Features

- High Performance, Low Power AVR®32 UC 32-bit Microcontroller
  - Compact Single-Cycle RISC Instruction Set Including DSP Instructions
  - Read-Modify-Write Instructions and Atomic Bit Manipulation
  - Performance
    - Up to 64 DMIPS Running at 50MHz from Flash (1 Flash Wait State)
    - Up to 36 DMIPS Running at 25MHz from Flash (0 Flash Wait State)
  - Memory Protection Unit
- picoPower™ Technology for Ultra-Low Power Consumption
- Multi-Hierarchy Bus System
  - High-Performance Data Transfers on Separate Buses for Increased Performance
  - 12 Peripheral DMA Channels Improve Speed for Peripheral Communication
- Internal High-Speed Flash
  - 64Kbytes, 32Kbytes, and 16Kbytes Versions
  - Single-Cycle Access up to 25MHz
  - FlashVault™ Technology Allows Pre-programmed Secure Library Support for End User Applications
  - Prefetch Buffer Optimizing Instruction Execution at Maximum Speed
  - 4ms Page Programming Time and 8ms 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, Single-Cycle Access at Full Speed
  - 16Kbytes (64Kbytes and 32Kbytes Flash), or 8Kbytes (16Kbytes Flash)
- Interrupt Controller (INTC)
  - Autovectored Low Latency Interrupt Service with Programmable Priority
- External Interrupt Controller (EIC)
- Peripheral Event System for Direct Peripheral to Peripheral Communication
- System Functions
  - Power and Clock Manager
  - SleepWalking™ Power Saving Control
  - Internal System RC Oscillator (RCSYS)
  - 32 KHz Oscillator
  - Multipurpose Oscillator and Digital Frequency Locked Loop (DFLL)
- Windowed Watchdog Timer (WDT)
- Asynchronous Timer (AST) with Real-Time Clock Capability
  - Counter or Calendar Mode Supported
- Frequency Meter (FREQM) for Accurate Measuring of Clock Frequency
- Six 16-bit Timer/Counter (TC) Channels
  - External Clock Inputs, PWM, Capture and Various Counting Capabilities
- PWM Channels on All I/O Pins (PWMA)
  - 8-bit PWM up to 150MHz Source Clock
- Four Universal Synchronous/Asynchronous Receiver/Transmitters (USART)
  - Independent Baudrate Generator, Support for SPI
  - Support for Hardware Handshaking
- One Master/Slave Serial Peripheral Interfaces (SPI) with Chip Select Signals
  - Up to 15 SPI Slaves can be Addressed
- Two Master and Two Slave Two-Wire Interfaces (TWI), 400kbit/s I^2C-compatible
- One 9-channel Analog-To-Digital Converter (ADC) with up to 12 Bits Resolution
  - Internal Temperature Sensor

AVR®32 32-bit Microcontroller

AT32UC3L064
AT32UC3L032
AT32UC3L016

Preliminary

Summary

32099AS–AVR32–06/09

- Eight Analog Comparators (AC) with Optional Window Detection
- Capacitive Touch (CAT) Module
  - Support QTouch™ and QMatrix™ Capture from Capacitive Touch Sensors
- On-Chip Non-Intrusive Debug System
  - Nexus Class 2+, Runtime Control, Non-Intrusive Data and Program Trace
  - aWire™ Single-Pin Programming Trace and Debug Interface Muxed with Reset Pin
  - NanoTrace™ Provides Trace Capabilities through JTAG or aWire Interface
- 48-pin TQFP/QFN/TLLGA (36 GPIO Pins)
- Five High-Drive I/O Pins
- Single 1.62-3.6V Power Supply

1. Description

The AT32UC3L is a complete System-On-Chip microcontroller based on the AVR32 UC RISC processor running at frequencies up to 50MHz. 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 capability is achieved using a rich set of DSP instructions.

The AT32UC3L embeds state-of-the-art picoPower technology for ultra-low power consumption. Combined power control techniques are used to bring active power as low as 0.5mW/MHz, and leakage down to 100nA while still retaining a bank of backup registers. The device allows a wide range of trade-offs between functionality and power consumption, giving the user the ability to reach the lowest possible power consumption with the feature set required for the application.

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

The AT32UC3L incorporates on-chip Flash and SRAM memories for secure and fast access. The FlashVault technology allows secure libraries to be programmed into the device. The secure libraries can be executed while the CPU is in Secure State, but not read by non-secure software in the device. The device can thus be shipped to end costumers, who will be able to program their own code into the device, accessing the secure libraries, but without risk of compromising the proprietary secure code.

The Peripheral Event System allows peripherals to receive, react to, and send peripheral events without CPU intervention. Asynchronous interrupts allow advanced peripheral operation in low power sleep modes.

The Power Manager improves design flexibility and security. The Power Manager supports SleepWalking functionