | VDDIO | PX54 | PX53 | VDDIO | PX15 | PB09 | VDDIN | PA25 | PA07 | VDDCORE | PA12 |
| F | PX39 | PX07 | PX06 | PX49 | PX48 | GNDIO | GNDIO | PA06 | PA04 | PA05 | PA13 | PA16 |
| G | PX00 | PX05 | PX59 | PX50 | PX51 | GNDIO | GNDIO | PA23 | PA24 | PA03 | PA00 | PA01 |
| H | PX01 | VDDIO | PX58 | PX57 | VDDIO | PC01 | PA17 | VDDIO | PA21 | PA22 | VDDANA | PB05 |
| J | PX04 | PX02 | PX34 | PX56 | PX55 | PA14 | PA15 | PA19 | PA20 | TMS | TDO | RESET_N |
| K | PX03 | PX44 | GNDIO | PX46 | PC00 | PX17 | PX52 | PA18 | PX27 | GNDIO | PX29 | TCK |
| L | PX11 | GNDIO | PX45 | PX20 | VDDIO | PX18 | PX43 | VDDIN | PX26 | PX28 | GNDANA | TDI |
| M | PX22 | PX41 | PX42 | PX14 | PX21 | PX23 | PX24 | PX25 | PX32 | PX31 | PX30 | PX33 |



Figure 3-3. VFBGA100 Pinout (top view)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|------------------------------|-------|-------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|
| A | PA28 | PA27 | PB04 | PA30 | PC02 | PC03 | PC05 | DPHS | DMHS | USB_VBUS |
| B | PB00 | PB01 | PB02 | PA29 | VDDIO | VDDIO | PC04 | DPFS | DMFS | GNDPLL |
| C | PB11 | PA31 | GNDIO | PB03 | PB09 | PB08 | USB_VBIAS | GNDIO | PA11 | PA10 |
| D | PX12 | PX10 | PX13 | PX16/ PX53 ⁽¹⁾ | PB10 | PB07 | PB06 | PA09 | VDDIN | VDDIN |
| E | PA02/ PX47 ⁽¹⁾ | GNDIO | PX08 | PX09 | VDDIO | GNDIO | PA16 | PA06/ PA13 ⁽¹⁾ | PA04 | VDDCORE |
| F | PX19/ PX59 ⁽¹⁾ | VDDIO | PX06 | PX07 | GNDIO | VDDIO | PA26/ PB05 ⁽¹⁾ | PA08 | PA03 | GNDCORE |
| G | PX05 | PX01 | PX02 | PX00 | PX30 | PA23/ PX46 ⁽¹⁾ | PA12/ PA25 ⁽¹⁾ | PA00/ PA18 ⁽¹⁾ | PA05 | PA01/ PA17 ⁽¹⁾ |
| H | PX04 | PX21 | GNDIO | PX25 | PX31 | PA22/ PX20 ⁽¹⁾ | TMS | GNDANA | PA20/ PX18 ⁽¹⁾ | PA07/ PA19 ⁽¹⁾ |
| J | PX03 | PX24 | PX26 | PX29 | VDDIO | VDDANA | PA15/ PX45 ⁽¹⁾ | TDO | RESET_N | PA24/ PX17 ⁽¹⁾ |
| K | PX23 | PX27 | PX28 | PX15/ PX32 ⁽¹⁾ | PC00/ PX14 ⁽¹⁾ | PC01 | PA14/ PX11 ⁽¹⁾ | TDI | TCK | PA21/ PX22 ⁽¹⁾ |

Note: 1. Those balls are physically connected to 2 GPIOs. Software must managed carrefully the GPIO configuration to avoid electrical conflict

3.2 Peripheral Multiplexing on I/O lines

3.2.1 Multiplexed Signals

Each GPIO line can be assigned to one of the peripheral functions. The following table describes the peripheral signals multiplexed to the GPIO lines.

Table 3-1. GPIO Controller Function Multiplexing

| BGA 144 | QFP 144 | BGA 100 | PIN | G P I O | Supply | PIN Type (2) | GPIO function | | | |
|------------|------------|--------------------|------|------------------|------------|--------------------|----------------|---------------------|------------------|------------|
| | | | | | | | A | B | C | D |
| G11 | 122 | G8 ⁽¹⁾ | PA00 | 0 | VDDIO | x3 | USART0 - RTS | TC0 - CLK1 | SPI1 - NPCS[3] | |
| G12 | 123 | G10 ⁽¹⁾ | PA01 | 1 | VDDIO | x1 | USART0 - CTS | TC0 - A1 | USART2 - RTS | |
| D8 | 15 | E1 ⁽¹⁾ | PA02 | 2 | VDDIO | x1 | USART0 - CLK | TC0 - B1 | SPI0 - NPCS[0] | |
| G10 | 125 | F9 | PA03 | 3 | VDDIO | x1 | USART0 - RXD | EIC - EXTINT[4] | ABDAC - DATA[0] | |
| F9 | 126 | E9 | PA04 | 4 | VDDIO | x1 | USART0 - TXD | EIC - EXTINT[5] | ABDAC - DATAN[0] | |
| F10 | 124 | G9 | PA05 | 5 | VDDIO | x1 | USART1 - RXD | TC1 - CLK0 | USB - ID | |
| F8 | 127 | E8 ⁽¹⁾ | PA06 | 6 | VDDIO | x1 | USART1 - TXD | TC1 - CLK1 | USB - VBOF | |
| E10 | 133 | H10 ⁽¹⁾ | PA07 | 7 | VDDIO | x1 | SPI0 - NPCS[3] | ABDAC - DATAN[0] | USART1 - CLK | |
| C11 | 137 | F8 | PA08 | 8 | VDDIO | x3 | SPI0 - SPCK | ABDAC - DATA[0] | TC1 - B1 | |
| B12 | 139 | D8 | PA09 | 9 | VDDIO | x2 | SPI0 - NPCS[0] | EIC - EXTINT[6] | TC1 - A1 | |
| C12 | 138 | C10 | PA10 | 10 | VDDIO | x2 | SPI0 - MOSI | USB - VBOF | TC1 - B0 | |
| D10 | 136 | C9 | PA11 | 11 | VDDIO | x2 | SPI0 - MISO | USB - ID | TC1 - A2 | |
| E12 | 132 | G7 ⁽¹⁾ | PA12 | 12 | VDDIO | x1 | USART1 - CTS | SPI0 - NPCS[2] | TC1 - A0 | |
| F11 | 129 | E8 ⁽¹⁾ | PA13 | 13 | VDDIO | x1 | USART1 - RTS | SPI0 - NPCS[1] | EIC - EXTINT[7] | |
| J6 | 100 | K7 ⁽¹⁾ | PA14 | 14 | VDDIO | x1 | SPI0 - NPCS[1] | TWIMS0 - TWALM | TWIMS1 - TWCK | |
| J7 | 101 | J7 ⁽¹⁾ | PA15 | 15 | VDDIO | x1 | MCI - CMD[1] | SPI1 - SPCK | TWIMS1 - TWD | |
| F12 | 128 | E7 | PA16 | 16 | VDDIO | x1 | MCI - DATA[11] | SPI1 - MOSI | TC1 - CLK2 | |
| H7 | 116 | G10 ⁽¹⁾ | PA17 | 17 | VDDAN A | x1 | MCI - DATA[10] | SPI1 - NPCS[1] | ADC - AD[7] | |
| K8 | 115 | G8 ⁽¹⁾ | PA18 | 18 | VDDAN A | x1 | MCI - DATA[9] | SPI1 - NPCS[2] | ADC - AD[6] | |
| J8 | 114 | H10 ⁽¹⁾ | PA19 | 19 | VDDAN A | x1 | MCI - DATA[8] | SPI1 - MISO | ADC - AD[5] | |
| J9 | 113 | H9 ⁽¹⁾ | PA20 | 20 | VDDAN A | x1 | EIC - NMI | SSC - RX_FRAME_SYNC | ADC - AD[4] | |
| H9 | 109 | K10 ⁽¹⁾ | PA21 | 21 | VDDAN A | x1 | ADC - AD[0] | EIC - EXTINT[0] | USB - ID | |
| H10 | 110 | H6 ⁽¹⁾ | PA22 | 22 | VDDAN A | x1 | ADC - AD[1] | EIC - EXTINT[1] | USB - VBOF | |
| G8 | 111 | G6 ⁽¹⁾ | PA23 | 23 | VDDAN A | x1 | ADC - AD[2] | EIC - EXTINT[2] | ABDAC - DATA[1] | |
| G9 | 112 | J10 ⁽¹⁾ | PA24 | 24 | VDDAN A | x1 | ADC - AD[3] | EIC - EXTINT[3] | ABDAC - DATAN[1] | |
| E9 | 119 | G7 ⁽¹⁾ | PA25 | 25 | VDDIO | x1 | TWIMS0 - TWD | TWIMS1 - TWALM | USART1 - DCD | |
| D9 | 120 | F7 ⁽¹⁾ | PA26 | 26 | VDDIO | x1 | TWIMS0 - TWCK | USART2 - CTS | USART1 - DSR | |
| A4 | 26 | A2 | PA27 | 27 | VDDIO | x2 | MCI - CLK | SSC - RX_DATA | USART3 - RTS | MSI - SCLK |

Table 3-1. GPIO Controller Function Multiplexing

| BGA 144 | QFP 144 | BGA 100 | PIN | G P I O | Supply | PIN Type (2) | GPIO function | | | |
|------------|------------|-------------------|------|------------------|--------|--------------------|----------------|---------------------|------------------|---------------|
| | | | | | | | A | B | C | D |
| A3 | 28 | A1 | PA28 | 28 | VDDIO | x1 | MCI - CMD[0] | SSC - RX_CLOCK | USART3 - CTS | MSI - BS |
| A6 | 23 | B4 | PA29 | 29 | VDDIO | x1 | MCI - DATA[0] | USART3 - TXD | TC0 - CLK0 | MSI - DATA[0] |
| C7 | 14 | A4 | PA30 | 30 | VDDIO | x1 | MCI - DATA[1] | USART3 - CLK | DMAC - DMAACK[0] | MSI - DATA[1] |
| B3 | 29 | C2 | PA31 | 31 | VDDIO | x1 | MCI - DATA[2] | USART2 - RXD | DMAC - DMARQ[0] | MSI - DATA[2] |
| A2 | 30 | B1 | PB00 | 32 | VDDIO | x1 | MCI - DATA[3] | USART2 - TXD | ADC - TRIGGER | MSI - DATA[3] |
| C4 | 27 | B2 | PB01 | 33 | VDDIO | x1 | MCI - DATA[4] | ABDAC - DATA[1] | EIC - SCAN[0] | MSI - INS |
| B4 | 25 | B3 | PB02 | 34 | VDDIO | x1 | MCI - DATA[5] | ABDAC - DATAN[1] | EIC - SCAN[1] | |
| A5 | 24 | C4 | PB03 | 35 | VDDIO | x1 | MCI - DATA[6] | USART2 - CLK | EIC - SCAN[2] | |
| B6 | 22 | A3 | PB04 | 36 | VDDIO | x1 | MCI - DATA[7] | USART3 - RXD | EIC - SCAN[3] | |
| H12 | 121 | F7 ⁽¹⁾ | PB05 | 37 | VDDIO | x3 | USB - ID | TC0 - A0 | EIC - SCAN[4] | |
| D12 | 134 | D7 | PB06 | 38 | VDDIO | x1 | USB - VBOF | TC0 - B0 | EIC - SCAN[5] | |
| D11 | 135 | D6 | PB07 | 39 | VDDIO | x3 | SPI1 - SPCK | SSC - TX_CLOCK | EIC - SCAN[6] | |
| C8 | 11 | C6 | PB08 | 40 | VDDIO | x2 | SPI1 - MISO | SSC - TX_DATA | EIC - SCAN[7] | |
| E7 | 17 | C5 | PB09 | 41 | VDDIO | x2 | SPI1 - NPCS[0] | SSC - RX_DATA | EBI - NCS[4] | |
| D7 | 16 | D5 | PB10 | 42 | VDDIO | x2 | SPI1 - MOSI | SSC - RX_FRAME_SYNC | EBI - NCS[5] | |
| B2 | 31 | C1 | PB11 | 43 | VDDIO | x1 | USART1 - RXD | SSC - TX_FRAME_SYNC | PM - GCLK[1] | |
| K5 | 98 | K5 ⁽¹⁾ | PC00 | 45 | VDDIO | x1 | | | | |
| H6 | 99 | K6 | PC01 | 46 | VDDIO | x1 | | | | |
| A7 | 18 | A5 | PC02 | 47 | VDDIO | x1 | | | | |
| B7 | 19 | A6 | PC03 | 48 | VDDIO | x1 | | | | |
| A8 | 13 | B7 | PC04 | 49 | VDDIO | x1 | | | | |
| A9 | 12 | A7 | PC05 | 50 | VDDIO | x1 | | | | |
| G1 | 55 | G4 | PX00 | 51 | VDDIO | x2 | EBI - DATA[10] | USART0 - RXD | USART1 - RI | |
| H1 | 59 | G2 | PX01 | 52 | VDDIO | x2 | EBI - DATA[9] | USART0 - TXD | USART1 - DTR | |
| J2 | 62 | G3 | PX02 | 53 | VDDIO | x2 | EBI - DATA[8] | USART0 - CTS | PM - GCLK[0] | |
| K1 | 63 | J1 | PX03 | 54 | VDDIO | x2 | EBI - DATA[7] | USART0 - RTS | | |
| J1 | 60 | H1 | PX04 | 55 | VDDIO | x2 | EBI - DATA[6] | USART1 - RXD | | |
| G2 | 58 | G1 | PX05 | 56 | VDDIO | x2 | EBI - DATA[5] | USART1 - TXD | | |
| F3 | 53 | F3 | PX06 | 57 | VDDIO | x2 | EBI - DATA[4] | USART1 - CTS | | |
| F2 | 54 | F4 | PX07 | 58 | VDDIO | x2 | EBI - DATA[3] | USART1 - RTS | | |
| D1 | 50 | E3 | PX08 | 59 | VDDIO | x2 | EBI - DATA[2] | USART3 - RXD | | |
| C1 | 49 | E4 | PX09 | 60 | VDDIO | x2 | EBI - DATA[1] | USART3 - TXD | | |
| B1 | 37 | D2 | PX10 | 61 | VDDIO | x2 | EBI - DATA[0] | USART2 - RXD | | |
| L1 | 67 | K7 ⁽¹⁾ | PX11 | 62 | VDDIO | x2 | EBI - NWE1 | USART2 - TXD | | |
| D6 | 34 | D1 | PX12 | 63 | VDDIO | x2 | EBI - NWE0 | USART2 - CTS | MCI - CLK | |
| C6 | 33 | D3 | PX13 | 64 | VDDIO | x2 | EBI - NRD | USART2 - RTS | MCI - CLK | |

Table 3-1. GPIO Controller Function Multiplexing

| BGA 144 | QFP 144 | BGA 100 | PIN | G P I O | Supply | PIN Type (2) | GPIO function | | | |
|------------|------------|--------------------|------|------------------|--------|--------------------|----------------|-------------------|---------------------|---|
| | | | | | | | A | B | C | D |
| M4 | 68 | K5 ⁽¹⁾ | PX14 | 65 | VDDIO | x2 | EBI - NCS[1] | | TC0 - A0 | |
| E6 | 40 | K4 ⁽¹⁾ | PX15 | 66 | VDDIO | x2 | EBI - ADDR[19] | USART3 - RTS | TC0 - B0 | |
| C5 | 32 | D4 ⁽¹⁾ | PX16 | 67 | VDDIO | x2 | EBI - ADDR[18] | USART3 - CTS | TC0 - A1 | |
| K6 | 83 | J10 ⁽¹⁾ | PX17 | 68 | VDDIO | x2 | EBI - ADDR[17] | DMACA - DMARQ[1] | TC0 - B1 | |
| L6 | 84 | H9 ⁽¹⁾ | PX18 | 69 | VDDIO | x2 | EBI - ADDR[16] | DMACA - DMAACK[1] | TC0 - A2 | |
| D5 | 35 | F1 ⁽¹⁾ | PX19 | 70 | VDDIO | x2 | EBI - ADDR[15] | EIC - SCAN[0] | TC0 - B2 | |
| L4 | 73 | H6 ⁽¹⁾ | PX20 | 71 | VDDIO | x2 | EBI - ADDR[14] | EIC - SCAN[1] | TC0 - CLK0 | |
| M5 | 80 | H2 | PX21 | 72 | VDDIO | x2 | EBI - ADDR[13] | EIC - SCAN[2] | TC0 - CLK1 | |
| M1 | 72 | K10 ⁽¹⁾ | PX22 | 73 | VDDIO | x2 | EBI - ADDR[12] | EIC - SCAN[3] | TC0 - CLK2 | |
| M6 | 85 | K1 | PX23 | 74 | VDDIO | x2 | EBI - ADDR[11] | EIC - SCAN[4] | SSC - TX_CLOCK | |
| M7 | 86 | J2 | PX24 | 75 | VDDIO | x2 | EBI - ADDR[10] | EIC - SCAN[5] | SSC - TX_DATA | |
| M8 | 92 | H4 | PX25 | 76 | VDDIO | x2 | EBI - ADDR[9] | EIC - SCAN[6] | SSC - RX_DATA | |
| L9 | 90 | J3 | PX26 | 77 | VDDIO | x2 | EBI - ADDR[8] | EIC - SCAN[7] | SSC - RX_FRAME_SYNC | |
| K9 | 89 | K2 | PX27 | 78 | VDDIO | x2 | EBI - ADDR[7] | SPI0 - MISO | SSC - TX_FRAME_SYNC | |
| L10 | 91 | K3 | PX28 | 79 | VDDIO | x2 | EBI - ADDR[6] | SPI0 - MOSI | SSC - RX_CLOCK | |
| K11 | 94 | J4 | PX29 | 80 | VDDIO | x2 | EBI - ADDR[5] | SPI0 - SPCK | | |
| M11 | 96 | G5 | PX30 | 81 | VDDIO | x2 | EBI - ADDR[4] | SPI0 - NPCS[0] | | |
| M10 | 97 | H5 | PX31 | 82 | VDDIO | x2 | EBI - ADDR[3] | SPI0 - NPCS[1] | | |
| M9 | 93 | K4 ⁽¹⁾ | PX32 | 83 | VDDIO | x2 | EBI - ADDR[2] | SPI0 - NPCS[2] | | |
| M12 | 95 | | PX33 | 84 | VDDIO | x2 | EBI - ADDR[1] | SPI0 - NPCS[3] | | |
| J3 | 61 | | PX34 | 85 | VDDIO | x2 | EBI - ADDR[0] | SPI1 - MISO | PM - GCLK[0] | |
| C2 | 38 | | PX35 | 86 | VDDIO | x2 | EBI - DATA[15] | SPI1 - MOSI | PM - GCLK[1] | |
| D3 | 44 | | PX36 | 87 | VDDIO | x2 | EBI - DATA[14] | SPI1 - SPCK | PM - GCLK[2] | |
| D2 | 45 | | PX37 | 88 | VDDIO | x2 | EBI - DATA[13] | SPI1 - NPCS[0] | PM - GCLK[3] | |
| E1 | 51 | | PX38 | 89 | VDDIO | x2 | EBI - DATA[12] | SPI1 - NPCS[1] | USART1 - DCD | |
| F1 | 52 | | PX39 | 90 | VDDIO | x2 | EBI - DATA[11] | SPI1 - NPCS[2] | USART1 - DSR | |
| A1 | 36 | | PX40 | 91 | VDDIO | x2 | | MCI - CLK | | |
| M2 | 71 | | PX41 | 92 | VDDIO | x2 | EBI - CAS | | | |
| M3 | 69 | | PX42 | 93 | VDDIO | x2 | EBI - RAS | | | |
| L7 | 88 | | PX43 | 94 | VDDIO | x2 | EBI - SDA10 | USART1 - RI | | |
| K2 | 66 | | PX44 | 95 | VDDIO | x2 | EBI - SDWE | USART1 - DTR | | |
| L3 | 70 | J7 ⁽¹⁾ | PX45 | 96 | VDDIO | x3 | EBI - SDCK | | | |
| K4 | 74 | G6 ⁽¹⁾ | PX46 | 97 | VDDIO | x2 | EBI - SDCKE | | | |
| D4 | 39 | E1 ⁽¹⁾ | PX47 | 98 | VDDIO | x2 | EBI - NANDOE | ADC - TRIGGER | MCI - DATA[11] | |
| F5 | 41 | | PX48 | 99 | VDDIO | x2 | EBI - ADDR[23] | USB - VBOF | MCI - DATA[10] | |
| F4 | 43 | | PX49 | 100 | VDDIO | x2 | EBI - CFRNW | USB - ID | MCI - DATA[9] | |
| G4 | 75 | | PX50 | 101 | VDDIO | x2 | EBI - CFCE2 | TC1 - B2 | MCI - DATA[8] | |
| G5 | 77 | | PX51 | 102 | VDDIO | x2 | EBI - CFCE1 | DMACA - DMAACK[0] | MCI - DATA[15] | |

Table 3-1. GPIO Controller Function Multiplexing

| BGA 144 | QFP 144 | BGA 100 | PIN | G P I O | Supply | PIN Type (2) | GPIO function | | | |
|------------|------------|-------------------|------|------------------|--------|--------------------|----------------|------------------|----------------|---|
| | | | | | | | A | B | C | D |
| K7 | 87 | | PX52 | 103 | VDDIO | x2 | EBI - NCS[3] | DMACA - DMARQ[0] | MCI - DATA[14] | |
| E4 | 42 | D4 ⁽¹⁾ | PX53 | 104 | VDDIO | x2 | EBI - NCS[2] | | MCI - DATA[13] | |
| E3 | 46 | | PX54 | 105 | VDDIO | x2 | EBI - NWAIT | USART3 - TXD | MCI - DATA[12] | |
| J5 | 79 | | PX55 | 106 | VDDIO | x2 | EBI - ADDR[22] | EIC - SCAN[3] | USART2 - RXD | |
| J4 | 78 | | PX56 | 107 | VDDIO | x2 | EBI - ADDR[21] | EIC - SCAN[2] | USART2 - TXD | |
| H4 | 76 | | PX57 | 108 | VDDIO | x2 | EBI - ADDR[20] | EIC - SCAN[1] | USART3 - RXD | |
| H3 | 57 | | PX58 | 109 | VDDIO | x2 | EBI - NCS[0] | EIC - SCAN[0] | USART3 - TXD | |
| G3 | 56 | F1 ⁽¹⁾ | PX59 | 110 | VDDIO | x2 | EBI - NANDWE | | MCI - CMD[1] | |

- Note:
1. Those balls are physically connected to 2 GPIOs. Software must managed carefully the GPIO configuration to avoid electrical conflict.
 2. Refer to ["Electrical Characteristics" on page 40](#) for a description of the electrical properties of the pad types used.
 3. GPIO 44 is physically implemented in silicon but must be kept unused and configured in input mode.

3.2.2 Peripheral Functions

Each GPIO line can be assigned to one of several peripheral functions. The following table describes how the various peripheral functions are selected. The last listed function has priority in case multiple functions are enabled on the same pin.

Table 3-2. Peripheral Functions

| Function | Description |
|---------------------------------------|---|
| GPIO Controller Function multiplexing | GPIO and GPIO peripheral selection A to D |
| Nexus OCD AUX port connections | OCD trace system |
| JTAG port connections | JTAG debug port |
| Oscillators | OSC0, OSC1, OSC32 |

3.2.3 Oscillator Pinout

The oscillators are not mapped to the normal GPIO functions and their muxings are controlled by registers in the Power Manager (PM). Please refer to the PM chapter for more information about this.

Table 3-3. Oscillator Pinout

| TFBGA144 | QFP144 | VFBGA100 | Pin name | Oscillator pin |
|----------|--------|-------------------|----------|----------------|
| A7 | 18 | A5 | PC02 | XIN0 |
| B7 | 19 | A6 | PC03 | XOUT0 |
| A8 | 13 | B7 | PC04 | XIN1 |
| A9 | 12 | A7 | PC05 | XOUT1 |
| K5 | 98 | K5 ⁽¹⁾ | PC00 | XIN32 |
| H6 | 99 | K6 | PC01 | XOUT32 |

Note: 1. This ball is physically connected to 2 GPIOs. Software must managed carefully the GPIO configuration to avoid electrical conflict

3.2.4 JTAG port connections

Table 3-4. JTAG Pinout

| TFBGA144 | QFP144 | VFBGA100 | Pin name | JTAG pin |
|----------|--------|----------|----------|----------|
| K12 | 107 | K9 | TCK | TCK |
| L12 | 108 | K8 | TDI | TDI |
| J11 | 105 | J8 | TDO | TDO |
| J10 | 104 | H7 | TMS | TMS |

3.2.5 Nexus OCD AUX port connections

If the OCD trace system is enabled, the trace system will take control over a number of pins, irrespective of the GPIO configuration. Three different OCD trace pin mappings are possible, depending on the configuration of the OCD AXS register. For details, see the AVR32 UC Technical Reference Manual.

Table 3-5. Nexus OCD AUX port connections

| Pin | AXS=0 | AXS=1 | AXS=2 |
|---------|-------|-------|-------|
| EVTI_N | PB05 | PA08 | PX00 |
| MDO[5] | PA00 | PX56 | PX06 |
| MDO[4] | PA01 | PX57 | PX05 |
| MDO[3] | PA03 | PX58 | PX04 |
| MDO[2] | PA16 | PA24 | PX03 |
| MDO[1] | PA13 | PA23 | PX02 |
| MDO[0] | PA12 | PA22 | PX01 |
| MSEO[1] | PA10 | PA07 | PX08 |
| MSEO[0] | PA11 | PX55 | PX07 |
| MCKO | PB07 | PX00 | PB09 |
| EVTO_N | PB06 | PB06 | PB06 |

3.3 Signal Descriptions

The following table gives details on signal name classified by peripheral.

Table 3-6. Signal Description List

| Signal Name | Function | Type | Active Level | Comments |
|---------------------------------------|---|--------------|--------------|----------------|
| Power | | | | |
| VDDIO | I/O Power Supply | Power | | 3.0 to 3.6V |
| VDDANA | Analog Power Supply | Power | | 3.0 to 3.6V |
| VDDIN | Voltage Regulator Input Supply | Power | | 3.0 to 3.6V |
| VDDCORE | Voltage Regulator Output for Digital Supply | Power Output | | 1.65 to 1.95 V |
| GNDANA | Analog Ground | Ground | | |
| GNDIO | I/O Ground | Ground | | |
| GNDCORE | Digital Ground | Ground | | |
| GNDPLL | PLL Ground | Ground | | |
| Clocks, Oscillators, and PLL's | | | | |
| XIN0, XIN1, XIN32 | Crystal 0, 1, 32 Input | Analog | | |
| XOUT0, XOUT1, XOUT32 | Crystal 0, 1, 32 Output | Analog | | |
| JTAG | | | | |
| TCK | Test Clock | Input | | |
| TDI | Test Data In | Input | | |
| TDO | Test Data Out | Output | | |
| TMS | Test Mode Select | Input | | |
| Auxiliary Port - AUX | | | | |
| MCKO | Trace Data Output Clock | Output | | |
| MDO[5:0] | Trace Data Output | Output | | |
| MSEO[1:0] | Trace Frame Control | Output | | |
| EVTI_N | Event In | Input | Low | |
| EVTO_N | Event Out | Output | Low | |
| Power Manager - PM | | | | |
| GCLK[3:0] | Generic Clock Pins | Output | | |

Table 3-6. Signal Description List

| Signal Name | Function | Type | Active Level | Comments |
|--|-------------------------------------|--------|--------------|----------|
| RESET_N | Reset Pin | Input | Low | |
| DMA Controller - DMACA (optional) | | | | |
| DMAACK[1:0] | DMA Acknowledge | Output | | |
| DMARQ[1:0] | DMA Requests | Input | | |
| External Interrupt Controller - EIC | | | | |
| EXTINT[7:0] | External Interrupt Pins | Input | | |
| SCAN[7:0] | Keypad Scan Pins | Output | | |
| NMI | Non-Maskable Interrupt Pin | Input | Low | |
| General Purpose Input/Output pin - GPIOA, GPIOB, GPIOC, GPIOX | | | | |
| PA[31:0] | Parallel I/O Controller GPIO port A | I/O | | |
| PB[11:0] | Parallel I/O Controller GPIO port B | I/O | | |
| PC[5:0] | Parallel I/O Controller GPIO port C | I/O | | |
| PX[59:0] | Parallel I/O Controller GPIO port X | I/O | | |
| External Bus Interface - EBI | | | | |
| ADDR[23:0] | Address Bus | Output | | |
| CAS | Column Signal | Output | Low | |
| CFCE1 | Compact Flash 1 Chip Enable | Output | Low | |
| CFCE2 | Compact Flash 2 Chip Enable | Output | Low | |
| CFRNW | Compact Flash Read Not Write | Output | | |
| DATA[15:0] | Data Bus | I/O | | |
| NANDOE | NAND Flash Output Enable | Output | Low | |
| NANDWE | NAND Flash Write Enable | Output | Low | |
| NCS[5:0] | Chip Select | Output | Low | |
| NRD | Read Signal | Output | Low | |
| NWAIT | External Wait Signal | Input | Low | |
| NWE0 | Write Enable 0 | Output | Low | |
| NWE1 | Write Enable 1 | Output | Low | |
| RAS | Row Signal | Output | Low | |

Table 3-6. Signal Description List

| Signal Name | Function | Type | Active Level | Comments |
|---|----------------------------|--------|--------------|----------|
| SDA10 | SDRAM Address 10 Line | Output | | |
| SDCK | SDRAM Clock | Output | | |
| SDCKE | SDRAM Clock Enable | Output | | |
| SDWE | SDRAM Write Enable | Output | Low | |
| MultiMedia Card Interface - MCI | | | | |
| CLK | Multimedia Card Clock | Output | | |
| CMD[1:0] | Multimedia Card Command | I/O | | |
| DATA[15:0] | Multimedia Card Data | I/O | | |
| Memory Stick Interface - MSI | | | | |
| SCLK | Memory Stick Clock | Output | | |
| BS | Memory Stick Command | I/O | | |
| DATA[3:0] | Multimedia Card Data | I/O | | |
| Serial Peripheral Interface - SPI0, SPI1 | | | | |
| MISO | Master In Slave Out | I/O | | |
| MOSI | Master Out Slave In | I/O | | |
| NPCS[3:0] | SPI Peripheral Chip Select | I/O | Low | |
| SPCK | Clock | Output | | |
| Synchronous Serial Controller - SSC | | | | |
| RX_CLOCK | SSC Receive Clock | I/O | | |
| RX_DATA | SSC Receive Data | Input | | |
| RX_FRAME_SYNC | SSC Receive Frame Sync | I/O | | |
| TX_CLOCK | SSC Transmit Clock | I/O | | |
| TX_DATA | SSC Transmit Data | Output | | |
| TX_FRAME_SYNC | SSC Transmit Frame Sync | I/O | | |
| Timer/Counter - TC0, TC1 | | | | |
| A0 | Channel 0 Line A | I/O | | |
| A1 | Channel 1 Line A | I/O | | |
| A2 | Channel 2 Line A | I/O | | |

Table 3-6. Signal Description List

| Signal Name | Function | Type | Active Level | Comments |
|---|--------------------------------|--------------|--------------|-------------|
| B0 | Channel 0 Line B | I/O | | |
| B1 | Channel 1 Line B | I/O | | |
| B2 | Channel 2 Line B | I/O | | |
| CLK0 | Channel 0 External Clock Input | Input | | |
| CLK1 | Channel 1 External Clock Input | Input | | |
| CLK2 | Channel 2 External Clock Input | Input | | |
| Two-wire Interface - TWI0, TWI1 | | | | |
| TWCK | Serial Clock | I/O | | |
| TWD | Serial Data | I/O | | |
| TWALM | SMBALERT signal | I/O | | |
| Universal Synchronous Asynchronous Receiver Transmitter - USART0, USART1, USART2, USART3 | | | | |
| CLK | Clock | I/O | | |
| CTS | Clear To Send | Input | | |
| DCD | Data Carrier Detect | | | Only USART1 |
| DSR | Data Set Ready | | | Only USART1 |
| DTR | Data Terminal Ready | | | Only USART1 |
| RI | Ring Indicator | | | Only USART1 |
| RTS | Request To Send | Output | | |
| RXD | Receive Data | Input | | |
| TXD | Transmit Data | Output | | |
| Analog to Digital Converter - ADC | | | | |
| AD0 - AD7 | Analog input pins | Analog input | | |
| Audio Bitstream DAC (ABDAC) | | | | |
| DATA0-DATA1 | D/A Data out | Output | | |
| DATAN0-DATAN1 | D/A Data inverted out | Output | | |
| Universal Serial Bus Device - USB | | | | |
| DMFS | USB Full Speed Data - | Analog | | |
| DPFS | USB Full Speed Data + | Analog | | |

Table 3-6. Signal Description List

| Signal Name | Function | Type | Active Level | Comments |
|-------------|--|--------|--------------|--|
| DMHS | USB High Speed Data - | Analog | | |
| DPHS | USB High Speed Data + | Analog | | |
| USB_VBIAS | USB VBIAS reference | Analog | | Connect to the ground through a 6810 ohms (+/- 1%) resistor in parallel with a 10pf capacitor. If USB hi-speed feature is not required, leave this pin unconnected to save power |
| USB_VBUS | USB VBUS signal | Output | | |
| VBOF | USB VBUS on/off bus power control port | Output | | |
| ID | ID Pin fo the USB bus | Input | | |

3.4 I/O Line Considerations

3.4.1 JTAG Pins

TMS and TDI pins have pull-up resistors. TDO pin is an output, driven at up to VDDIO, and has no pull-up resistor.

3.4.2 RESET_N Pin

The RESET_N pin is a schmitt input and integrates a permanent pull-up resistor to VDDIO. As the product integrates a power-on reset cell, the RESET_N pin can be left unconnected in case no reset from the system needs to be applied to the product.

3.4.3 TWI Pins

When these pins are used for TWI, the pins are open-drain outputs with slew-rate limitation and inputs with inputs with spike filtering. When used as GPIO pins or used for other peripherals, the pins have the same characteristics as other GPIO pins.

3.4.4 GPIO Pins

All the I/O lines integrate a programmable pull-up resistor. Programming of this pull-up resistor is performed independently for each I/O line through the I/O Controller. After reset, I/O lines default as inputs with pull-up resistors disabled, except when indicated otherwise in the column “Reset State” of the I/O Controller multiplexing tables.

3.5 Power Considerations

3.5.1 Power Supplies

The AT32UC3A3 has several types of power supply pins:

- **VDDIO:** Powers I/O lines. Voltage is 3.3V nominal
- **VDDANA:** Powers the ADC. Voltage is 3.3V nominal
- **VDDIN:** Input voltage for the voltage regulator. Voltage is 3.3V nominal
- **VDDCORE:** Output voltage from regulator for filtering purpose and provides the supply to the core, memories, and peripherals. Voltage is 1.8V nominal

The ground pin GNDCORE is common to VDDCORE and VDDIN. The ground pin for VDDANA is GNDANA. The ground pins for VDDIO are GNDIO.

Refer to Electrical Characteristics chapter for power consumption on the various supply pins.

3.5.2 Voltage Regulator

The AT32UC3A3 embeds a voltage regulator that converts from 3.3V to 1.8V with a load of up to 100 mA. The regulator takes its input voltage from VDDIN, and supplies the output voltage on VDDCORE and powers the core, memories and peripherals.

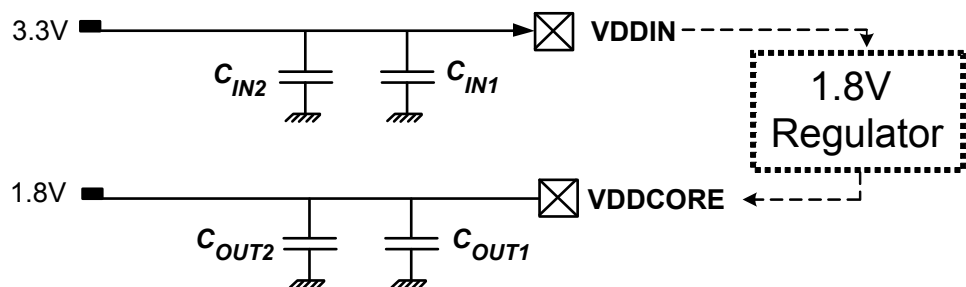
Adequate output supply decoupling is mandatory for VDDCORE to reduce ripple and avoid oscillations.

The best way to achieve this is to use two capacitors in parallel between VDDCORE and GNDCORE:

- One external 470pF (or 1nF) NPO capacitor (COUT1) should be connected as close to the chip as possible.
- One external 2.2μF (or 3.3μF) X7R capacitor (COUT2).

Adequate input supply decoupling is mandatory for VDDIN in order to improve startup stability and reduce source voltage drop.

The input decoupling capacitor should be placed close to the chip, e.g., two capacitors can be used in parallel (1nF NPO and 4.7μF X7R).



For decoupling recommendations for VDDIO and VDDANA please refer to the Schematic checklist.

4. Processor and Architecture

Rev: 1.4.2.0

This chapter gives an overview of the AVR32UC CPU. AVR32UC is an implementation of the AVR32 architecture. A summary of the programming model, instruction set, and MPU is presented. For further details, see the *AVR32 Architecture Manual* and the *AVR32UC Technical Reference Manual*.

4.1 Features

- **32-bit load/store AVR32A RISC architecture**
 - 15 general-purpose 32-bit registers
 - 32-bit Stack Pointer, Program Counter and Link Register reside in register file
 - Fully orthogonal instruction set
 - Privileged and unprivileged modes enabling efficient and secure Operating Systems
 - Innovative instruction set together with variable instruction length ensuring industry leading code density
 - DSP extension with saturating arithmetic, and a wide variety of multiply instructions
- **3-stage pipeline allows one instruction per clock cycle for most instructions**
 - Byte, halfword, word and double word memory access
 - Multiple interrupt priority levels
- **MPU allows for operating systems with memory protection**

4.2 AVR32 Architecture

AVR32 is a high-performance 32-bit RISC microprocessor architecture, designed for cost-sensitive embedded applications, with particular emphasis on low power consumption and high code density. In addition, the instruction set architecture has been tuned to allow a variety of micro-architectures, enabling the AVR32 to be implemented as low-, mid-, or high-performance processors. AVR32 extends the AVR family into the world of 32- and 64-bit applications.

Through a quantitative approach, a large set of industry recognized benchmarks has been compiled and analyzed to achieve the best code density in its class. In addition to lowering the memory requirements, a compact code size also contributes to the core's low power characteristics. The processor supports byte and halfword data types without penalty in code size and performance.

Memory load and store operations are provided for byte, halfword, word, and double word data with automatic sign- or zero extension of halfword and byte data. The C-compiler is closely linked to the architecture and is able to exploit code optimization features, both for size and speed.

In order to reduce code size to a minimum, some instructions have multiple addressing modes. As an example, instructions with immediates often have a compact format with a smaller immediate, and an extended format with a larger immediate. In this way, the compiler is able to use the format giving the smallest code size.

Another feature of the instruction set is that frequently used instructions, like add, have a compact format with two operands as well as an extended format with three operands. The larger format increases performance, allowing an addition and a data move in the same instruction in a single cycle. Load and store instructions have several different formats in order to reduce code size and speed up execution.

The register file is organized as sixteen 32-bit registers and includes the Program Counter, the Link Register, and the Stack Pointer. In addition, register R12 is designed to hold return values from function calls and is used implicitly by some instructions.

4.3 The AVR32UC CPU

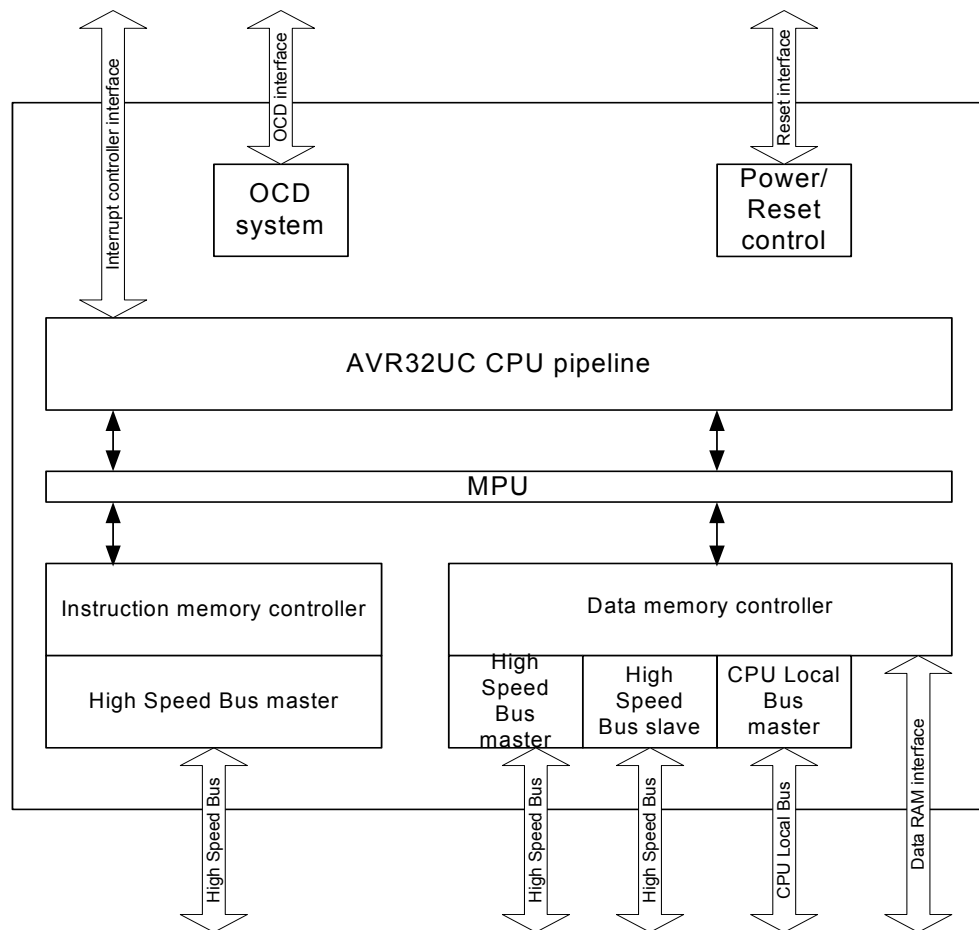
The AVR32UC CPU targets low- and medium-performance applications, and provides an advanced OCD system, no caches, and a Memory Protection Unit (MPU). Java acceleration hardware is not implemented.

AVR32UC provides three memory interfaces, one High Speed Bus master for instruction fetch, one High Speed Bus master for data access, and one High Speed Bus slave interface allowing other bus masters to access data RAMs internal to the CPU. Keeping data RAMs internal to the CPU allows fast access to the RAMs, reduces latency, and guarantees deterministic timing. Also, power consumption is reduced by not needing a full High Speed Bus access for memory accesses. A dedicated data RAM interface is provided for communicating with the internal data RAMs.

A local bus interface is provided for connecting the CPU to device-specific high-speed systems, such as floating-point units and fast GPIO ports. This local bus has to be enabled by writing the LOCEN bit in the CPUCR system register. The local bus is able to transfer data between the CPU and the local bus slave in a single clock cycle. The local bus has a dedicated memory range allocated to it, and data transfers are performed using regular load and store instructions. Details on which devices that are mapped into the local bus space is given in the Memories chapter of this data sheet.

[Figure 4-1 on page 23](#) displays the contents of AVR32UC.

Figure 4-1. Overview of the AVR32UC CPU



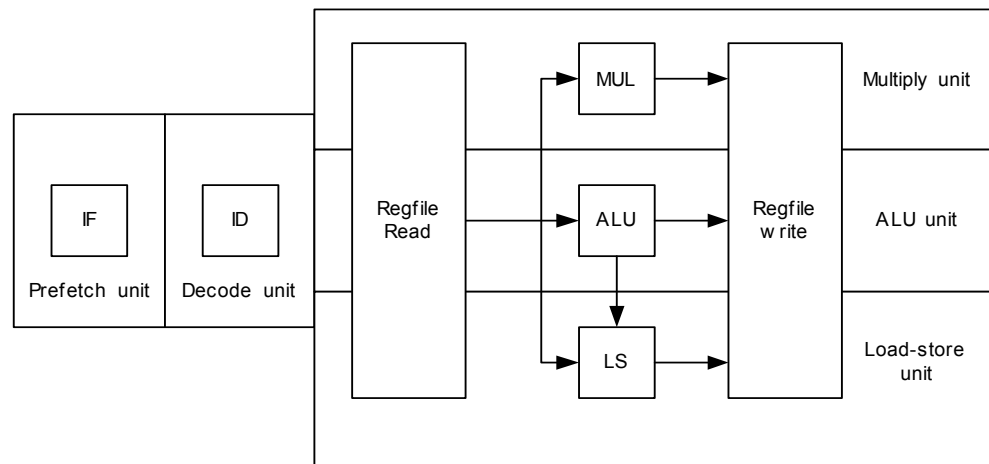
4.3.1 Pipeline Overview

AVR32UC has three pipeline stages, Instruction Fetch (IF), Instruction Decode (ID), and Instruction Execute (EX). The EX stage is split into three parallel subsections, one arithmetic/logic (ALU) section, one multiply (MUL) section, and one load/store (LS) section.

Instructions are issued and complete in order. Certain operations require several clock cycles to complete, and in this case, the instruction resides in the ID and EX stages for the required number of clock cycles. Since there is only three pipeline stages, no internal data forwarding is required, and no data dependencies can arise in the pipeline.

Figure 4-2 on page 24 shows an overview of the AVR32UC pipeline stages.

Figure 4-2. The AVR32UC Pipeline



4.3.2 AVR32A Microarchitecture Compliance

AVR32UC implements an AVR32A microarchitecture. The AVR32A microarchitecture is targeted at cost-sensitive, lower-end applications like smaller microcontrollers. This microarchitecture does not provide dedicated hardware registers for shadowing of register file registers in interrupt contexts. Additionally, it does not provide hardware registers for the return address registers and return status registers. Instead, all this information is stored on the system stack. This saves chip area at the expense of slower interrupt handling.

Upon interrupt initiation, registers R8-R12 are automatically pushed to the system stack. These registers are pushed regardless of the priority level of the pending interrupt. The return address and status register are also automatically pushed to stack. The interrupt handler can therefore use R8-R12 freely. Upon interrupt completion, the old R8-R12 registers and status register are restored, and execution continues at the return address stored popped from stack.

The stack is also used to store the status register and return address for exceptions and *scall*. Executing the *rete* or *rets* instruction at the completion of an exception or system call will pop this status register and continue execution at the popped return address.

4.3.3 Java Support

AVR32UC does not provide Java hardware acceleration.

4.3.4 Memory Protection

The MPU allows the user to check all memory accesses for privilege violations. If an access is attempted to an illegal memory address, the access is aborted and an exception is taken. The MPU in AVR32UC is specified in the AVR32UC Technical Reference manual.

4.3.5 Unaligned Reference Handling

AVR32UC does not support unaligned accesses, except for doubleword accesses. AVR32UC is able to perform word-aligned *st.d* and *ld.d*. Any other unaligned memory access will cause an 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 |

4.3.6 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

4.3.7 CPU and Architecture Revision

Three major revisions of the AVR32UC CPU currently exist.

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.

4.4 Programming Model

4.4.1 Register File Configuration

The AVR32UC register file is shown below.

Figure 4-3. The AVR32UC Register File

| Application | Supervisor | INT0 | INT1 | INT2 | INT3 | Exception | NMI | Secure | | | |
|-------------|------------|--------|--------|--------|--------|-----------|--------|--------|--------|--------|--------|
| Bit 31 | Bit 0 | Bit 31 | Bit 0 | Bit 31 | Bit 0 | Bit 31 | Bit 0 | Bit 31 | Bit 0 | Bit 31 | Bit 0 |
| PC | PC | PC | PC | PC | PC | PC | PC | PC | PC | PC | PC |
| LR | LR | LR | LR | LR | LR | LR | LR | LR | LR | LR | LR |
| SP_APP | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SYS | SP_SEC | SP_SEC |
| R12 | R12 | R12 | R12 | R12 | R12 | R12 | R12 | R12 | R12 | R12 | R12 |
| R11 | R11 | R11 | R11 | R11 | R11 | R11 | R11 | R11 | R11 | R11 | R11 |
| R10 | R10 | R10 | R10 | R10 | R10 | R10 | R10 | R10 | R10 | R10 | R10 |
| R9 | R9 | R9 | R9 | R9 | R9 | R9 | R9 | R9 | R9 | R9 | R9 |
| R8 | R8 | R8 | R8 | R8 | R8 | R8 | R8 | R8 | R8 | R8 | R8 |
| R7 | R7 | R7 | R7 | R7 | R7 | R7 | R7 | R7 | R7 | R7 | R7 |
| R6 | R6 | R6 | R6 | R6 | R6 | R6 | R6 | R6 | R6 | R6 | R6 |
| R5 | R5 | R5 | R5 | R5 | R5 | R5 | R5 | R5 | R5 | R5 | R5 |
| R4 | R4 | R4 | R4 | R4 | R4 | R4 | R4 | R4 | R4 | R4 | R4 |
| R3 | R3 | R3 | R3 | R3 | R3 | R3 | R3 | R3 | R3 | R3 | R3 |
| R2 | R2 | R2 | R2 | R2 | R2 | R2 | R2 | R2 | R2 | R2 | R2 |
| R1 | R1 | R1 | R1 | R1 | R1 | R1 | R1 | R1 | R1 | R1 | R1 |
| R0 | R0 | R0 | R0 | R0 | R0 | R0 | R0 | R0 | R0 | R0 | R0 |
| SR | SR | SR | SR | SR | SR | SR | SR | SR | SR | SR | SR |

| |
|-----------|
| SS_STATUS |
| SS_ADRF |
| SS_ADRR |
| SS_ADR0 |
| SS_ADR1 |
| SS_SP_SYS |
| SS_SP_APP |
| SS_RAR |
| SS_RSR |

4.4.2 Status Register Configuration

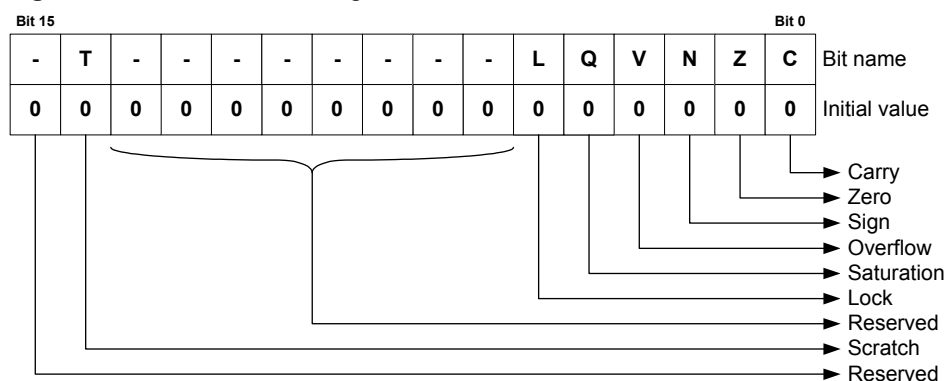
The Status Register (SR) is split into two halfwords, one upper and one lower, see [Figure 4-4 on page 26](#) and [Figure 4-5 on page 27](#). The lower word contains the C, Z, N, V, and Q condition code flags and the R, T, and L bits, while the upper halfword contains information about the mode and state the processor executes in. Refer to the *AVR32 Architecture Manual* for details.

Figure 4-4. The Status Register High Halfword

| Bit 31 | Bit 30 | Bit 29 | Bit 28 | DM | D | | M2 | M1 | M0 | EM | I3M | I2M | I1M | I0M | GM | Bit name |
|--------|--------|--------|--------|----|---|---|----|----|----|----|-----|-----|-----|-----|----|---------------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | Initial value |

- Global Interrupt Mask
- Interrupt Level 0 Mask
- Interrupt Level 1 Mask
- Interrupt Level 2 Mask
- Interrupt Level 3 Mask
- Exception Mask
- Mode Bit 0
- Mode Bit 1
- Mode Bit 2
- Reserved
- Debug State
- Debug State Mask
- Reserved

Figure 4-5. The Status Register Low Halfword



4.4.3 Processor States

4.4.3.1 Normal RISC State

The AVR32 processor supports several different execution contexts as shown in [Table 4-2 on page 27](#).

Table 4-2. Overview of Execution Modes, their Priorities and Privilege Levels.

| Priority | Mode | Security | Description |
|----------|------------------------|--------------|---|
| 1 | Non Maskable Interrupt | Privileged | Non Maskable high priority interrupt mode |
| 2 | Exception | Privileged | Execute exceptions |
| 3 | Interrupt 3 | Privileged | General purpose interrupt mode |
| 4 | Interrupt 2 | Privileged | General purpose interrupt mode |
| 5 | Interrupt 1 | Privileged | General purpose interrupt mode |
| 6 | Interrupt 0 | Privileged | General purpose interrupt mode |
| N/A | Supervisor | Privileged | Runs supervisor calls |
| N/A | Application | Unprivileged | Normal program execution mode |

Mode changes can be made under software control, or can be caused by external interrupts or exception processing. A mode can be interrupted by a higher priority mode, but never by one with lower priority. Nested exceptions can be supported with a minimal software overhead.

When running an operating system on the AVR32, user processes will typically execute in the application mode. The programs executed in this mode are restricted from executing certain instructions. Furthermore, most system registers together with the upper halfword of the status register cannot be accessed. Protected memory areas are also not available. All other operating modes are privileged and are collectively called System Modes. They have full access to all privileged and unprivileged resources. After a reset, the processor will be in supervisor mode.

4.4.3.2 Debug State

The AVR32 can be set in a debug state, which allows implementation of software monitor routines that can read out and alter system information for use during application development. This implies that all system and application registers, including the status registers and program counters, are accessible in debug state. The privileged instructions are also available.

All interrupt levels are by default disabled when debug state is entered, but they can individually be switched on by the monitor routine by clearing the respective mask bit in the status register.

Debug state can be entered as described in the *AVR32UC Technical Reference Manual*.

Debug state is exited by the *retd* instruction.

4.4.4 System Registers

The system registers are placed outside of the virtual memory space, and are only accessible using the privileged *mfsr* and *mtsr* instructions. The table below lists the system registers specified in the AVR32 architecture, some of which are unused in AVR32UC. The programmer is responsible for maintaining correct sequencing of any instructions following a *mtsr* instruction. For detail on the system registers, refer to the *AVR32UC Technical Reference Manual*.

Table 4-3. System Registers

| Reg # | Address | Name | Function |
|-------|---------|----------|--|
| 0 | 0 | SR | Status Register |
| 1 | 4 | EVBA | Exception Vector Base Address |
| 2 | 8 | ACBA | Application Call Base Address |
| 3 | 12 | CPUCR | CPU Control Register |
| 4 | 16 | ECR | Exception Cause Register |
| 5 | 20 | RSR_SUP | Unused in AVR32UC |
| 6 | 24 | RSR_INT0 | Unused in AVR32UC |
| 7 | 28 | RSR_INT1 | Unused in AVR32UC |
| 8 | 32 | RSR_INT2 | Unused in AVR32UC |
| 9 | 36 | RSR_INT3 | Unused in AVR32UC |
| 10 | 40 | RSR_EX | Unused in AVR32UC |
| 11 | 44 | RSR_NMI | Unused in AVR32UC |
| 12 | 48 | RSR_DBG | Return Status Register for Debug mode |
| 13 | 52 | RAR_SUP | Unused in AVR32UC |
| 14 | 56 | RAR_INT0 | Unused in AVR32UC |
| 15 | 60 | RAR_INT1 | Unused in AVR32UC |
| 16 | 64 | RAR_INT2 | Unused in AVR32UC |
| 17 | 68 | RAR_INT3 | Unused in AVR32UC |
| 18 | 72 | RAR_EX | Unused in AVR32UC |
| 19 | 76 | RAR_NMI | Unused in AVR32UC |
| 20 | 80 | RAR_DBG | Return Address Register for Debug mode |
| 21 | 84 | JECCR | Unused in AVR32UC |
| 22 | 88 | JOSP | Unused in AVR32UC |
| 23 | 92 | JAVA_LV0 | Unused in AVR32UC |
| 24 | 96 | JAVA_LV1 | Unused in AVR32UC |
| 25 | 100 | JAVA_LV2 | Unused in AVR32UC |

Table 4-3. System Registers (Continued)

| Reg # | Address | Name | Function |
|-------|---------|----------|--|
| 26 | 104 | JAVA_LV3 | Unused in AVR32UC |
| 27 | 108 | JAVA_LV4 | Unused in AVR32UC |
| 28 | 112 | JAVA_LV5 | Unused in AVR32UC |
| 29 | 116 | JAVA_LV6 | Unused in AVR32UC |
| 30 | 120 | JAVA_LV7 | Unused in AVR32UC |
| 31 | 124 | JTBA | Unused in AVR32UC |
| 32 | 128 | JBCR | Unused in AVR32UC |
| 33-63 | 132-252 | Reserved | Reserved for future use |
| 64 | 256 | CONFIG0 | Configuration register 0 |
| 65 | 260 | CONFIG1 | Configuration register 1 |
| 66 | 264 | COUNT | Cycle Counter register |
| 67 | 268 | COMPARE | Compare register |
| 68 | 272 | TLBEHI | Unused in AVR32UC |
| 69 | 276 | TLBELO | Unused in AVR32UC |
| 70 | 280 | PTBR | Unused in AVR32UC |
| 71 | 284 | TLBEAR | Unused in AVR32UC |
| 72 | 288 | MMUCR | Unused in AVR32UC |
| 73 | 292 | TLBARLO | Unused in AVR32UC |
| 74 | 296 | TLBARHI | Unused in AVR32UC |
| 75 | 300 | PCCNT | Unused in AVR32UC |
| 76 | 304 | PCNT0 | Unused in AVR32UC |
| 77 | 308 | PCNT1 | Unused in AVR32UC |
| 78 | 312 | PCCR | Unused in AVR32UC |
| 79 | 316 | BEAR | Bus Error Address Register |
| 80 | 320 | MPUAR0 | MPU Address Register region 0 |
| 81 | 324 | MPUAR1 | MPU Address Register region 1 |
| 82 | 328 | MPUAR2 | MPU Address Register region 2 |
| 83 | 332 | MPUAR3 | MPU Address Register region 3 |
| 84 | 336 | MPUAR4 | MPU Address Register region 4 |
| 85 | 340 | MPUAR5 | MPU Address Register region 5 |
| 86 | 344 | MPUAR6 | MPU Address Register region 6 |
| 87 | 348 | MPUAR7 | MPU Address Register region 7 |
| 88 | 352 | MPUPSR0 | MPU Privilege Select Register region 0 |
| 89 | 356 | MPUPSR1 | MPU Privilege Select Register region 1 |
| 90 | 360 | MPUPSR2 | MPU Privilege Select Register region 2 |
| 91 | 364 | MPUPSR3 | MPU Privilege Select Register region 3 |

Table 4-3. System Registers (Continued)

| Reg # | Address | Name | Function |
|---------|----------|----------|--|
| 92 | 368 | MPUPSR4 | MPU Privilege Select Register region 4 |
| 93 | 372 | MPUPSR5 | MPU Privilege Select Register region 5 |
| 94 | 376 | MPUPSR6 | MPU Privilege Select Register region 6 |
| 95 | 380 | MPUPSR7 | MPU Privilege Select Register region 7 |
| 96 | 384 | MPUCRA | Unused in this version of AVR32UC |
| 97 | 388 | MPUCRB | Unused in this version of AVR32UC |
| 98 | 392 | MPUBRA | Unused in this version of AVR32UC |
| 99 | 396 | MPUBRB | Unused in this version of AVR32UC |
| 100 | 400 | MPUAPRA | MPU Access Permission Register A |
| 101 | 404 | MPUAPRB | MPU Access Permission Register B |
| 102 | 408 | MPUCR | MPU Control Register |
| 103-191 | 448-764 | Reserved | Reserved for future use |
| 192-255 | 768-1020 | IMPL | IMPLEMENTATION DEFINED |

4.5 Exceptions and Interrupts

AVR32UC incorporates a powerful exception handling scheme. The different exception sources, like Illegal Op-code and external interrupt requests, have different priority levels, ensuring a well-defined behavior when multiple exceptions are received simultaneously. Additionally, pending exceptions of a higher priority class may preempt handling of ongoing exceptions of a lower priority class.

When an event occurs, the execution of the instruction stream is halted, and execution control is passed to an event handler at an address specified in [Table 4-4 on page 33](#). Most of the handlers are placed sequentially in the code space starting at the address specified by EVBA, with four bytes between each handler. This gives ample space for a jump instruction to be placed there, jumping to the event routine itself. A few critical handlers have larger spacing between them, allowing the entire event routine to be placed directly at the address specified by the EVBA-relative offset generated by hardware. All external interrupt sources have autovectorized interrupt service routine (ISR) addresses. This allows the interrupt controller to directly specify the ISR address as an address relative to EVBA. The autovector offset has 14 address bits, giving an offset of maximum 16384 bytes. The target address of the event handler is calculated as (EVBA | event_handler_offset), not (EVBA + event_handler_offset), so EVBA and exception code segments must be set up appropriately. The same mechanisms are used to service all different types of events, including external interrupt requests, yielding a uniform event handling scheme.

An interrupt controller does the priority handling of the external interrupts and provides the autovector offset to the CPU.

4.5.1 System Stack Issues

Event handling in AVR32UC uses the system stack pointed to by the system stack pointer, SP_SYS, for pushing and popping R8-R12, LR, status register, and return address. Since event code may be timing-critical, SP_SYS should point to memory addresses in the IIRAM section, since the timing of accesses to this memory section is both fast and deterministic.

The user must also make sure that the system stack is large enough so that any event is able to push the required registers to stack. If the system stack is full, and an event occurs, the system will enter an UNDEFINED state.

4.5.2 Exceptions and Interrupt Requests

When an event other than *scall* or debug request is received by the core, the following actions are performed atomically:

1. The pending event will not be accepted if it is masked. The I3M, I2M, I1M, I0M, EM, and GM bits in the Status Register are used to mask different events. Not all events can be masked. A few critical events (NMI, Unrecoverable Exception, TLB Multiple Hit, and Bus Error) can not be masked. When an event is accepted, hardware automatically sets the mask bits corresponding to all sources with equal or lower priority. This inhibits acceptance of other events of the same or lower priority, except for the critical events listed above. Software may choose to clear some or all of these bits after saving the necessary state if other priority schemes are desired. It is the event source's responsibility to ensure that their events are left pending until accepted by the CPU.
2. When a request is accepted, the Status Register and Program Counter of the current context is stored to the system stack. If the event is an INT0, INT1, INT2, or INT3, registers R8-R12 and LR are also automatically stored to stack. Storing the Status Register ensures that the core is returned to the previous execution mode when the current event handling is completed. When exceptions occur, both the EM and GM bits are set, and the application may manually enable nested exceptions if desired by clearing the appropriate bit. Each exception handler has a dedicated handler address, and this address uniquely identifies the exception source.
3. The Mode bits are set to reflect the priority of the accepted event, and the correct register file bank is selected. The address of the event handler, as shown in Table 4-4, is loaded into the Program Counter.

The execution of the event handler routine then continues from the effective address calculated.

The *rete* instruction signals the end of the event. When encountered, the Return Status Register and Return Address Register are popped from the system stack and restored to the Status Register and Program Counter. If the *rete* instruction returns from INT0, INT1, INT2, or INT3, registers R8-R12 and LR are also popped from the system stack. The restored Status Register contains information allowing the core to resume operation in the previous execution mode. This concludes the event handling.

4.5.3 Supervisor Calls

The AVR32 instruction set provides a supervisor mode call instruction. The *scall* instruction is designed so that privileged routines can be called from any context. This facilitates sharing of code between different execution modes. The *scall* mechanism is designed so that a minimal execution cycle overhead is experienced when performing supervisor routine calls from time-critical event handlers.

The *scall* instruction behaves differently depending on which mode it is called from. The behaviour is detailed in the instruction set reference. In order to allow the *scall* routine to return to the correct context, a return from supervisor call instruction, *rets*, is implemented. In the AVR32UC CPU, *scall* and *rets* uses the system stack to store the return address and the status register.

4.5.4 Debug Requests

The AVR32 architecture defines a dedicated Debug mode. When a debug request is received by the core, Debug mode is entered. Entry into Debug mode can be masked by the DM bit in the

status register. Upon entry into Debug mode, hardware sets the SR[D] bit and jumps to the Debug Exception handler. By default, Debug mode executes in the exception context, but with dedicated Return Address Register and Return Status Register. These dedicated registers remove the need for storing this data to the system stack, thereby improving debuggability. The mode bits in the status register can freely be manipulated in Debug mode, to observe registers in all contexts, while retaining full privileges.

Debug mode is exited by executing the *retd* instruction. This returns to the previous context.

4.5.5 Entry Points for Events

Several different event handler entry points exist. In AVR32UC, the reset address is 0x8000_0000. This places the reset address in the boot flash memory area.

TLB miss exceptions and *scall* have a dedicated space relative to EVBA where their event handler can be placed. This speeds up execution by removing the need for a jump instruction placed at the program address jumped to by the event hardware. All other exceptions have a dedicated event routine entry point located relative to EVBA. The handler routine address identifies the exception source directly.

AVR32UC uses the ITLB and DTLB protection exceptions to signal a MPU protection violation. ITLB and DTLB miss exceptions are used to signal that an access address did not map to any of the entries in the MPU. TLB multiple hit exception indicates that an access address did map to multiple TLB entries, signalling an error.

All external interrupt requests have entry points located at an offset relative to EVBA. This autovector offset is specified by an external Interrupt Controller. The programmer must make sure that none of the autovector offsets interfere with the placement of other code. The autovector offset has 14 address bits, giving an offset of maximum 16384 bytes.

Special considerations should be made when loading EVBA with a pointer. Due to security considerations, the event handlers should be located in non-writeable flash memory, or optionally in a privileged memory protection region if an MPU is present.

If several events occur on the same instruction, they are handled in a prioritized way. The priority ordering is presented in Table 4-4. If events occur on several instructions at different locations in the pipeline, the events on the oldest instruction are always handled before any events on any younger instruction, even if the younger instruction has events of higher priority than the oldest instruction. An instruction B is younger than an instruction A if it was sent down the pipeline later than A.

The addresses and priority of simultaneous events are shown in Table 4-4. Some of the exceptions are unused in AVR32UC since it has no MMU, coprocessor interface, or floating-point unit.

Table 4-4. Priority and Handler Addresses for Events

| Priority | Handler Address | Name | Event source | Stored Return Address |
|----------|------------------------|-----------------------------|----------------|---------------------------------|
| 1 | 0x8000_0000 | 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 | |
| 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 | |
| 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 | |
| 25 | EVBA+0x70 | DTLB Miss (Write) | MPU | |
| 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 | |

5. Memories

5.1 Embedded Memories

- Internal High-Speed Flash
 - 256KBytes (AT32UC3A3256/S)
 - 128Kbytes (AT32UC3A3128/S)
 - 64Kbytes (AT32UC3A364/S)
 - 0 wait state access at up to 36MHz in worst case conditions
 - 1 wait state access at up to 66MHz in worst case conditions
 - Pipelined Flash architecture, allowing burst reads from sequential Flash locations, hiding penalty of 1 wait state access
 - Pipelined Flash architecture typically reduces the cycle penalty of 1 wait state operation to only 15% compared to 0 wait state operation
 - 100 000 write cycles, 15-year data retention capability
 - Sector lock capabilities, Bootloader protection, Security Bit
 - 32 Fuses, Erased During Chip Erase
 - User page for data to be preserved during Chip Erase
- Internal High-Speed SRAM
 - 64KBytes, Single-cycle access at full speed on CPU Local Bus and accessible through the High Speed Bud (HSB) matrix
 - 2x32KBytes, accessible independently through the High Speed Bud (HSB) matrix

5.2 Physical Memory Map

The System Bus is implemented as a bus matrix. All system bus addresses are fixed, and they are never remapped in any way, not even in boot.

Note that AVR32 UC CPU uses unsegmented translation, as described in the AVR32UC Technical Architecture Manual.

The 32-bit physical address space is mapped as follows:

Table 5-1. AT32UC3A3A4 Physical Memory Map

| Device | Start Address | Size | Size | Size |
|-------------------------|---------------|--|--|--|
| | | AT32UC3A3256S AT32UC3A3256 AT32UC3A4256S AT32UC3A4256 | AT32UC3A3128S AT32UC3A3128 AT32UC3A4128S AT32UC3A4128 | AT32UC3A364S AT32UC3A364 AT32UC3A464S AT32UC3A464 |
| Embedded CPU SRAM | 0x00000000 | 64 KByte | 64 KByte | 64 KByte |
| Embedded Flash | 0x80000000 | 256 KByte | 128 KByte | 64 KByte |
| EBI SRAM CS0 | 0xC0000000 | 16 MByte | 16 MByte | 16 MByte |
| EBI SRAM CS2 | 0xC8000000 | 16 MByte | 16 MByte | 16 MByte |
| EBI SRAM CS3 | 0xCC000000 | 16 MByte | 16 MByte | 16 MByte |
| EBI SRAM CS4 | 0xD8000000 | 16 MByte | 16 MByte | 16 MByte |
| EBI SRAM CS5 | 0xDC000000 | 16 MByte | 16 MByte | 16 MByte |
| EBI SRAM CS1 /SDRAM CS0 | 0xD0000000 | 128 MByte | 128 MByte | 128 MByte |
| USB Data | 0xE0000000 | 64 KByte | 64 KByte | 64 KByte |

Table 5-1. AT32UC3A3A4 Physical Memory Map

| Device | Start Address | Size | Size | Size |
|-----------------|---------------|--|--|--|
| | | AT32UC3A3256S AT32UC3A3256 AT32UC3A4256S AT32UC3A4256 | AT32UC3A3128S AT32UC3A3128 AT32UC3A4128S AT32UC3A4128 | AT32UC3A364S AT32UC3A364 AT32UC3A464S AT32UC3A464 |
| HRAMC0 | 0xFF000000 | 32KByte | 32KByte | 32KByte |
| HRAMC1 | 0xFF008000 | 32KByte | 32KByte | 32KByte |
| HSB-PB Bridge A | 0xFFFF0000 | 64KByte | 64KByte | 64KByte |
| HSB-PB Bridge B | 0xFFFE0000 | 64KByte | 64KByte | 64KByte |

5.3 Peripheral Address Map

Table 5-2. Peripheral Address Mapping

| Address | | Peripheral Name |
|------------|---------|--|
| 0xFF100000 | DMACA | DMA Controller - DMACA |
| 0xFFFD0000 | AES | Advanced Encryption Standard - AES |
| 0xFFFE0000 | USB | USB 2.0 Device and Host Interface - USB |
| 0xFFFE1000 | HMATRIX | HSB Matrix - HMATRIX |
| 0xFFFE1400 | FLASHC | Flash Controller - FLASHC |
| 0xFFFE1C00 | SMC | Static Memory Controller - SMC |
| 0xFFFE2000 | SDRAMC | SDRAM Controller - SDRAMC |
| 0xFFFE2400 | ECCHRS | Error code corrector Hamming and Reed Solomon - ECCHRS |
| 0xFFFE2800 | BUSMON | Bus Monitor module - BUSMON |
| 0xFFFE4000 | MCI | Multimedia Card Interface - MCI |
| 0xFFFE8000 | MSI | Memory Stick Interface - MSI |
| 0xFFFF0000 | PDCA | Peripheral DMA Controller - PDCA |
| 0xFFFF0800 | INTC | Interrupt controller - INTC |

Table 5-2. Peripheral Address Mapping

| | | |
|------------|--------|--|
| 0xFFFF0C00 | PM | Power Manager - PM |
| 0xFFFF0D00 | RTC | Real Time Counter - RTC |
| 0xFFFF0D30 | WDT | Watchdog Timer - WDT |
| 0xFFFF0D80 | EIC | External Interrupt Controller - EIC |
| 0xFFFF1000 | GPIO | General Purpose Input/Output Controller - GPIO |
| 0xFFFF1400 | USART0 | Universal Synchronous/Asynchronous Receiver/Transmitter - USART0 |
| 0xFFFF1800 | USART1 | Universal Synchronous/Asynchronous Receiver/Transmitter - USART1 |
| 0xFFFF1C00 | USART2 | Universal Synchronous/Asynchronous Receiver/Transmitter - USART2 |
| 0xFFFF2000 | USART3 | Universal Synchronous/Asynchronous Receiver/Transmitter - USART3 |
| 0xFFFF2400 | SPI0 | Serial Peripheral Interface - SPI0 |
| 0xFFFF2800 | SPI1 | Serial Peripheral Interface - SPI1 |
| 0xFFFF2C00 | TWIM0 | Two-wire Master Interface - TWIM0 |
| 0xFFFF3000 | TWIM1 | Two-wire Master Interface - TWIM1 |
| 0xFFFF3400 | SSC | Synchronous Serial Controller - SSC |
| 0xFFFF3800 | TC0 | Timer/Counter - TC0 |
| 0xFFFF3C00 | ADC | Analog to Digital Converter - ADC |
| 0xFFFF4000 | ABDAC | Audio Bitstream DAC - ABDAC |
| 0xFFFF4400 | TC1 | Timer/Counter - TC1 |

Table 5-2. Peripheral Address Mapping

| | | |
|------------|-------|----------------------------------|
| 0xFFFF5000 | TWIS0 | Two-wire Slave Interface - TWIS0 |
| 0xFFFF5400 | TWIS1 | Two-wire Slave Interface - TWIS1 |

5.4 CPU Local Bus Mapping

Some of the registers in the GPIO module are mapped onto the CPU local bus, in addition to being mapped on the Peripheral Bus. These registers can therefore be reached both by accesses on the Peripheral Bus, and by accesses on the local bus.

Mapping these registers on the local bus allows cycle-deterministic toggling of GPIO pins since the CPU and GPIO are the only modules connected to this bus. Also, since the local bus runs at CPU speed, one write or read operation can be performed per clock cycle to the local bus-mapped GPIO registers.

The following GPIO registers are mapped on the local bus:

Table 5-3. Local Bus Mapped GPIO Registers

| Port | Register | Mode | Local Bus Address | Access |
|------|--------------------------------------|--------|-------------------|------------|
| 0 | Output Driver Enable Register (ODER) | WRITE | 0x40000040 | Write-only |
| | | SET | 0x40000044 | Write-only |
| | | CLEAR | 0x40000048 | Write-only |
| | | TOGGLE | 0x4000004C | Write-only |
| | Output Value Register (OVR) | WRITE | 0x40000050 | Write-only |
| | | SET | 0x40000054 | Write-only |
| | | CLEAR | 0x40000058 | Write-only |
| | | TOGGLE | 0x4000005C | Write-only |
| | Pin Value Register (PVR) | - | 0x40000060 | Read-only |
| 1 | Output Driver Enable Register (ODER) | WRITE | 0x40000140 | Write-only |
| | | SET | 0x40000144 | Write-only |
| | | CLEAR | 0x40000148 | Write-only |
| | | TOGGLE | 0x4000014C | Write-only |
| | Output Value Register (OVR) | WRITE | 0x40000150 | Write-only |
| | | SET | 0x40000154 | Write-only |
| | | CLEAR | 0x40000158 | Write-only |
| | | TOGGLE | 0x4000015C | Write-only |
| | Pin Value Register (PVR) | - | 0x40000160 | Read-only |

Table 5-3. Local Bus Mapped GPIO Registers

| Port | Register | Mode | Local Bus Address | Access |
|------|--------------------------------------|--------|-------------------|------------|
| 2 | Output Driver Enable Register (ODER) | WRITE | 0x40000240 | Write-only |
| | | SET | 0x40000244 | Write-only |
| | | CLEAR | 0x40000248 | Write-only |
| | | TOGGLE | 0x4000024C | Write-only |
| | Output Value Register (OVR) | WRITE | 0x40000250 | Write-only |
| | | SET | 0x40000254 | Write-only |
| | | CLEAR | 0x40000258 | Write-only |
| | | TOGGLE | 0x4000025C | Write-only |
| | Pin Value Register (PVR) | - | 0x40000260 | Read-only |
| 3 | Output Driver Enable Register (ODER) | WRITE | 0x40000340 | Write-only |
| | | SET | 0x40000344 | Write-only |
| | | CLEAR | 0x40000348 | Write-only |
| | | TOGGLE | 0x4000034C | Write-only |
| | Output Value Register (OVR) | WRITE | 0x40000350 | Write-only |
| | | SET | 0x40000354 | Write-only |
| | | CLEAR | 0x40000358 | Write-only |
| | | TOGGLE | 0x4000035C | Write-only |
| | Pin Value Register (PVR) | - | 0x40000360 | Read-only |

6. Boot Sequence

This chapter summarizes the boot sequence of the AT32UC3A3/A4. The behavior after power-up is controlled by the Power Manager. For specific details, refer to [Section 7. "Power Manager \(PM\)" on page 86](#).

6.1 Starting of Clocks

After power-up, the device will be held in a reset state by the Power-On Reset circuitry, until the power has stabilized throughout the device. Once the power has stabilized, the device will use the internal RC Oscillator as clock source.

On system start-up, the PLLs are disabled. All clocks to all modules are running. No clocks have a divided frequency, all parts of the system receives a clock with the same frequency as the internal RC Oscillator.

6.2 Fetching of Initial Instructions

After reset has been released, the AVR32 UC CPU starts fetching instructions from the reset address, which is 0x8000_0000. This address points to the first address in the internal Flash.

The internal Flash uses VDDIO voltage during read and write operations. It is recommended to use the BOD33 to monitor this voltage and make sure the VDDIO is above the minimum level (3.0V).

The code read from the internal Flash is free to configure the system to use for example the PLLs, to divide the frequency of the clock routed to some of the peripherals, and to gate the clocks to unused peripherals.

When powering up the device, there may be a delay before the voltage has stabilized, depending on the rise time of the supply used. The CPU can start executing code as soon as the supply is above the POR threshold, and before the supply is stable. Before switching to a high-speed clock source, the user should use the BOD to make sure the VDDCORE is above the minimum-level (1.62V).

7. Electrical Characteristics

7.1 Absolute Maximum Ratings*

| | |
|---|-----------------|
| Operating Temperature..... | -40°C to +85°C |
| Storage Temperature | -60°C to +150°C |
| Voltage on Input Pin with respect to Ground | -0.3V to 3.6V |
| Maximum Operating Voltage (VDDCORE) | 1.95V |
| Maximum Operating Voltage (VDDIO)..... | 3.6V |
| Total DC Output Current on all I/O Pin for TQFP144 package | 370 mA |
| for TFBGA144 package | 370 mA |

*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.

7.2 DC Characteristics

The following characteristics are applicable to the operating temperature range: $T_A = -40^{\circ}\text{C}$ to 85°C , unless otherwise specified and are certified for a junction temperature up to $T_J = 100^{\circ}\text{C}$.

Table 7-1. DC Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------------|--|---|----------------------------|------|----------------------------|------|
| V _{VDDIO} | DC Supply Peripheral I/Os | | 3.0 | | 3.6 | V |
| V _{VDDANA} | DC Analog Supply | | 3.0 | | 3.6 | V |
| V _{IL} | Input Low-level Voltage | All I/O pins except TWCK, TWD, RESET_N, TCK, TDI | -0.3 | | +0.8 | V |
| | | TWCK, TWD | V _{VDDIO} x0.7 | | V _{VDDIO} +0.5 | V |
| | | RESET_N, TCK, TDI | +0.8V | | | V |
| V _{IH} | Input High-level Voltage | All I/O pins except TWCK, TWD | 2.0 | | 3.6 | V |
| | | TWCK, TWD | | | | V |
| V _{OL} | Output Low-level Voltage | I _{OL} = -2mA for Pin drive x1 I _{OL} = -4mA for Pin drive x2 I _{OL} = -8mA for Pin drive x3 | | | 0.4 | V |
| V _{OH} | Output High-level Voltage | I _{OH} = 2mA for Pin drive x1 I _{OH} = 4mA for Pin drive x2 I _{OH} = 8mA for Pin drive x3 | V _{VDDIO} -0.4 | | | V |
| I _{LEAK} | Input Leakage Current | Pullup resistors disabled | | 0.05 | 1 | μA |
| C _{IN} | Input Capacitance | | | 7 | | pF |
| R _{PULLUP} | Pull-up Resistance | All I/O pins except RESET_N, TCK, TDI, TMS | 9 | 15 | 25 | KΩ |
| | | RESET_N, TCK, TDI, TMS | 5 | | 25 | KΩ |
| I _O | Output Current Pin drive 1x Pin drive 2x Pin drive 3x | | | | 2.0 4.0 8.0 | mA |
| I _{SC} | Static Current | On V _{VDDIN} = 3.3V, CPU in static mode | T _A = 25°C | | 30 | μA |
| | | | T _A = 85°C | | 175 | μA |

7.2.1 I/O Pin Output Level Typical Characteristics

Figure 7-1. I/O Pin drive x2 Output Low Level Voltage (VOL) vs. Source Current

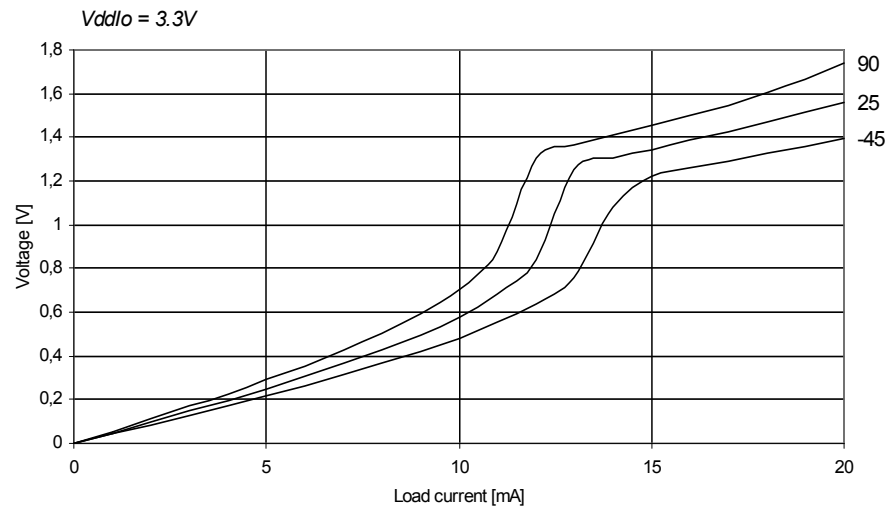
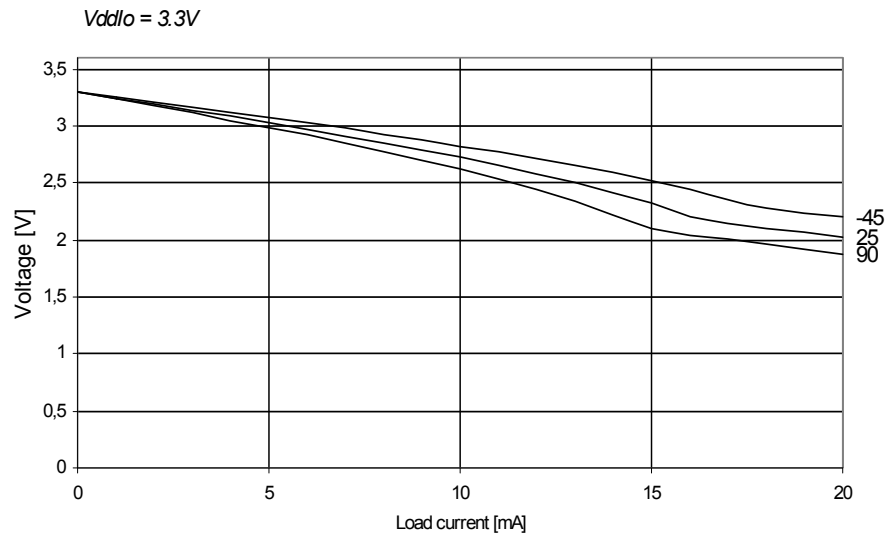


Figure 7-2. I/O Pin drive x2 Output High Level Voltage (VOH) vs. Source Current



7.3 I/O pin Characteristics

These parameters are given in the following conditions:

- $V_{DDCORE} = 1.8V$
- $V_{DDIO} = 3.3V$
- Ambient Temperature = 25°C

Table 7-2. Normal I/O Pin Characteristics

| Symbol | Parameter | Conditions | drive x2 | drive x2 | drive x3 | Unit |
|------------|------------------|------------|----------|----------|----------|------|
| f_{MAX} | Output frequency | 10pf | 40 | 66 | 100 | MHz |
| | | 30pf | 18.2 | 35.7 | 61.6 | MHz |
| | | 60pf | 7.5 | 18.5 | 36.3 | MHz |
| t_{RISE} | Rise time | 10pf | 2.7 | 1.4 | 0.9 | ns |
| | | 30pf | 6.9 | 3.5 | 1.9 | ns |
| | | 60pf | 13.4 | 6.7 | 3.5 | ns |
| t_{FALL} | Fall time | 10pf | 3.2 | 1.7 | 0.9 | ns |
| | | 30pf | 8.6 | 4.3 | 2.26 | ns |
| | | 60pf | 16.5 | 8.3 | 4.3 | ns |

7.4 Regulator characteristics

Table 7-3. Electrical Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------|---------------------------|--------------------|------|------|------|------|
| V_{VDDIN} | Supply voltage (input) | | 3.0 | 3.3 | 3.6 | V |
| $V_{VDDCORE}$ | Supply voltage (output) | | 1.75 | 1.85 | 1.95 | V |
| I_{OUT} | Maximum DC output current | $V_{VDDIN} = 3.3V$ | | | 100 | mA |

Table 7-4. Decoupling Requirements

| Symbol | Parameter | Conditions | Typ. | Technology | Unit |
|------------|------------------------------|------------|------|------------|---------|
| C_{IN1} | Input Regulator Capacitor 1 | | 1 | NPO | nF |
| C_{IN2} | Input Regulator Capacitor 2 | | 4.7 | X7R | μF |
| C_{OUT1} | Output Regulator Capacitor 1 | | 470 | NPO | pF |
| C_{OUT2} | Output Regulator Capacitor 2 | | 2.2 | X7R | μF |

7.5 Analog characteristics

7.5.1 ADC

Table 7-5. Electrical Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------------|---------------------|------------|------|------|------|------|
| V _{VDDANA} | Analog Power Supply | | 3.0 | | 3.6 | V |

Table 7-6. Decoupling Requirements

| Symbol | Parameter | Conditions | Typ. | Technology | Unit |
|---------------------|------------------------|------------|------|------------|------|
| C _{VDDANA} | Power Supply Capacitor | | 100 | NPO | nF |

7.5.2 BOD

Table 7-7. 1.8V BOD Level Values

| Symbol | Parameter Value | Conditions | Min. | Typ. | Max. | Unit |
|----------|-----------------|------------|------|------|------|------|
| BODLEVEL | 00 1111b | | | 1.79 | | V |
| | 01 0111b | | | 1.70 | | V |
| | 01 1111b | | | 1.61 | | V |
| | 10 0111b | | | 1.52 | | V |

Table 7-7 describes the values of the BODLEVEL field in the flash FGPFRR register.

Table 7-8. 3.3V BOD Level Values

| Symbol | Parameter Value | Conditions | Min. | Typ. | Max. | Unit |
|------------|-----------------|------------|------|------|------|------|
| BOD33LEVEL | Reset value | | | 2.71 | | V |
| | 1011 | | | 2.27 | | V |
| | 1010 | | | 2.37 | | V |
| | 1001 | | | 2.46 | | V |
| | 1000 | | | 2.56 | | V |
| | 0111 | | | 2.66 | | V |
| | 0110 | | | 2.76 | | V |
| | 0101 | | | 2.86 | | V |
| | 0100 | | | 2.96 | | V |
| | 0011 | | | 3.06 | | V |
| | 0010 | | | 3.15 | | V |
| | 0001 | | | 3.25 | | V |
| | 0000 | | | 3.35 | | V |

Table 7-8 describes the values of the BOD33.LEVEL field in the PM module

Table 7-9. BOD Timing

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------|--|-----------------------------------|------|------|------|------|
| T_{BOD} | Minimum time with VDDCORE < VBOD to detect power failure | Falling VDDCORE from 1.8V to 1.1V | | 300 | 800 | ns |

7.5.3 Reset Sequence

Table 7-10. Electrical Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------|--|--|------|------|------|---------|
| V_{DDRR} | VDDIN/VDDIO rise rate to ensure power-on-reset | | 0.8 | | | V/ms |
| V_{POR+} | Rising threshold voltage: voltage up to which device is kept under reset by POR on rising VDDIN | Rising VDDIN: $V_{RESTART} \rightarrow V_{POR+}$ | | 2.7 | | V |
| V_{POR-} | Falling threshold voltage: voltage when POR resets device on falling VDDIN | Falling VDDIN: 3.3V $\rightarrow V_{POR-}$ | | 2.7 | | V |
| $V_{RESTART}$ | On falling VDDIN, voltage must go down to this value before supply can rise again to ensure reset signal is released at V_{POR+} | Falling VDDIN: 3.3V $\rightarrow V_{RESTART}$ | | | 0.2 | V |
| T_{SSU1} | Time for Cold System Startup: Time for CPU to fetch its first instruction (RCosc not calibrated) | | 480 | | 960 | μ s |
| T_{SSU2} | Time for Hot System Startup: Time for CPU to fetch its first instruction (RCosc calibrated) | | | 420 | | μ s |

Figure 7-3. MCU Cold Start-Up

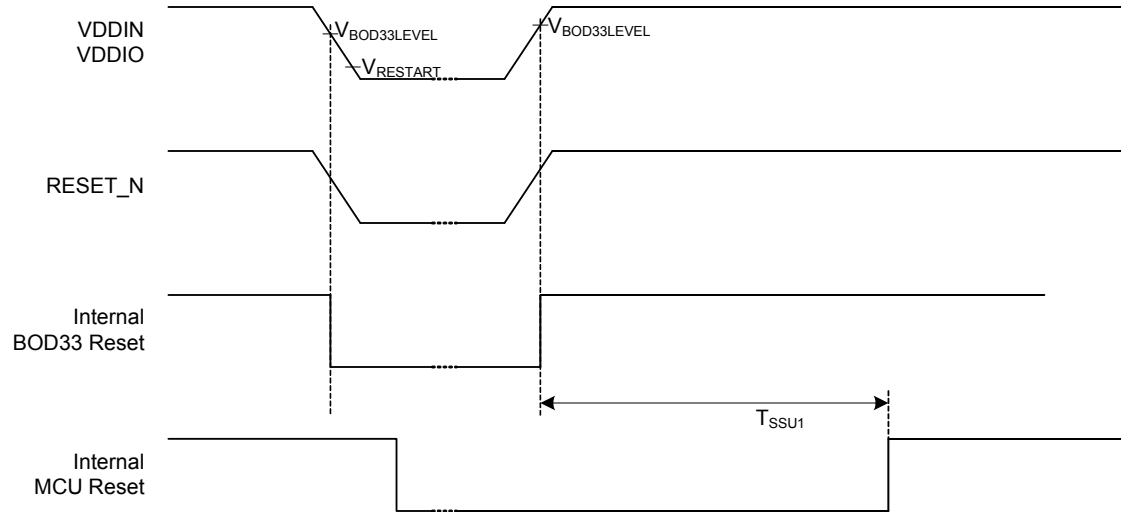


Figure 7-4. MCU Cold Start-Up RESET_N Externally Driven

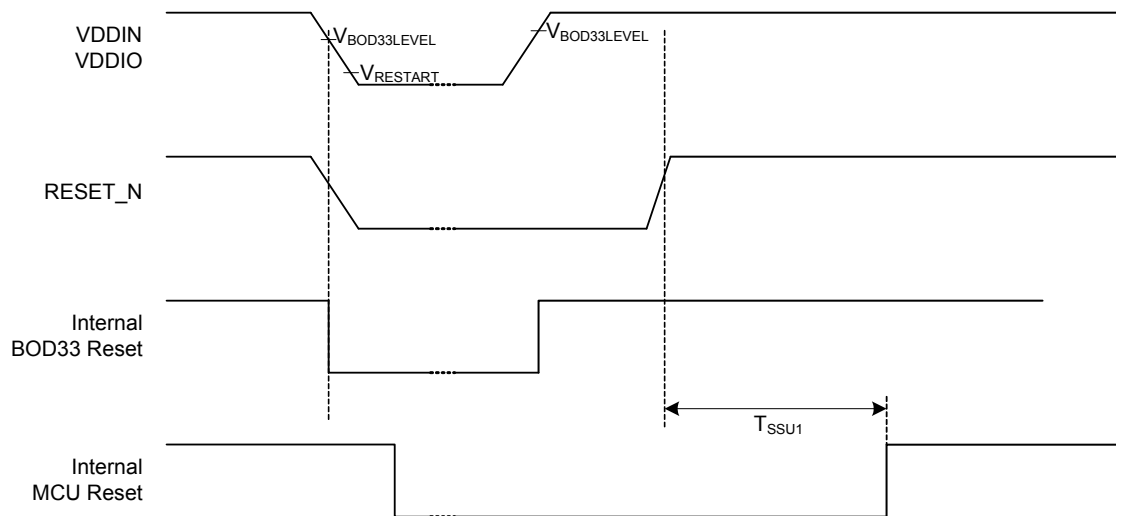
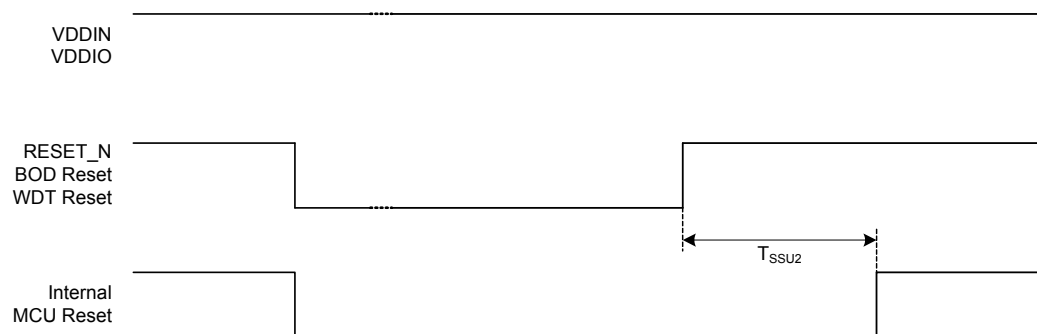


Figure 7-5. MCU Hot Start-Up



7.5.4 RESET_N Characteristics

Table 7-11. RESET_N Waveform Parameters

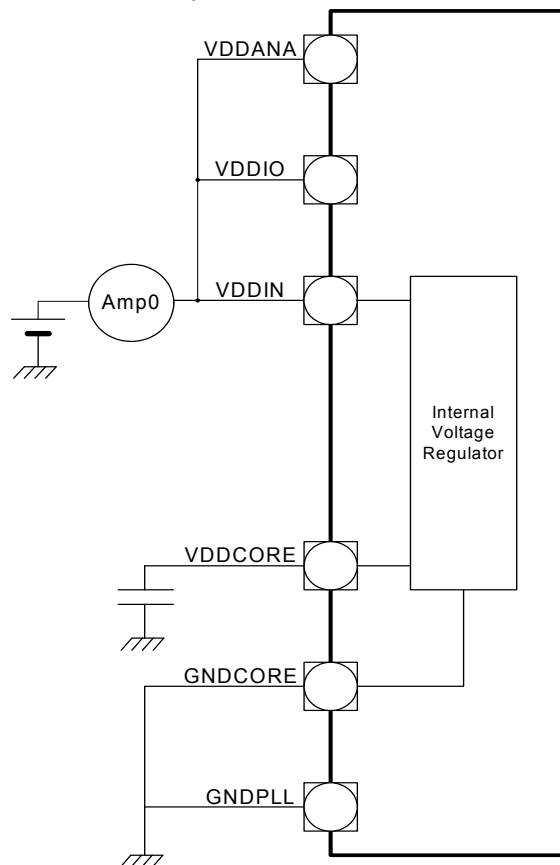
| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|--------------------|-----------------------------|------------|------|------|------|------|
| t_{RESET} | RESET_N minimum pulse width | | 10 | | | ns |

7.6 Power Consumption

The values in [Table 7-12](#) and [Table 7-13 on page 50](#) are measured values of power consumption with operating conditions as follows:

- $V_{DDIO} = 3.3V$
- $T_A = 25^{\circ}C$
- I/Os are configured in input, pull-up enabled.

Figure 7-6. Measurement Setup



These figures represent the power consumption measured on the power supplies

7.6.1 Power Consumption for Different Sleep Modes

Table 7-12. Power Consumption for Different Sleep Modes

| Mode | Conditions ⁽¹⁾ | | Typ. | Unit |
|----------|--|---------|--------------------------------------|--------|
| Active | <ul style="list-style-type: none"> - CPU running a recursive Fibonacci Algorithm from flash and clocked from PLL0 at f MHz. - Voltage regulator is on. - XIN0: external clock. Xin1 Stopped. XIN32 stopped. - All peripheral clocks activated with a division by 8. - GPIOs are inactive with internal pull-up, JTAG unconnected with external pullup and Input pins are connected to GND | | $0.626 \times f(\text{MHz}) + 2.257$ | mA/MHz |
| | Same conditions at 60 MHz | | 40 | mA |
| Idle | See Active mode conditions | | $0.349 \times f(\text{MHz}) + 0.968$ | mA/MHz |
| | Same conditions at 60 MHz | | 21.8 | mA |
| Frozen | See Active mode conditions | | $0.098 \times f(\text{MHz}) + 1.012$ | mA/MHz |
| | Same conditions at 60 MHz | | 6.6 | mA |
| Standby | See Active mode conditions | | $0.066 \times f(\text{MHz}) + 1.010$ | mA/MHz |
| | Same conditions at 60 MHz | | 4.6 | mA |
| Stop | <ul style="list-style-type: none"> - CPU running in sleep mode - XIN0, Xin1 and XIN32 are stopped. - All peripheral clocks are desactivated. - GPIOs are inactive with internal pull-up, JTAG unconnected with external pullup and Input pins are connected to GND. | | 96 | μA |
| Deepstop | See Stop mode conditions | | 54 | μA |
| Static | T _A = 25 °C CPU is in static mode GPIOs on internal pull-up All peripheral clocks de-activated DM and DP pins connected to ground XIN0, Xin1 and XIN32 are stopped | on Amp0 | 31 | μA |

Notes: 1. Core frequency is generated from XIN0 using the PLL.

Table 7-13. Typical Current Consumption by Peripheral

| Peripheral | Typ. | Unit |
|------------|------|--------------------------|
| ADC | 7 | $\mu\text{A}/\text{MHz}$ |
| AES | 80 | |
| ABDAC | 10 | |
| DMACA | 70 | |
| EBI | 23 | |
| EIC | 0.5 | |
| GPIO | 37 | |
| INTC | 3 | |
| MCI | 40 | |
| MSI | 10 | |
| PDCA | 20 | |
| SDRAM | 5 | |
| SMC | 9 | |
| SPI | 6 | |
| SSC | 10 | |
| RTC | 5 | |
| TC | 8 | |
| TWIM | 2 | |
| TWIS | 2 | |
| USART | 10 | |
| USBB | 90 | |
| WDT | 2 | |

7.7 System Clock Characteristics

These parameters are given in the following conditions:

- $V_{DDCORE} = 1.8V$
- Ambient Temperature = 25°C

7.7.1 CPU/HSB Clock Characteristics

Table 7-14. Core Clock Waveform Parameters

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------------|---------------------|------------|-------|------|------|------|
| $1/(t_{CPCPU})$ | CPU Clock Frequency | | | | 66 | MHz |
| t_{CPCPU} | CPU Clock Period | | 15,15 | | | ns |

7.7.2 PBA Clock Characteristics

Table 7-15. PBA Clock Waveform Parameters

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------------|---------------------|------------|-------|------|------|------|
| $1/(t_{CPPBA})$ | PBA Clock Frequency | | | | 66 | MHz |
| t_{CPPBA} | PBA Clock Period | | 15.15 | | | ns |

7.7.3 PBB Clock Characteristics

Table 7-16. PBB Clock Waveform Parameters

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------------|---------------------|------------|-------|------|------|------|
| $1/(t_{CPPBB})$ | PBB Clock Frequency | | | | 66 | MHz |
| t_{CPPBB} | PBB Clock Period | | 15.15 | | | ns |

7.8 Oscillator Characteristics

The following characteristics are applicable to the operating temperature range: $T_A = -40^{\circ}\text{C}$ to 85°C and worst case of power supply, unless otherwise specified.

7.8.1 Slow Clock RC Oscillator

Table 7-17. RC Oscillator Frequency

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|----------|-------------------------|---|------|-------|------|------|
| F_{RC} | RC Oscillator Frequency | Calibration point: $T_A = 85^{\circ}\text{C}$ | | 115.2 | 116 | KHz |
| | | $T_A = 25^{\circ}\text{C}$ | | 112 | | KHz |
| | | $T_A = -40^{\circ}\text{C}$ | 105 | 108 | | KHz |

7.8.2 32 KHz Oscillator

Table 7-18. 32 KHz Oscillator Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-------------------|--------------------------------------|---|--------------|--------|--------------|---------------|
| $1/(t_{CP32KHz})$ | Oscillator Frequency | External clock on XIN32 | | | 30 | MHz |
| | | Crystal | | 32 768 | | Hz |
| C_L | Equivalent Load Capacitance | | 6 | | 12.5 | pF |
| ESR | Crystal Equivalent Series Resistance | | | | 100 | K Ω |
| t_{ST} | Startup Time | $C_L = 6\text{pF}^{(1)}$ $C_L = 12.5\text{pF}^{(1)}$ | | | 600 1200 | ms |
| t_{CH} | XIN32 Clock High Half-period | | $0.4 t_{CP}$ | | $0.6 t_{CP}$ | |
| t_{CL} | XIN32 Clock Low Half-period | | $0.4 t_{CP}$ | | $0.6 t_{CP}$ | |
| C_{IN} | XIN32 Input Capacitance | | | | 5 | pF |
| I_{OSC} | Current Consumption | Active mode | | | 1.8 | μA |
| | | Standby mode | | | 0.1 | μA |

Note: 1. C_L is the equivalent load capacitance.

7.8.3 Main Oscillators

Table 7-19. Main Oscillators Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|------------------|---|--|--------------|-----------------------|--------------|----------|
| $1/(t_{CPMAIN})$ | Oscillator Frequency | External clock on XIN | | | 50 | MHz |
| | | Crystal | 0.4 | | 20 | MHz |
| C_{L1}, C_{L2} | Internal Load Capacitance ($C_{L1} = C_{L2}$) | | | 7 | | pF |
| ESR | Crystal Equivalent Series Resistance | | | | 75 | Ω |
| | Duty Cycle | | 40 | 50 | 60 | % |
| t_{ST} | Startup Time | f = 400 KHz f = 8 MHz f = 16 MHz f = 20 MHz | | 25 4 1.4 1 | | ms |
| t_{CH} | XIN Clock High Half-period | | 0.4 t_{CP} | | 0.6 t_{CP} | |
| t_{CL} | XIN Clock Low Half-period | | 0.4 t_{CP} | | 0.6 t_{CP} | |
| C_{IN} | XIN Input Capacitance | | | 7 | | pF |
| I_{OSC} | Current Consumption | Active mode at 400 KHz. Gain = G0 Active mode at 8 MHz. Gain = G1 Active mode at 16 MHz. Gain = G2 Active mode at 20 MHz. Gain = G3 | | 30 45 95 205 | | μA |

7.8.4 Phase Lock Loop

Table 7-20. PLL Characteristics

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------|---------------------------------------|----------------------------------|------|------|------|---------|
| F_{OUT} | VCO Output Frequency | | 80 | | 240 | MHz |
| F_{IN} | Input Frequency (after input divider) | | 4 | | 16 | MHz |
| I_{PLL} | Current Consumption | Active mode ($F_{out}=80$ MHz) | | 250 | | μA |
| | | Active mode ($F_{out}=240$ MHz) | | 600 | | μA |

7.9 ADC Characteristics

Table 7-21. Channel Conversion Time and ADC Clock

| Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------------------------|------------------------|------|------|--------------------|------|
| ADC Clock Frequency | 10-bit resolution mode | | | 5 | MHz |
| | 8-bit resolution mode | | | 8 | MHz |
| Startup Time | Return from Idle Mode | | | 20 | μs |
| Track and Hold Acquisition Time | | 600 | | | ns |
| Conversion Time | ADC Clock = 5 MHz | | | 2 | μs |
| | ADC Clock = 8 MHz | | | 1.25 | μs |
| Throughput Rate | ADC Clock = 5 MHz | | | 384 ⁽¹⁾ | kSPS |
| | ADC Clock = 8 MHz | | | 533 ⁽²⁾ | kSPS |

1. Corresponds to 13 clock cycles: 3 clock cycles for track and hold acquisition time and 10 clock cycles for conversion.

2. Corresponds to 15 clock cycles: 5 clock cycles for track and hold acquisition time and 10 clock cycles for conversion.

Table 7-22. ADC Power Consumption

| Parameter | Conditions | Min. | Typ. | Max. | Unit |
|--|--------------------------------------|------|------|------|------|
| Current Consumption on VDDANA ⁽¹⁾ | On 13 samples with ADC clock = 5 MHz | | | 1.25 | mA |

1. Including internal reference input current

Table 7-23. Analog Inputs

| Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-----------------------|------------|------|------|--------|------|
| Input Voltage Range | | 0 | | VDDANA | V |
| Input Leakage Current | | | | 1 | μA |
| Input Capacitance | | | 7 | | pF |
| Input Resistance | | | 350 | 850 | Ohm |

Table 7-24. Transfer Characteristics in 8-bit mode

| Parameter | Conditions | Min. | Typ. | Max. | Unit |
|----------------------------|-------------------|------|------|------|------|
| Resolution | | | 8 | | Bit |
| Absolute Accuracy | ADC Clock = 5 MHz | | | 0.8 | LSB |
| | ADC Clock = 8 MHz | | | 1.5 | LSB |
| Integral Non-linearity | ADC Clock = 5 MHz | | 0.35 | 0.5 | LSB |
| | ADC Clock = 8 MHz | | 0.5 | 1.0 | LSB |
| Differential Non-linearity | ADC Clock = 5 MHz | | 0.3 | 0.5 | LSB |
| | ADC Clock = 8 MHz | | 0.5 | 1.0 | LSB |
| Offset Error | ADC Clock = 5 MHz | -0.5 | | 0.5 | LSB |
| Gain Error | ADC Clock = 5 MHz | -0.5 | | 0.5 | LSB |

Table 7-25. Transfer Characteristics in 10-bit mode

| Parameter | Conditions | Min. | Typ. | Max. | Unit |
|----------------------------|---------------------|------|------|------|------|
| Resolution | | | 10 | | Bit |
| Absolute Accuracy | ADC Clock = 5 MHz | | | 3 | LSB |
| Integral Non-linearity | ADC Clock = 5 MHz | | 1.5 | 2 | LSB |
| Differential Non-linearity | ADC Clock = 5 MHz | | 1 | 2 | LSB |
| | ADC Clock = 2.5 MHz | | 0.6 | 1 | LSB |
| Offset Error | ADC Clock = 5 MHz | -2 | | 2 | LSB |
| Gain Error | ADC Clock = 5 MHz | -2 | | 2 | LSB |

7.10 USB Transceiver Characteristics

7.10.1 Electrical Characteristics

Table 7-26. Electrical Parameters

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-------------------|--|--|------|------|------|----------|
| R _{EXT} | Recommended External USB Series Resistor | In series with each USB pin with $\pm 5\%$ | | 39 | | Ω |
| R _{BIAS} | VBIAS External Resistor ⁽¹⁾ | $\pm 1\%$ | | 6810 | | Ω |
| C _{BIAS} | VBIAS External Capacitor | | | 10 | | pF |

1. The USB on-chip buffers comply with the Universal Serial Bus (USB) v2.0 standard. All AC parameters related to these buffers can be found within the USB 2.0 electrical specifications.

7.10.2 Static Power Consumption

Table 7-27. Static Power Consumption

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|----------------------|---|---|------|------|------|---------------|
| I _{BIAS} | Bias current consumption on VBG | | | | 1 | μA |
| I _{VDDUTMI} | HS Transceiver and I/O current consumption | | | | 8 | μA |
| | FS/HS Transceiver and I/O current consumption | If cable is connected, add 200 μA (typical) due to Pull-up/Pull-down current consumption | | | 3 | μA |

7.10.3 Dynamic Power Consumption

Table 7-28. Dynamic Power Consumption

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|-------------------|---------------------------------|------------|------|------|------|------|
| I _{BIAS} | Bias current consumption on VBG | | | 0.7 | 0.8 | mA |

Table 7-28. Dynamic Power Consumption

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------|---------------------------------------|---|------|------|------|------|
| $I_{VDDUTMI}$ | HS Transceiver current consumption | HS transmission | | 47 | 60 | mA |
| | HS Transceiver current consumption | HS reception | | 18 | 27 | mA |
| | FS/HS Transceiver current consumption | FS transmission 0m cable ⁽¹⁾ | | 4 | 6 | mA |
| | FS/HS Transceiver current consumption | FS transmission 5m cable | | 26 | 30 | mA |
| | FS/HS Transceiver current consumption | FS reception | | 3 | 4.5 | mA |

1. Including 1 mA due to Pull-up/Pull-down current consumption.

34.5.5 USB High Speed Design Guidelines

In order to facilitate hardware design, Atmel provides an application note on www.atmel.com.

7.11 EBI Timings

7.11.1 SMC Signals

These timings are given for worst case process, T = 85°C, VDDIO = 3V and 40 pF load capacitance.

Table 7-29. SMC Clock Signal

| Symbol | Parameter | Max. ⁽¹⁾ | Unit |
|-------------------------|--------------------------------|-------------------------|------|
| 1/(t _{CPSMC}) | SMC Controller Clock Frequency | 1/(t _{cpCPU}) | MHz |

Note: 1. The maximum frequency of the SMC interface is the same as the max frequency for the HSB.

Table 7-30. SMC Read Signals with Hold Settings

| Symbol | Parameter | Min. | Unit |
|---------------------------------------|--|---|------|
| NRD Controlled (READ_MODE = 1) | | | |
| SMC ₁ | Data Setup before NRD High | 12 | ns |
| SMC ₂ | Data Hold after NRD High | 0 | ns |
| SMC ₃ | NRD High to NBS0/A0 Change ⁽¹⁾ | nrd hold length * t _{CPSMC} - 1.3 | ns |
| SMC ₄ | NRD High to NBS1 Change ⁽¹⁾ | nrd hold length * t _{CPSMC} - 1.3 | ns |
| SMC ₅ | NRD High to NBS2/A1 Change ⁽¹⁾ | nrd hold length * t _{CPSMC} - 1.3 | ns |
| SMC ₇ | NRD High to A2 - A23 Change ⁽¹⁾ | nrd hold length * t _{CPSMC} - 1.3 | ns |
| SMC ₈ | NRD High to NCS Inactive ⁽¹⁾ | (nrd hold length - ncs rd hold length) * t _{CPSMC} - 2.3 | ns |
| SMC ₉ | NRD Pulse Width | nrd pulse length * t _{CPSMC} - 1.4 | ns |
| NRD Controlled (READ_MODE = 0) | | | |
| SMC ₁₀ | Data Setup before NCS High | 11.5 | ns |
| SMC ₁₁ | Data Hold after NCS High | 0 | ns |
| SMC ₁₂ | NCS High to NBS0/A0 Change ⁽¹⁾ | ncs rd hold length * t _{CPSMC} - 2.3 | ns |
| SMC ₁₃ | NCS High to NBS0/A0 Change ⁽¹⁾ | ncs rd hold length * t _{CPSMC} - 2.3 | ns |
| SMC ₁₄ | NCS High to NBS2/A1 Change ⁽¹⁾ | ncs rd hold length * t _{CPSMC} - 2.3 | ns |
| SMC ₁₆ | NCS High to A2 - A23 Change ⁽¹⁾ | ncs rd hold length * t _{CPSMC} - 4 | ns |
| SMC ₁₇ | NCS High to NRD Inactive ⁽¹⁾ | ncs rd hold length - nrd hold length) * t _{CPSMC} - 1.3 | ns |
| SMC ₁₈ | NCS Pulse Width | ncs rd pulse length * t _{CPSMC} - 3.6 | ns |

Note: 1. hold length = total cycle duration - setup duration - pulse duration. "hold length" is for "ncs rd hold length" or "nrd hold length".

Table 7-31. SMC Read Signals with no Hold Settings

| Symbol | Parameter | Min. | Unit |
|---------------------------------------|----------------------------|------|------|
| NRD Controlled (READ_MODE = 1) | | | |
| SMC ₁₉ | Data Setup before NRD High | 13.7 | ns |
| SMC ₂₀ | Data Hold after NRD High | 1 | ns |
| NRD Controlled (READ_MODE = 0) | | | |
| SMC ₂₁ | Data Setup before NCS High | 13.3 | ns |
| SMC ₂₂ | Data Hold after NCS High | 0 | ns |

Table 7-32. SMC Write Signals with Hold Settings

| Symbol | Parameter | Min. | Unit |
|---------------------------------------|--|--|------|
| NRD Controlled (READ_MODE = 1) | | | |
| SMC ₂₃ | Data Out Valid before NWE High | $(nwe \text{ pulse length} - 1) * t_{CPSMC} - 0.9$ | ns |
| SMC ₂₄ | Data Out Valid after NWE High ⁽¹⁾ | $nwe \text{ hold length} * t_{CPSMC} - 6$ | ns |
| SMC ₂₅ | NWE High to NBS0/A0 Change ⁽¹⁾ | $nwe \text{ hold length} * t_{CPSMC} - 1.9$ | ns |
| SMC ₂₆ | NWE High to NBS1 Change ⁽¹⁾ | $nwe \text{ hold length} * t_{CPSMC} - 1.9$ | ns |
| SMC ₂₉ | NWE High to A1 Change ⁽¹⁾ | $nwe \text{ hold length} * t_{CPSMC} - 1.9$ | ns |
| SMC ₃₁ | NWE High to A2 - A23 Change ⁽¹⁾ | $nwe \text{ hold length} * t_{CPSMC} - 1.7$ | ns |
| SMC ₃₂ | NWE High to NCS Inactive ⁽¹⁾ | $(nwe \text{ hold length} - ncs \text{ wr hold length}) * t_{CPSMC} - 2.9$ | ns |
| SMC ₃₃ | NWE Pulse Width | $nwe \text{ pulse length} * t_{CPSMC} - 0.9$ | ns |
| NRD Controlled (READ_MODE = 0) | | | |
| SMC ₃₄ | Data Out Valid before NCS High | $(ncs \text{ wr pulse length} - 1) * t_{CPSMC} - 4.6$ | ns |
| SMC ₃₅ | Data Out Valid after NCS High ⁽¹⁾ | $ncs \text{ wr hold length} * t_{CPSMC} - 5.8$ | ns |
| SMC ₃₆ | NCS High to NWE Inactive ⁽¹⁾ | $(ncs \text{ wr hold length} - nwe \text{ hold length}) * t_{CPSMC} - 0.6$ | ns |

Note: 1. hold length = total cycle duration - setup duration - pulse duration. "hold length" is for "ncs wr hold length" or "nwe hold length"

Table 7-33. SMC Write Signals with No Hold Settings (NWE Controlled only)

| Symbol | Parameter | Min. | Unit |
|-------------------|------------------------------|------|------|
| SMC ₃₇ | NWE Rising to A2-A25 Valid | 5.4 | ns |
| SMC ₃₈ | NWE Rising to NBS0/A0 Valid | 5 | ns |
| SMC ₃₉ | NWE Rising to NBS1 Change | 5 | ns |
| SMC ₄₀ | NWE Rising to A1/NBS2 Change | 5 | ns |
| SMC ₄₁ | NWE Rising to NBS3 Change | 5 | ns |
| SMC ₄₂ | NWE Rising to NCS Rising | 5.1 | ns |

Table 7-33. SMC Write Signals with No Hold Settings (NWE Controlled only)

| Symbol | Parameter | Min. | Unit |
|-------------------|----------------------------------|--|------|
| SMC ₄₃ | Data Out Valid before NWE Rising | $(nwe \text{ pulse length} - 1) * t_{CPSMC} - 1.2$ | ns |
| SMC ₄₄ | Data Out Valid after NWE Rising | 5 | ns |
| SMC ₄₅ | NWE Pulse Width | $nwe \text{ pulse length} * t_{CPSMC} - 0.9$ | ns |

Figure 7-7. SMC Signals for NCS Controlled Accesses.

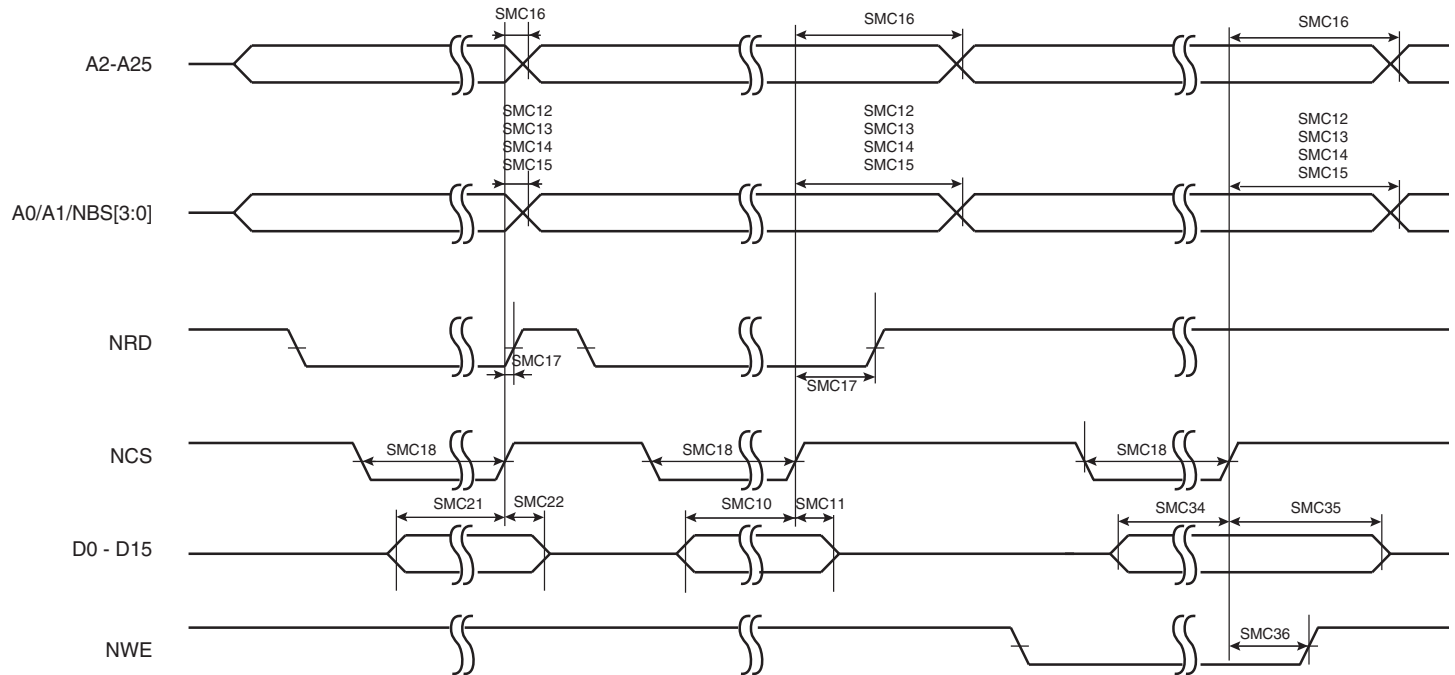
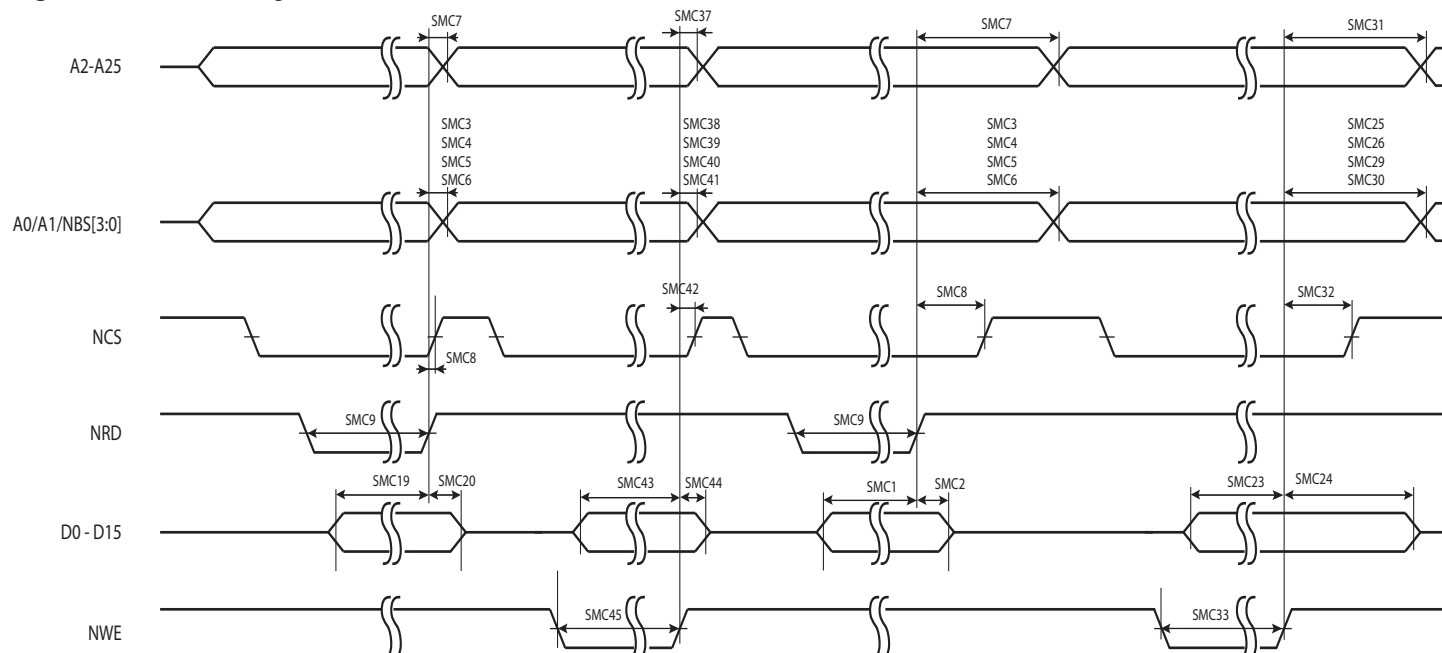


Figure 7-8. SMC Signals for NRD and NRW Controlled Accesses.



7.11.2 SDRAM Signals

These timings are given for 10 pF load on SDCK and 40 pF on other signals.

Table 7-34. SDRAM Clock Signal.

| Symbol | Parameter | Conditions | Min. | Max. ⁽¹⁾ | Unit |
|--------------------------|----------------------------------|------------|------|-----------------------|------|
| 1/(t _{CPSDCK}) | SDRAM Controller Clock Frequency | | | 1/(t _{cpu}) | MHz |

Note: 1. The maximum frequency of the SDRAMC interface is the same as the max frequency for the HSB.

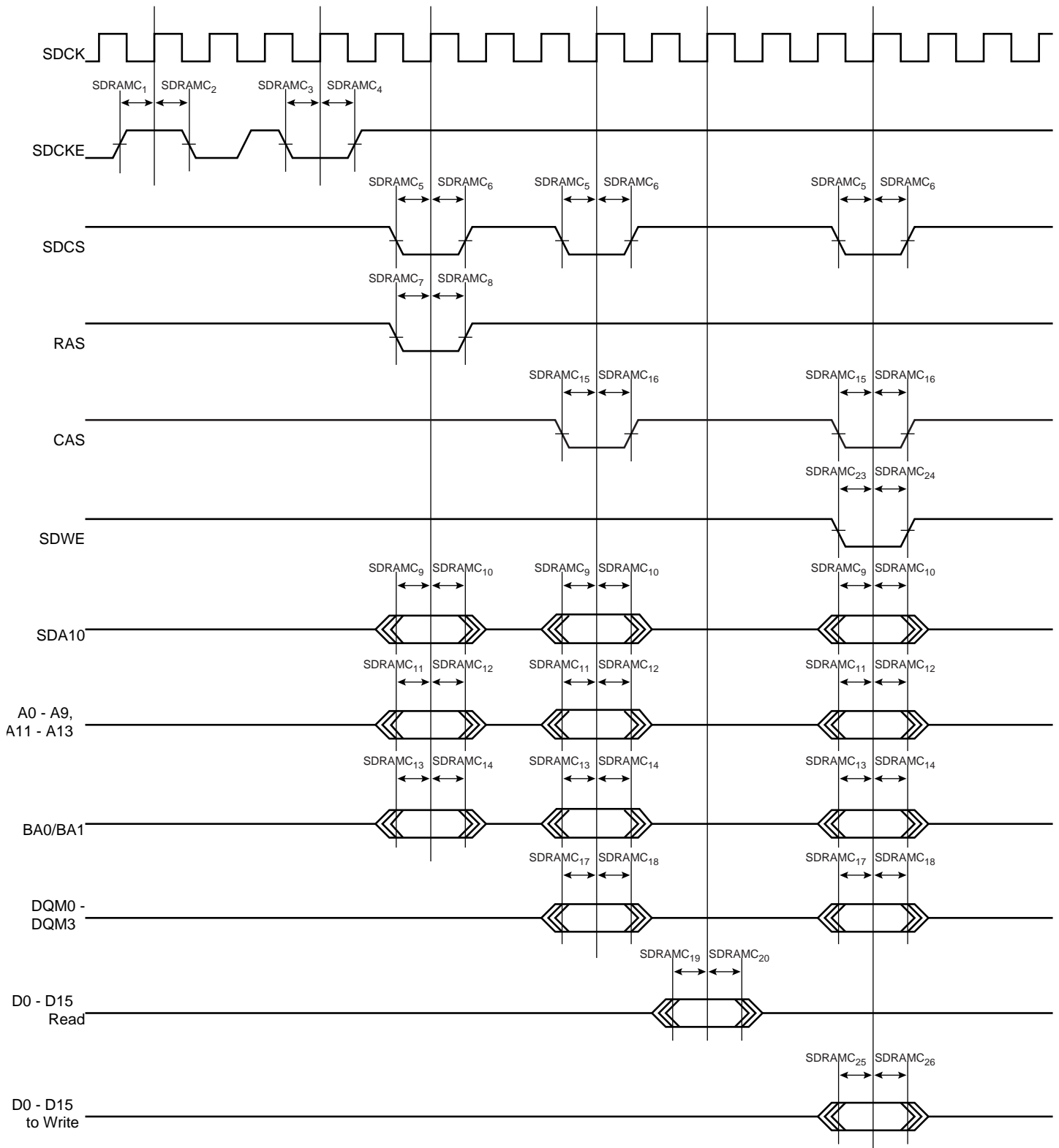
Table 7-35. SDRAM Clock Signal

| Symbol | Parameter | Conditions | Min. | Max. | Unit |
|----------------------|--|------------|------|------|------|
| SDRAMC ₁ | SDCKE High before SDCK Rising Edge | | 7.4 | | ns |
| SDRAMC ₂ | SDCKE Low after SDCK Rising Edge | | 3.2 | | ns |
| SDRAMC ₃ | SDCKE Low before SDCK Rising Edge | | 7 | | ns |
| SDRAMC ₄ | SDCKE High after SDCK Rising Edge | | 2.9 | | ns |
| SDRAMC ₅ | SDCS Low before SDCK Rising Edge | | 7.5 | | ns |
| SDRAMC ₆ | SDCS High after SDCK Rising Edge | | 1.6 | | ns |
| SDRAMC ₇ | RAS Low before SDCK Rising Edge | | 7.2 | | ns |
| SDRAMC ₈ | RAS High after SDCK Rising Edge | | 2.3 | | ns |
| SDRAMC ₉ | SDA10 Change before SDCK Rising Edge | | 7.6 | | ns |
| SDRAMC ₁₀ | SDA10 Change after SDCK Rising Edge | | 1.9 | | ns |
| SDRAMC ₁₁ | Address Change before SDCK Rising Edge | | 6.2 | | ns |
| SDRAMC ₁₂ | Address Change after SDCK Rising Edge | | 2.2 | | ns |

Table 7-35. SDRAM Clock Signal

| Symbol | Parameter | Conditions | Min. | Max. | Unit |
|----------------------|--|------------|------|------|------|
| SDRAMC ₁₃ | Bank Change before SDCK Rising Edge | | 6.3 | | ns |
| SDRAMC ₁₄ | Bank Change after SDCK Rising Edge | | 2.4 | | ns |
| SDRAMC ₁₅ | CAS Low before SDCK Rising Edge | | 7.4 | | ns |
| SDRAMC ₁₆ | CAS High after SDCK Rising Edge | | 1.9 | | ns |
| SDRAMC ₁₇ | DQM Change before SDCK Rising Edge | | 6.4 | | ns |
| SDRAMC ₁₈ | DQM Change after SDCK Rising Edge | | 2.2 | | ns |
| SDRAMC ₁₉ | D0-D15 in Setup before SDCK Rising Edge | | 9 | | ns |
| SDRAMC ₂₀ | D0-D15 in Hold after SDCK Rising Edge | | 0 | | ns |
| SDRAMC ₂₃ | SDWE Low before SDCK Rising Edge | | 7.6 | | ns |
| SDRAMC ₂₄ | SDWE High after SDCK Rising Edge | | 1.8 | | ns |
| SDRAMC ₂₅ | D0-D15 Out Valid before SDCK Rising Edge | | 7.1 | | ns |
| SDRAMC ₂₆ | D0-D15 Out Valid after SDCK Rising Edge | | 1.5 | | ns |

Figure 7-9. SDRAMC Signals relative to SDCK.



7.12 JTAG Characteristics

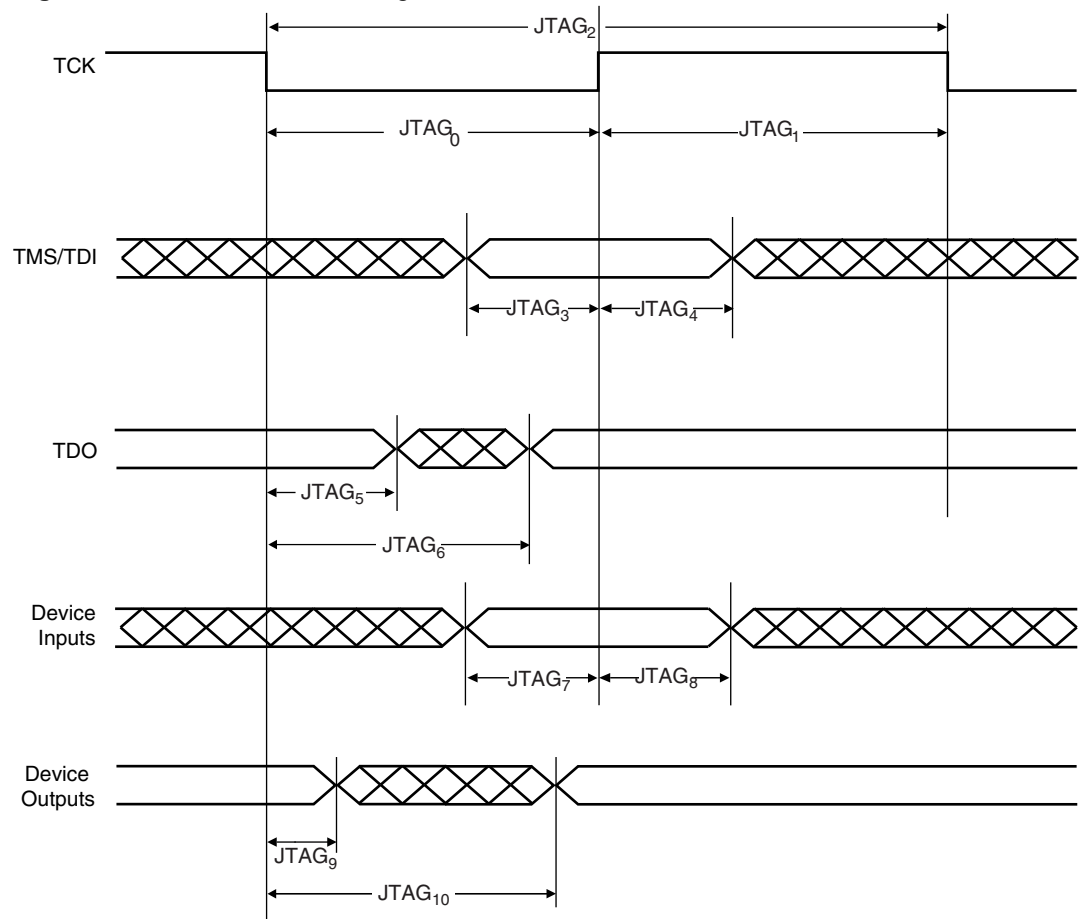
7.12.1 JTAG Interface Signals

Table 7-36. JTAG Interface Timing Specification

| Symbol | Parameter | Conditions ⁽¹⁾ | Min. | Max. | Unit |
|--------------------|--------------------------------|---------------------------|------|------|------|
| JTAG ₀ | TCK Low Half-period | | 6 | | ns |
| JTAG ₁ | TCK High Half-period | | 3 | | ns |
| JTAG ₂ | TCK Period | | 9 | | ns |
| JTAG ₃ | TDI, TMS Setup before TCK High | | 1 | | ns |
| JTAG ₄ | TDI, TMS Hold after TCK High | | 0 | | ns |
| JTAG ₅ | TDO Hold Time | | 4 | | ns |
| JTAG ₆ | TCK Low to TDO Valid | | | 6 | ns |
| JTAG ₇ | Device Inputs Setup Time | | | | ns |
| JTAG ₈ | Device Inputs Hold Time | | | | ns |
| JTAG ₉ | Device Outputs Hold Time | | | | ns |
| JTAG ₁₀ | TCK to Device Outputs Valid | | | | ns |

1. V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40pF

Figure 7-10. JTAG Interface Signals



7.13 SPI Characteristics

Figure 7-11. SPI Master mode with (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

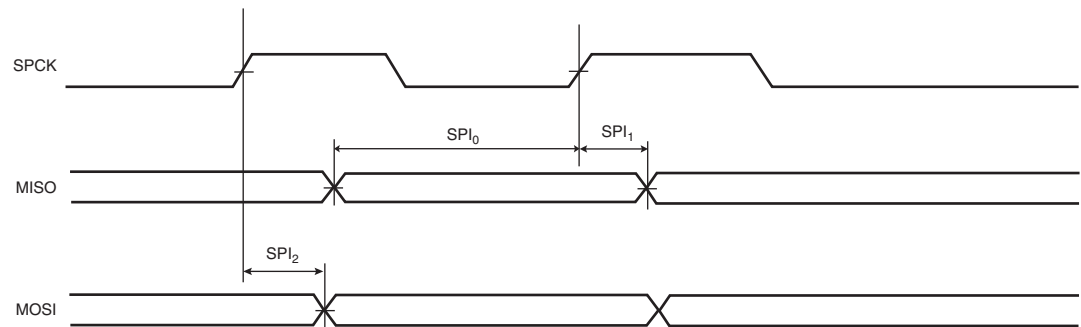


Figure 7-12. SPI Master mode with (CPOL= 0 and NCPHA= 1) or (CPOL= 1 and NCPHA= 0)

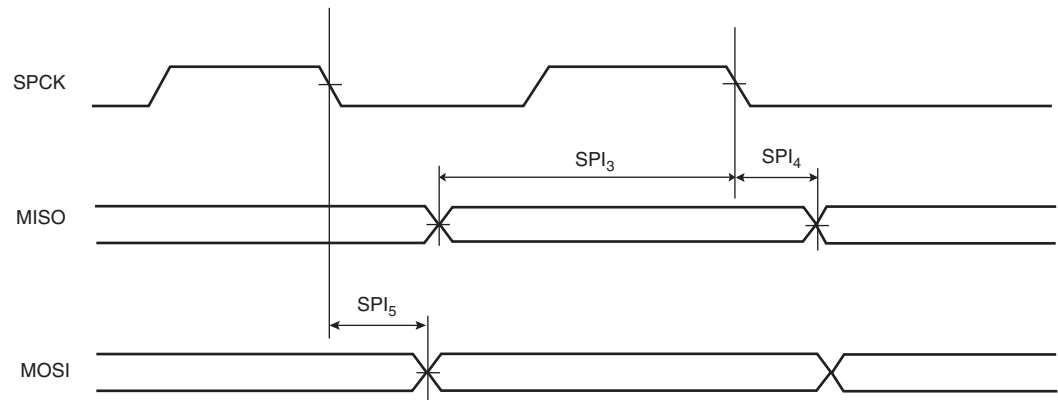


Figure 7-13. SPI Slave mode with (CPOL= 0 and NCPHA= 1) or (CPOL= 1 and NCPHA= 0)

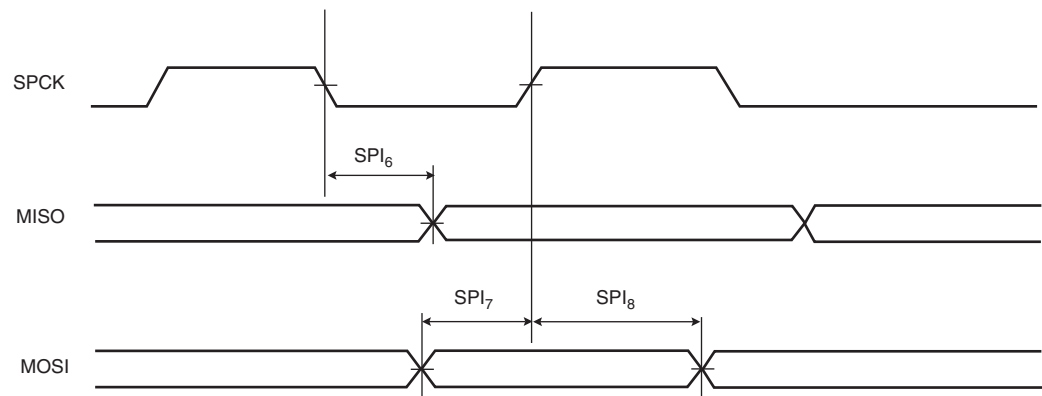


Figure 7-14. SPI Slave mode with (CPOL= NCPHA= 0) or (CPOL= NCPHA= 1)

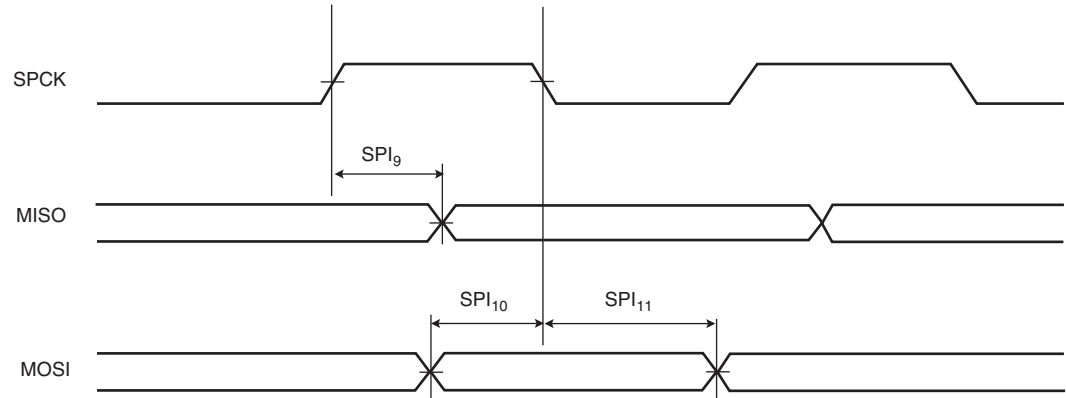


Table 7-37. SPI Timings

| Symbol | Parameter | Conditions ⁽¹⁾ | Min. | Max. | Unit |
|-------------------|---|---------------------------|-------------------------------------|------|------|
| SPI ₀ | MISO Setup time before SPCK rises (master) | 3.3V domain | 22 + $(t_{CPMCK})/2$ ⁽²⁾ | | ns |
| SPI ₁ | MISO Hold time after SPCK rises (master) | 3.3V domain | 0 | | ns |
| SPI ₂ | SPCK rising to MOSI Delay (master) | 3.3V domain | | 7 | ns |
| SPI ₃ | MISO Setup time before SPCK <u>falls</u> (master) | 3.3V domain | 22 + $(t_{CPMCK})/2$ ⁽³⁾ | | ns |
| SPI ₄ | MISO Hold time after SPCK falls (master) | 3.3V domain | 0 | | ns |
| SPI ₅ | SPCK falling to MOSI Delay (master) | 3.3V domain | | 7 | ns |
| SPI ₆ | SPCK falling to MISO Delay (slave) | 3.3V domain | | 26.5 | ns |
| SPI ₇ | MOSI Setup time before SPCK rises (slave) | 3.3V domain | 0 | | ns |
| SPI ₈ | MOSI Hold time after SPCK rises (slave) | 3.3V domain | 1.5 | | ns |
| SPI ₉ | SPCK rising to MISO Delay (slave) | 3.3V domain | | 27 | ns |
| SPI ₁₀ | MOSI Setup time before SPCK falls (slave) | 3.3V domain | 0 | | ns |
| SPI ₁₁ | MOSI Hold time after SPCK falls (slave) | 3.3V domain | 1 | | ns |

1. 3.3V domain: V_{VDDIO} from 3.0V to 3.6V, maximum external capacitor = 40 pF

2. t_{CPMCK} : Master Clock period in ns.

3. t_{CPMCK} : Master Clock period in ns.

7.14 MCI

The High Speed MultiMedia Card Interface (MCI) supports the MultiMedia Card (MMC) Specification V4.2, the SD Memory Card Specification V2.0, the SDIO V1.1 specification and CE-ATA V1.1.

7.15 Flash Memory Characteristics

The following table gives the device maximum operating frequency depending on the field FWS of the Flash FSR register. This field defines the number of wait states required to access the Flash Memory. Flash operating frequency equals the CPU/HSB frequency.

Table 7-38. Flash Operating Frequency

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|------------------|---------------------------|------------|------|------|------|------|
| F _{FOP} | Flash Operating Frequency | FWS = 0 | | | 36 | MHz |
| | | FWS = 1 | | | 66 | MHz |

Table 7-39. Parts Programming Time

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|------------------|-----------------------|------------|------|------|------|------|
| T _{FPP} | Page Programming Time | | | 5 | | ms |
| T _{FFP} | Fuse Programming Time | | | 0.5 | | ms |
| T _{FCE} | Chip erase Time | | | 8 | | ms |

Table 7-40. Flash Parameters

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---------------------|-----------------------------------|------------|------|------|------|-------|
| N _{FARRAY} | Flash Array Write/Erase cycle | | | | 100K | cycle |
| N _{FFUSE} | General Purpose Fuses write cycle | | | | 1000 | cycle |
| T _{FDR} | Flash Data Retention Time | | | 15 | | year |

8. Mechanical Characteristics

8.1 Thermal Considerations

8.1.1 Thermal Data

Table 8-1 summarizes the thermal resistance data depending on the package.

Table 8-1. Thermal Resistance Data

| Symbol | Parameter | Condition | Package | Typ | Unit |
|---------------|--|-----------|----------|------|------|
| θ_{JA} | Junction-to-ambient thermal resistance | Still Air | TQFP144 | 40.3 | °C/W |
| θ_{JC} | Junction-to-case thermal resistance | | TQFP144 | 9.5 | |
| θ_{JA} | Junction-to-ambient thermal resistance | Still Air | TFBGA144 | 28.5 | °C/W |
| θ_{JC} | Junction-to-case thermal resistance | | TFBGA144 | 6.9 | |
| θ_{JA} | Junction-to-ambient thermal resistance | Still Air | VFBGA100 | 31.1 | °C/W |
| θ_{JC} | Junction-to-case thermal resistance | | VFBGA100 | 6.9 | |

8.1.2 Junction Temperature

The average chip-junction temperature, T_J , in °C can be obtained from the following:

1. $T_J = T_A + (P_D \times \theta_{JA})$
2. $T_J = T_A + (P_D \times (\theta_{HEATSINK} + \theta_{JC}))$

where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in [Table 8-1 on page 68](#).
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in [Table 8-1 on page 68](#).
- $\theta_{HEAT\ SINK}$ = cooling device thermal resistance (°C/W), provided in the device datasheet.
- P_D = device power consumption (W) estimated from data provided in the section ["Regulator characteristics" on page 43](#).
- T_A = ambient temperature (°C).

From the first equation, the user can derive the estimated lifetime of the chip and decide if a cooling device is necessary or not. If a cooling device is to be fitted on the chip, the second equation should be used to compute the resulting average chip-junction temperature T_J in °C.

8.2 Package Drawings

Figure 8-1. TFBGA 144 package drawing

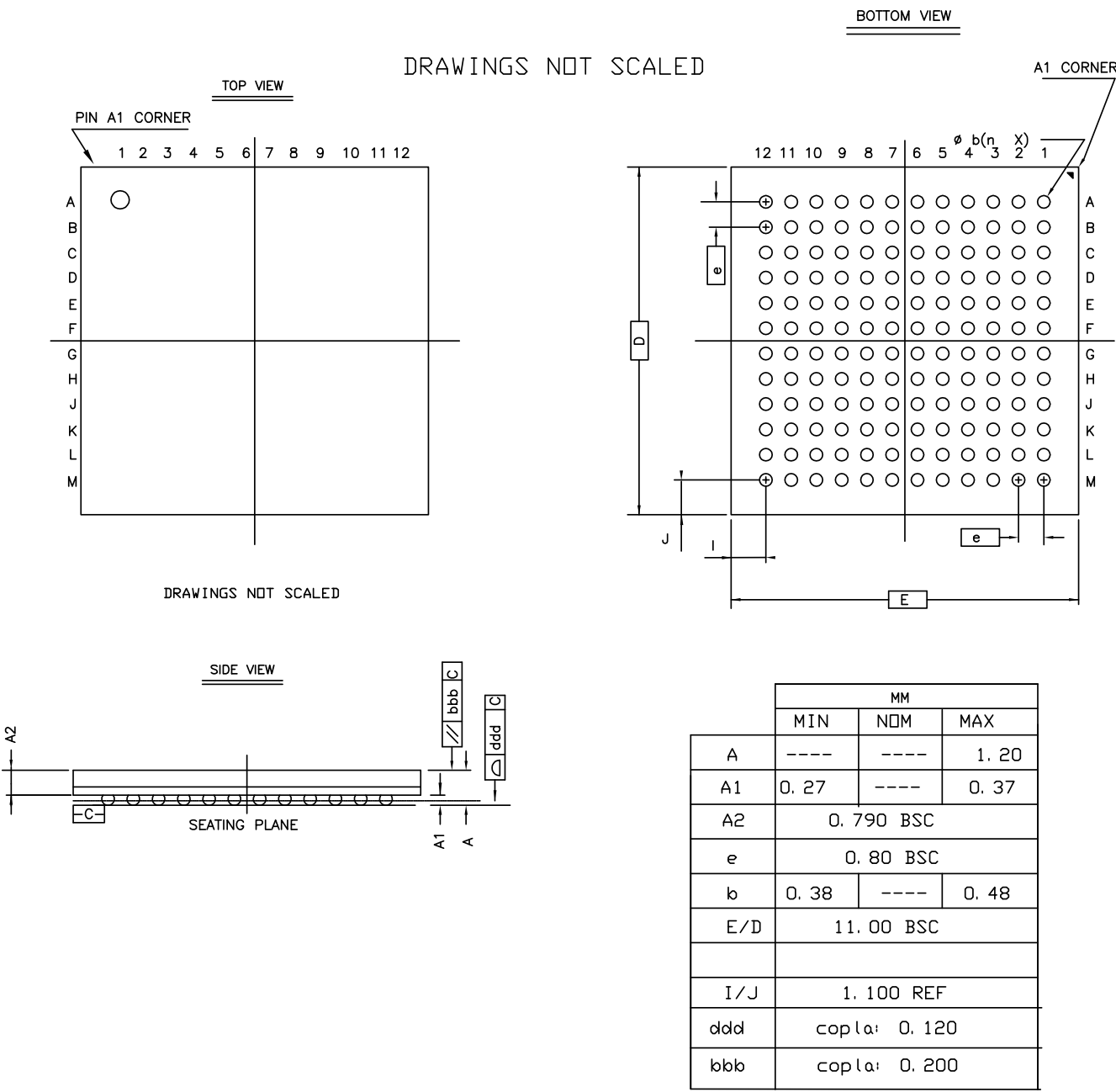
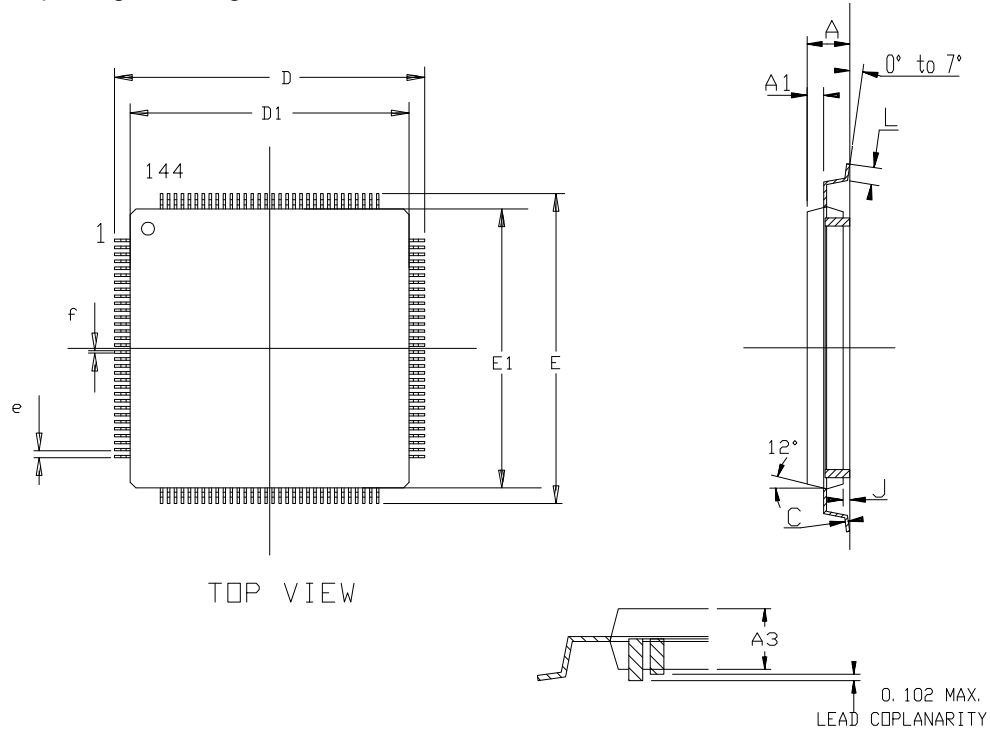


Figure 8-2. LQFP-144 package drawing



| | Min | MM Nom | Max | Min | INCH Nom | Max |
|----|----------|-----------|-------|-----------|-------------|------|
| A | – | – | 1.60 | – | – | .063 |
| C | 0.09 | – | 0.20 | .004 | – | .008 |
| A3 | 1.35 | 1.40 | 1.45 | .053 | .055 | .057 |
| D | 21.90 | 22.00 | 22.10 | .862 | .866 | .870 |
| D1 | 19.90 | 20.00 | 20.10 | .783 | .787 | .791 |
| E | 21.90 | 22.00 | 22.10 | .862 | .866 | .870 |
| E1 | 19.90 | 20.00 | 20.10 | .783 | .787 | .791 |
| J | 0.05 | – | 0.15 | .002 | – | .006 |
| L | 0.45 | 0.60 | 0.75 | .018 | .024 | .030 |
| e | 0.50 BSC | | | .0197 BSC | | |
| f | 0.22 BSC | | | .009 BSC | | |

Table 8-2. Device and Package Maximum Weight

| | |
|------|----|
| 1300 | mg |
|------|----|

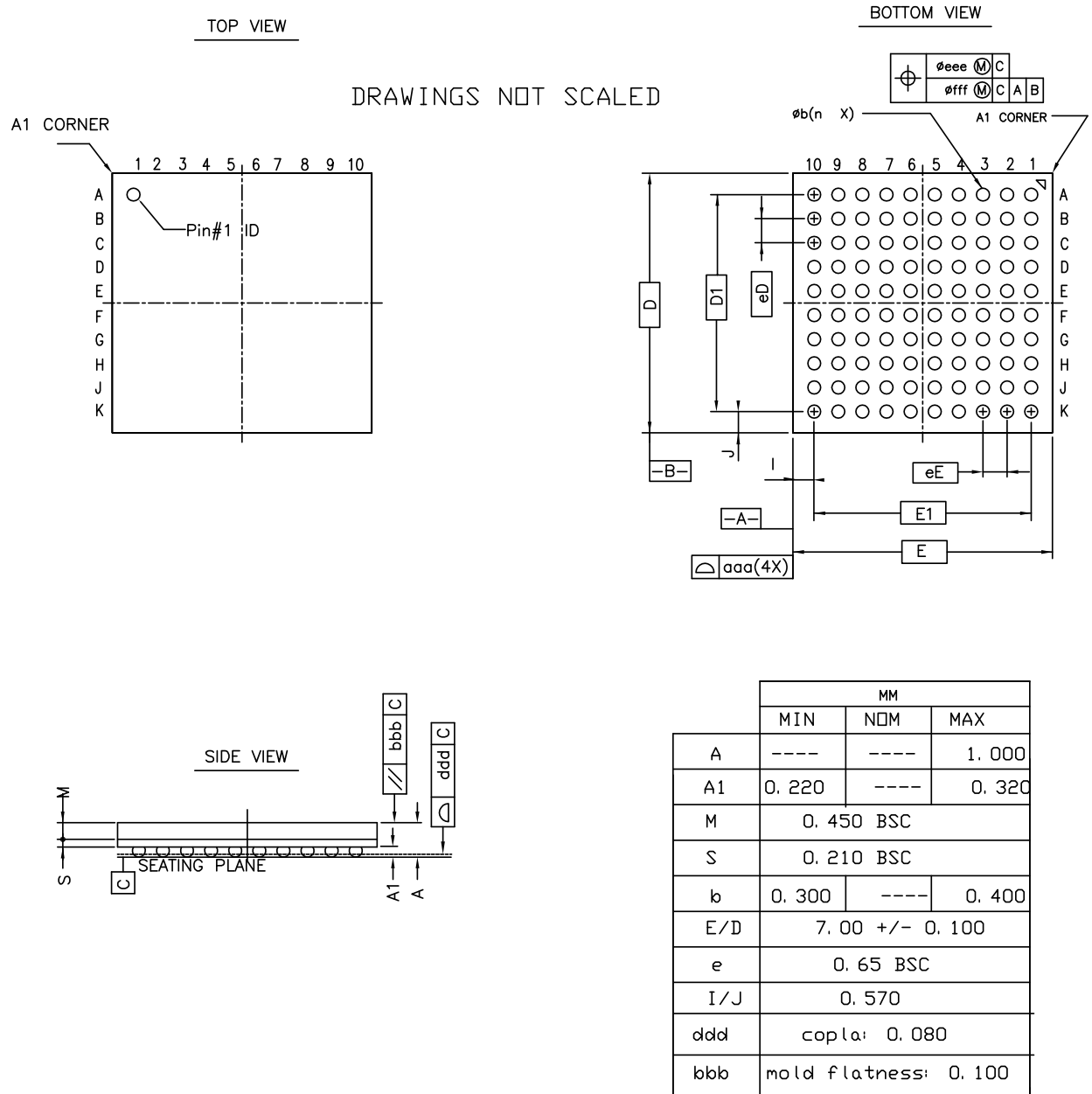
Table 8-3. Package Characteristics

| | |
|----------------------------|------|
| Moisture Sensitivity Level | MSL3 |
|----------------------------|------|

Table 8-4. Package Reference

| | |
|-------------------------|--------|
| JEDEC Drawing Reference | MS-026 |
| JESD97 Classification | E3 |

Figure 8-3. VFBGA-100 package drawing



8.3 Soldering Profile

Table 8-5 gives the recommended soldering profile from J-STD-20.

Table 8-5. Soldering Profile

| Profile Feature | Green Package |
|--|-----------------|
| Average Ramp-up Rate (217°C to Peak) | 3°C/Second max |
| Preheat Temperature 175°C ±25°C | 150-200°C |
| Time Maintained Above 217°C | 60-150 seconds |
| Time within 5°C of Actual Peak Temperature | 30 seconds |
| Peak Temperature Range | 260 (+0/-5°C) |
| Ramp-down Rate | 6°C/Second max. |
| Time 25°C to Peak Temperature | 8 minutes max |

Note: It is recommended to apply a soldering temperature higher than 250°C.
A maximum of three reflow passes is allowed per component.

9. Ordering Information

| Device | Ordering Code | Package | Conditioning | Temperature Operating Range |
|---------------|--------------------|----------------|--------------|-----------------------------|
| AT32UC3A3256S | AT32UC3A3256S-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3256S-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A3256S-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3256S-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A3256 | AT32UC3A3256-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3256-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A3256-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3256-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A3128S | AT32UC3A3128S-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3128S-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A3128S-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3128S-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A3128 | AT32UC3A3128-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3128-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A3128-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A3128-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A364S | AT32UC3A364S-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A364S-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A364S-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A364S-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A364 | AT32UC3A364-ALUT | 144-lead LQFP | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A364-ALUR | 144-lead LQFP | Reels | Industrial (-40-C to 85-C) |
| | AT32UC3A364-CTUT | 144-ball TFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A364-CTUR | 144-ball TFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A4256S | AT32UC3A4256S-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A4256S-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A4256 | AT32UC3A4256-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A4256-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A4128S | AT32UC3A4128S-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A4128S-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A4128 | AT32UC3A4128-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A4128-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A464S | AT32UC3A464S-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A464S-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |
| AT32UC3A464 | AT32UC3A464-C1UT | 100-ball VFBGA | Tray | Industrial (-40-C to 85-C) |
| | AT32UC3A464-C1UR | 100-ball VFBGA | Reels | Industrial (-40-C to 85-C) |

10. Errata

10.1 Rev. H

10.1.1 General

DMACA data transfer fails when CTLx.SRC_TR_WIDTH is not equal to CTLx.DST_TR_WIDTH

Fix/Workaround

For any DMACA transfer make sure CTLx.SRC_TR_WIDTH = CTLx.DST_TR_WIDTH.

10.1.2 Processor and Architecture

LDM instruction with PC in the register list and without ++ increments Rp

For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, ie the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.

Fix/Workaround

None.

Hardware breakpoints may corrupt MAC results

Hardware breakpoints on MAC instructions may corrupt the destination register of the MAC instruction.

Fix/Workaround

Place breakpoints on earlier or later instructions.

When the main clock is RCSYS, TIMER_CLOCK5 is equal to PBA clock

When the main clock is generated from RCSYS, TIMER_CLOCK5 is equal to PBA Clock and not PBA Clock / 128.

Fix/Workaround

None.

MPU

Privilege violation when using interrupts in application mode with protected system stack

If the system stack is protected by the MPU and an interrupt occurs in application mode, an MPU DTLB exception will occur.

Fix/Workaround

Make a DTLB Protection (Write) exception handler which permits the interrupt request to be handled in privileged mode.

10.1.3 USB

UPCFGn.INTFRQ is irrelevant for isochronous pipe

As a consequence, isochronous IN and OUT tokens are sent every 1 ms (Full Speed), or every 125uS (High Speed).

Fix/Workaround

For higher polling time, the software must freeze the pipe for the desired period in order to prevent any "extra" token.

10.1.4 ADC

Sleep Mode activation needs additional A to D conversion

If the ADC sleep mode is activated when the ADC is idle the ADC will not enter sleep mode before after the next AD conversion.

Fix/Workaround

Activate the sleep mode in the mode register and then perform an AD conversion.

10.1.5 USART

ISO7816 info register US_NER cannot be read

The NER register always returns zero.

Fix/Workaround

None.

The LIN ID is not transmitted in mode PDCM='0'

Fix/Workaround

Using USART in mode LIN master with the PDCM bit = '0', the LINID written at the first address of the transmit buffer is not used. The LINID must be written in the LINIR register, after the configuration and start of the PDCA transfer. Writing the LINID in the LINIR register will start the transfer whenever the PDCA transfer is ready.

The LINID interrupt is only available for the header reception and not available for the header transmission

Fix/Workaround

None.

USART LIN mode is not functional with the PDCA if PDCM bit in LINMR register is set to 1

If a PDCA transfer is initiated in USART LIN mode with PDCM bit set to 1, the transfer never starts.

Fix/Workaround

Only use PDCM=0 configuration with the PDCA transfer.

SPI

SPI disable does not work in SLAVE mode

SPI disable does not work in SLAVE mode.

Fix/Workaround

Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0

When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.

Fix/Workaround

When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0

When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.

Fix/Workaround

Disable mode fault detection by writing a one to MR.MODFDIS.

Disabling SPI has no effect on the SR.TDRE bit

Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.

Fix/Workaround

Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

Power Manager

Clock sources will not be stopped in STATIC sleep mode if the difference between CPU and PBx division factor is too high

If the division factor between the CPU/HSB and PBx frequencies is more than 4 when going to a sleep mode where the system RC oscillator is turned off, then high speed clock sources will not be turned off. This will result in a significantly higher power consumption during the sleep mode.

Fix/Workaround

Before going to sleep modes where the system RC oscillator is stopped, make sure that the factor between the CPU/HSB and PBx frequencies is less than or equal to 4.

10.1.6 PDCA

PCONTROL.CHxRES is non-functional

PCONTROL.CHxRES is non-functional. Counters are reset at power-on, and cannot be reset by software.

Fix/Workaround

Software needs to keep history of performance counters.

Transfer error will stall a transmit peripheral handshake interface

If a transfer error is encountered on a channel transmitting to a peripheral, the peripheral handshake of the active channel will stall and the PDCA will not do any more transfers on the affected peripheral handshake interface.

Fix/Workaround

Disable and then enable the peripheral after the transfer error.

AES

URAD (Unspecified Register Access Detection Status) does not detect read accesses to the write-only KEYW[5..8]R registers

Fix/Workaround

None.

10.1.7 HMATRIX

In the PRAS and PRBS registers, the MxPR fields are only two bits

In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.

Fix/Workaround

Mask undefined bits when reading PRAS and PRBS.

10.1.8 TWIM

TWIM SR.IDLE goes high immediately when NAK is received

When a NAK is received and there is a non-zero number of bytes to be transmitted, SR.IDLE goes high immediately and does not wait for the STOP condition to be sent. This does not cause any problem just by itself, but can cause a problem if software waits for SR.IDLE to go high and then immediately disables the TWIM by writing a one to CR.MDIS. Disabling the TWIM causes the TWCK and TWD pins to go high immediately, so the STOP condition will not be transmitted correctly.

Fix/Workaround

If possible, do not disable the TWIM. If it is absolutely necessary to disable the TWIM, there must be a software delay of at least two TWCK periods between the detection of SR.IDLE==1 and the disabling of the TWIM.

TWIM TWALM polarity is wrong

The TWALM signal in the TWIM is active high instead of active low.

Fix/Workaround

Use an external inverter to invert the signal going into the TWIM. When using both TWIM and TWIS on the same pins, the TWALM cannot be used.

SMBALERT bit may be set after reset

The SMBus Alert (SMBALERT) bit in the Status Register (SR) might be erroneously set after system reset.

Fix/Workaround

After system reset, clear the SR.SMBALERT bit before commencing any TWI transfer.

TWIS

Clearing the NAK bit before the BTF bit is set locks up the TWI bus

When the TWIS is in transmit mode, clearing the NAK Received (NAK) bit of the Status Register (SR) before the end of the Acknowledge/Not Acknowledge cycle will cause the TWIS to attempt to continue transmitting data, thus locking up the bus.

Fix/Workaround

Clear SR.NAK only after the Byte Transfer Finished (BTF) bit of the same register has been set.

TWIS stretch on Address match error

When the TWIS stretches TWCK due to a slave address match, it also holds TWD low for the same duration if it is to be receiving data. When TWIS releases TWCK, it releases TWD at the same time. This can cause a TWI timing violation.

Fix/Workaround

None.

SSC

Frame Synchro and Frame Synchro Data are delayed by one clock cycle

The frame synchro and the frame synchro data are delayed from 1 SSC_CLOCK when:

- Clock is CKDIV
- The START is selected on either a frame synchro edge or a level
- Frame synchro data is enabled
- Transmit clock is gated on output (through CKO field)

Fix/Workaround

Transmit or receive CLOCK must not be gated (by the mean of CKO field) when START condition is performed on a generated frame synchro.

10.1.9 FLASHC

Corrupted read in flash may happen after fuses write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands)

After a flash fuse write or erase operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands), reading (data read or code fetch) in flash may fail. This may lead to an exception or to other errors derived from this corrupted read access.

Fix/Workaround

Before the flash fuse write or erase operation, enable the flash high speed mode (FLASHC HSEN command). The flash fuse write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands) must be issued from RAM or through the EBI. After these commands, read 3 times one flash page initialized to 00h. Disable the flash high speed mode (FLASHC HSDIS command). It is then possible to safely read or code fetch the flash.

10.2 Rev. E

10.2.1 General

Increased Power Consumption in VDDIO in sleep modes

If the 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

Disable the OSC0 through the System Control Interface (SCIF) before going to any sleep mode where the OSC0 is disabled, or pull down or up XIN0 and XOUT0 with 1 Mohm resistor.

Power consumption in static mode The power consumption in static mode can be up to 330µA on some parts (typical at 25°C)

Fix/Workaround

Set to 1b bit CORRS4 of the ECCHRS mode register (MD). In C-code: *((volatile int*)(0xFFFE2404))= 0x400.

DMACA data transfer fails when CTLx.SRC_TR_WIDTH is not equal to CTLx.DST_TR_WIDTH

Fix/Workaround

For any DMACA transfer make sure CTLx.SRC_TR_WIDTH = CTLx.DST_TR_WIDTH.

3.3V supply monitor is not available

FGPFRLO[30:29] are reserved and should not be used by the application.

Fix/Workaround

None.

Service access bus (SAB) can not access DMACA registers

Fix/Workaround

None.

Processor and Architecture

LDM instruction with PC in the register list and without ++ increments Rp

For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, ie the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.

Fix/Workaround

None.

Hardware breakpoints may corrupt MAC results

Hardware breakpoints on MAC instructions may corrupt the destination register of the MAC instruction.

Fix/Workaround

Place breakpoints on earlier or later instructions.

When the main clock is RCSYS, TIMER_CLOCK5 is equal to PBA clock

When the main clock is generated from RCSYS, TIMER_CLOCK5 is equal to PBA Clock and not PBA Clock / 128.

Fix/Workaround

None.

MPU

Privilege violation when using interrupts in application mode with protected system stack

If the system stack is protected by the MPU and an interrupt occurs in application mode, an MPU DTLB exception will occur.

Fix/Workaround

Make a DTLB Protection (Write) exception handler which permits the interrupt request to be handled in privileged mode.

10.2.2 USB

UPCFGn.INTFRQ is irrelevant for isochronous pipe

As a consequence, isochronous IN and OUT tokens are sent every 1 ms (Full Speed), or every 125uS (High Speed).

Fix/Workaround

For higher polling time, the software must freeze the pipe for the desired period in order to prevent any "extra" token.

10.2.3 ADC

Sleep Mode activation needs additional A to D conversion

If the ADC sleep mode is activated when the ADC is idle the ADC will not enter sleep mode before after the next AD conversion.

Fix/Workaround

Activate the sleep mode in the mode register and then perform an AD conversion.

10.2.4 USART

ISO7816 info register US_NER cannot be read

The NER register always returns zero.

Fix/Workaround

None.

The LIN ID is not transmitted in mode PDCM='0'

Fix/Workaround

Using USART in mode LIN master with the PDCM bit = '0', the LINID written at the first address of the transmit buffer is not used. The LINID must be written in the LINIR register,

after the configuration and start of the PDCA transfer. Writing the LINID in the LINIR register will start the transfer whenever the PDCA transfer is ready.

The LINID interrupt is only available for the header reception and not available for the header transmission

Fix/Workaround

None.

USART LIN mode is not functional with the PDCA if PDCM bit in LINMR register is set to 1

If a PDCA transfer is initiated in USART LIN mode with PDCM bit set to 1, the transfer never starts.

Fix/Workaround

Only use PDCM=0 configuration with the PDCA transfer.

The RTS output does not function correctly in hardware handshaking mode

The RTS signal is not generated properly when the USART receives data in hardware handshaking mode. When the Peripheral DMA receive buffer becomes full, the RTS output should go high, but it will stay low.

Fix/Workaround

Do not use the hardware handshaking mode of the USART. If it is necessary to drive the RTS output high when the Peripheral DMA receive buffer becomes full, use the normal mode of the USART. Configure the Peripheral DMA Controller to signal an interrupt when the receive buffer is full. In the interrupt handler code, write a one to the RTSDIS bit in the USART Control Register (CR). This will drive the RTS output high. After the next DMA transfer is started and a receive buffer is available, write a one to the RTSEN bit in the USART CR so that RTS will be driven low.

ISO7816 Mode T1: RX impossible after any TX

RX impossible after any TX.

Fix/Workaround

SOFT_RESET on RX+ Config US_MR + Config_US_CR.

SPI

SPI disable does not work in SLAVE mode

SPI disable does not work in SLAVE mode.

Fix/Workaround

Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0

When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.

Fix/Workaround

When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0

When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.

Fix/Workaround

Disable mode fault detection by writing a one to MR.MODFDIS.

Disabling SPI has no effect on the SR.TDRE bit

Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.

Fix/Workaround

Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

Power Manager**Clock sources will not be stopped in STATIC sleep mode if the difference between CPU and PBx division factor is too high**

If the division factor between the CPU/HSB and PBx frequencies is more than 4 when going to a sleep mode where the system RC oscillator is turned off, then high speed clock sources will not be turned off. This will result in a significantly higher power consumption during the sleep mode.

Fix/Workaround

Before going to sleep modes where the system RC oscillator is stopped, make sure that the factor between the CPU/HSB and PBx frequencies is less than or equal to 4.

10.2.5 PDCA**PCONTROL.CHxRES is non-functional**

PCONTROL.CHxRES is non-functional. Counters are reset at power-on, and cannot be reset by software.

Fix/Workaround

Software needs to keep history of performance counters.

Transfer error will stall a transmit peripheral handshake interface

If a transfer error is encountered on a channel transmitting to a peripheral, the peripheral handshake of the active channel will stall and the PDCA will not do any more transfers on the affected peripheral handshake interface.

Fix/Workaround

Disable and then enable the peripheral after the transfer error.

AES**URAD (Unspecified Register Access Detection Status) does not detect read accesses to the write-only KEYW[5..8]R registers****Fix/Workaround**

None.

10.2.6 HMATRIX**In the PRAS and PRBS registers, the MxPR fields are only two bits**

In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.

Fix/Workaround

Mask undefined bits when reading PRAS and PRBS.

10.2.7 TWIM

TWIM SR.IDLE goes high immediately when NAK is received

When a NAK is received and there is a non-zero number of bytes to be transmitted, SR.IDLE goes high immediately and does not wait for the STOP condition to be sent. This does not cause any problem just by itself, but can cause a problem if software waits for SR.IDLE to go high and then immediately disables the TWIM by writing a one to CR.MDIS. Disabling the TWIM causes the TWCK and TWD pins to go high immediately, so the STOP condition will not be transmitted correctly.

Fix/Workaround

If possible, do not disable the TWIM. If it is absolutely necessary to disable the TWIM, there must be a software delay of at least two TWCK periods between the detection of SR.IDLE==1 and the disabling of the TWIM.

TWIM TWALM polarity is wrong

The TWALM signal in the TWIM is active high instead of active low.

Fix/Workaround

Use an external inverter to invert the signal going into the TWIM. When using both TWIM and TWIS on the same pins, the TWALM cannot be used.

SMBALERT bit may be set after reset

The SMBus Alert (SMBALERT) bit in the Status Register (SR) might be erroneously set after system reset.

Fix/Workaround

After system reset, clear the SR.SMBALERT bit before commencing any TWI transfer.

TWIS

Clearing the NAK bit before the BTF bit is set locks up the TWI bus

When the TWIS is in transmit mode, clearing the NAK Received (NAK) bit of the Status Register (SR) before the end of the Acknowledge/Not Acknowledge cycle will cause the TWIS to attempt to continue transmitting data, thus locking up the bus.

Fix/Workaround

Clear SR.NAK only after the Byte Transfer Finished (BTF) bit of the same register has been set.

TWIS stretch on Address match error

When the TWIS stretches TWCK due to a slave address match, it also holds TWD low for the same duration if it is to be receiving data. When TWIS releases TWCK, it releases TWD at the same time. This can cause a TWI timing violation.

Fix/Workaround

None.

MCI

MCI_CLK features is not available on PX12, PX13 and PX40**Fix/Workaround**

MCI_CLK feature is available on PA27 only.

The busy signal of the responses R1b is not taken in account (excepting for CMD12 STOP_TRANSFER)

It is not possible to know the busy status of the card during the response (R1b) for the commands CMD7, CMD28, CMD29, CMD38, CMD42, CMD56.

Fix/Workaround

The card busy line should be polled through the GPIO pin for commands CMD7, CMD28, CMD29, CMD38, CMD42 and CMD56. The GPIO alternate configuration should be restored after.

SSC

Frame Synchro and Frame Synchro Data are delayed by one clock cycle

The frame synchro and the frame synchro data are delayed from 1 SSC_CLOCK when:

- Clock is CKDIV
- The START is selected on either a frame synchro edge or a level
- Frame synchro data is enabled
- Transmit clock is gated on output (through CKO field)

Fix/Workaround

Transmit or receive CLOCK must not be gated (by the mean of CKO field) when START condition is performed on a generated frame synchro.

10.2.8 FLASHC

Corrupted read in flash may happen after fuses write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands)

After a flash fuse write or erase operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands), reading (data read or code fetch) in flash may fail. This may lead to an exception or to other errors derived from this corrupted read access.

Fix/Workaround

Before the flash fuse write or erase operation, enable the flash high speed mode (FLASHC HSEN command). The flash fuse write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands) must be issued from RAM or through the EBI. After these commands, read 3 times one flash page initialized to 00h. Disable the flash high speed mode (FLASHC HSDIS command). It is then possible to safely read or code fetch the flash.

10.3 Rev. D

10.3.1 General

DMACA data transfer fails when CTLx.SRC_TR_WIDTH is not equal to CTLx.DST_TR_WIDTH

Fix/Workaround

For any DMACA transfer make sure CTLx.SRC_TR_WIDTH = CTLx.DST_TR_WIDTH.

3.3V supply monitor is not available

FGPFRLO[30:29] are reserved and should not be used by the application.

Fix/Workaround

None.

Service access bus (SAB) can not access DMACA registers

Fix/Workaround

None.

Processor and Architecture

LDM instruction with PC in the register list and without ++ increments Rp

For LDM with PC in the register list: the instruction behaves as if the ++ field is always set, ie the pointer is always updated. This happens even if the ++ field is cleared. Specifically, the increment of the pointer is done in parallel with the testing of R12.

Fix/Workaround

None.

Hardware breakpoints may corrupt MAC results

Hardware breakpoints on MAC instructions may corrupt the destination register of the MAC instruction.

Fix/Workaround

Place breakpoints on earlier or later instructions.

When the main clock is RCSYS, TIMER_CLOCK5 is equal to PBA clock

When the main clock is generated from RCSYS, TIMER_CLOCK5 is equal to PBA Clock and not PBA Clock / 128.

Fix/Workaround

None.

RETE instruction does not clear SREG[L] from interrupts

The RETE instruction clears SREG[L] as expected from exceptions.

Fix/Workaround

When using the STCOND instruction, clear SREG[L] in the stacked value of SR before returning from interrupts with RETE.

RETS behaves incorrectly when MPU is enabled

RETS behaves incorrectly when MPU is enabled and MPU is configured so that system stack is not readable in unprivileged mode.

Fix/Workaround

Make system stack readable in unprivileged mode, or return from supervisor mode using rete instead of rets. This requires:

1. Changing the mode bits from 001 to 110 before issuing the instruction. Updating the mode bits to the desired value must be done using a single mtsr instruction so it is done atomically. Even if this step is generally described as not safe in the UC technical reference manual, it is safe in this very specific case.
2. Execute the RETE instruction.

In the PRAS and PRBS registers, the MxPR fields are only two bits

In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.

Fix/Workaround

Mask undefined bits when reading PRAS and PRBS.

Multiply instructions do not work on RevD

All the multiply instructions do not work.

Fix/Workaround

Do not use the multiply instructions.

MPU

Privilege violation when using interrupts in application mode with protected system stack

If the system stack is protected by the MPU and an interrupt occurs in application mode, an MPU DTLB exception will occur.

Fix/Workaround

Make a DTLB Protection (Write) exception handler which permits the interrupt request to be handled in privileged mode.

10.3.2 USB

UPCFGn.INTFRQ is irrelevant for isochronous pipe

As a consequence, isochronous IN and OUT tokens are sent every 1 ms (Full Speed), or every 125uS (High Speed).

Fix/Workaround

For higher polling time, the software must freeze the pipe for the desired period in order to prevent any "extra" token.

10.3.3 ADC

Sleep Mode activation needs additional A to D conversion

If the ADC sleep mode is activated when the ADC is idle the ADC will not enter sleep mode before after the next AD conversion.

Fix/Workaround

Activate the sleep mode in the mode register and then perform an AD conversion.

10.3.4 USART

ISO7816 info register US_NER cannot be read

The NER register always returns zero.

Fix/Workaround

None.

The LIN ID is not transmitted in mode PDCM='0'

Fix/Workaround

Using USART in mode LIN master with the PDCM bit = '0', the LINID written at the first address of the transmit buffer is not used. The LINID must be written in the LINIR register, after the configuration and start of the PDCA transfer. Writing the LINID in the LINIR register will start the transfer whenever the PDCA transfer is ready.

The LINID interrupt is only available for the header reception and not available for the header transmission

Fix/Workaround

None.

USART LIN mode is not functional with the PDCA if PDCM bit in LINMR register is set to 1

If a PDCA transfer is initiated in USART LIN mode with PDCM bit set to 1, the transfer never starts.

Fix/Workaround

Only use PDCM=0 configuration with the PDCA transfer.

The RTS output does not function correctly in hardware handshaking mode

The RTS signal is not generated properly when the USART receives data in hardware handshaking mode. When the Peripheral DMA receive buffer becomes full, the RTS output should go high, but it will stay low.

Fix/Workaround

Do not use the hardware handshaking mode of the USART. If it is necessary to drive the RTS output high when the Peripheral DMA receive buffer becomes full, use the normal

mode of the USART. Configure the Peripheral DMA Controller to signal an interrupt when the receive buffer is full. In the interrupt handler code, write a one to the RTSDIS bit in the USART Control Register (CR). This will drive the RTS output high. After the next DMA transfer is started and a receive buffer is available, write a one to the RTSEN bit in the USART CR so that RTS will be driven low.

ISO7816 Mode T1: RX impossible after any TX

RX impossible after any TX.

Fix/Workaround

SOFT_RESET on RX+ Config US_MR + Config_US_CR.

SPI

SPI disable does not work in SLAVE mode

SPI disable does not work in SLAVE mode.

Fix/Workaround

Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0

When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.

Fix/Workaround

When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0

When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.

Fix/Workaround

Disable mode fault detection by writing a one to MR.MODFDIS.

Disabling SPI has no effect on the SR.TDRE bit

Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.

Fix/Workaround

Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

Power Manager

Clock sources will not be stopped in STATIC sleep mode if the difference between CPU and PBx division factor is too high

If the division factor between the CPU/HSB and PBx frequencies is more than 4 when going to a sleep mode where the system RC oscillator is turned off, then high speed clock sources will not be turned off. This will result in a significantly higher power consumption during the sleep mode.

Fix/Workaround

Before going to sleep modes where the system RC oscillator is stopped, make sure that the factor between the CPU/HSB and PBx frequencies is less than or equal to 4.

10.3.5 PDCA

PCONTROL.CHxRES is non-functional

PCONTROL.CHxRES is non-functional. Counters are reset at power-on, and cannot be reset by software.

Fix/Workaround

Software needs to keep history of performance counters.

Transfer error will stall a transmit peripheral handshake interface

If a transfer error is encountered on a channel transmitting to a peripheral, the peripheral handshake of the active channel will stall and the PDCA will not do any more transfers on the affected peripheral handshake interface.

Fix/Workaround

Disable and then enable the peripheral after the transfer error.

AES

URAD (Unspecified Register Access Detection Status) does not detect read accesses to the write-only KEYW[5..8]R registers

Fix/Workaround

None.

10.3.6 HMATRIX

In the PRAS and PRBS registers, the MxPR fields are only two bits

In the PRAS and PRBS registers, the MxPR fields are only two bits wide, instead of four bits. The unused bits are undefined when reading the registers.

Fix/Workaround

Mask undefined bits when reading PRAS and PRBS.

10.3.7 TWIM

TWIM SR.IDLE goes high immediately when NAK is received

When a NAK is received and there is a non-zero number of bytes to be transmitted, SR.IDLE goes high immediately and does not wait for the STOP condition to be sent. This does not cause any problem just by itself, but can cause a problem if software waits for SR.IDLE to go high and then immediately disables the TWIM by writing a one to CR.MDIS. Disabling the TWIM causes the TWCK and TWD pins to go high immediately, so the STOP condition will not be transmitted correctly.

Fix/Workaround

If possible, do not disable the TWIM. If it is absolutely necessary to disable the TWIM, there must be a software delay of at least two TWCK periods between the detection of SR.IDLE==1 and the disabling of the TWIM.

TWIM TWALM polarity is wrong

The TWALM signal in the TWIM is active high instead of active low.

Fix/Workaround

Use an external inverter to invert the signal going into the TWIM. When using both TWIM and TWIS on the same pins, the TWALM cannot be used.

TWIS

TWIS Version Register reads zero

TWIS Version Register (VR) reads zero instead of 0x112.

Fix/Workaround

None.

10.3.8 MCI

The busy signal of the responses R1b is not taken in account (excepting for CMD12 STOP_TRANSFER)

It is not possible to know the busy status of the card during the response (R1b) for the commands CMD7, CMD28, CMD29, CMD38, CMD42, CMD56.

Fix/Workaround

The card busy line should be polled through the GPIO pin for commands CMD7, CMD28, CMD29, CMD38, CMD42 and CMD56. The GPIO alternate configuration should be restored after.

10.3.9 SSC

Frame Synchro and Frame Synchro Data are delayed by one clock cycle

The frame synchro and the frame synchro data are delayed from 1 SSC_CLOCK when:

- Clock is CKDIV
- The START is selected on either a frame synchro edge or a level
- Frame synchro data is enabled
- Transmit clock is gated on output (through CKO field)

Fix/Workaround

Transmit or receive CLOCK must not be gated (by the mean of CKO field) when START condition is performed on a generated frame synchro.

10.3.10 FLASHC

Corrupted read in flash may happen after fuses write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands)

After a flash fuse write or erase operation (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands), reading (data read or code fetch) in flash may fail. This may lead to an exception or to other errors derived from this corrupted read access.

Fix/Workaround

Before the flash fuse write or erase operation, enable the flash high speed mode (FLASHC HSEN command). The flash fuse write or erase operations (FLASHC LP, UP, WGPB, EGPB, SSB, PGPFB, EAGPF commands) must be issued from RAM or through the EBI. After these commands, read 3 times one flash page initialized to 00h. Disable the flash high speed mode (FLASHC HSDIS command). It is then possible to safely read or code fetch the flash.

11. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

11.1 Rev. G– 11/11

1. Add recommendation for MCI connection with more than 1 slot

11.2 Rev. F – 08/11

1. Final version

11.3 Rev. E – 06/11

1. Updated Errata for E and D
2. Updated FLASHC chapter with HSEN and HSDIS commands

11.4 Rev. D – 04/11

1. Updated Errata for revision H and E
2. Updated Reset Sequence
3. Updated Peripherals' current consumption and others minor electrical characteristics
4. Updated Peripherals chapters

11.5 Rev. C – 03/10

1. Updated the datasheet with new revision H features.

11.6 Rev. B – 08/09

1. Updated the datasheet with new device AT32UC3A4.

11.7 Rev. A – 03/09

1. Initial revision.

| | | |
|----------|--|-----------|
| 1 | Description | 3 |
| 2 | Overview | 4 |
| 2.1 | Block Diagram | 4 |
| 2.2 | Configuration Summary | 5 |
| 3 | Package and Pinout | 6 |
| 3.1 | Package | 6 |
| 3.2 | Peripheral Multiplexing on I/O lines | 9 |
| 3.3 | Signal Descriptions | 14 |
| 3.4 | I/O Line Considerations | 19 |
| 3.5 | Power Considerations | 20 |
| 4 | Processor and Architecture | 21 |
| 4.1 | Features | 21 |
| 4.2 | AVR32 Architecture | 21 |
| 4.3 | The AVR32UC CPU | 22 |
| 4.4 | Programming Model | 26 |
| 4.5 | Exceptions and Interrupts | 30 |
| 5 | Memories | 34 |
| 5.1 | Embedded Memories | 34 |
| 5.2 | Physical Memory Map | 34 |
| 5.3 | Peripheral Address Map | 35 |
| 5.4 | CPU Local Bus Mapping | 37 |
| 6 | Boot Sequence | 39 |
| 6.1 | Starting of Clocks | 39 |
| 6.2 | Fetching of Initial Instructions | 39 |
| 7 | Electrical Characteristics | 40 |
| 7.1 | Absolute Maximum Ratings* | 40 |
| 7.2 | DC Characteristics | 41 |
| 7.3 | I/O pin Characteristics | 42 |
| 7.4 | Regulator characteristics | 43 |
| 7.5 | Analog characteristics | 44 |
| 7.6 | Power Consumption | 48 |
| 7.7 | System Clock Characteristics | 51 |
| 7.8 | Oscillator Characteristics | 52 |
| 7.9 | ADC Characteristics | 54 |

| | | |
|-----------|--|-----------|
| 7.10 | USB Transceiver Characteristics | 55 |
| 7.11 | EBI Timings | 57 |
| 7.12 | JTAG Characteristics | 63 |
| 7.13 | SPI Characteristics | 64 |
| 7.14 | MCI | 66 |
| 7.15 | Flash Memory Characteristics | 67 |
| 8 | <i>Mechanical Characteristics</i> | 68 |
| 8.1 | Thermal Considerations | 68 |
| 8.2 | Package Drawings | 69 |
| 8.3 | Soldering Profile | 72 |
| 9 | <i>Ordering Information</i> | 73 |
| 10 | <i>Errata</i> | 74 |
| 10.1 | Rev. H | 74 |
| 10.2 | Rev. E | 78 |
| 10.3 | Rev. D | 83 |
| 11 | <i>Datasheet Revision History</i> | 89 |
| 11.1 | Rev. G– 11/11 | 89 |
| 11.2 | Rev. F – 08/11 | 89 |
| 11.3 | Rev. E – 06/11 | 89 |
| 11.4 | Rev. D – 04/11 | 89 |
| 11.5 | Rev. C – 03/10 | 89 |
| 11.6 | Rev. B – 08/09 | 89 |
| 11.7 | Rev. A – 03/09 | 89 |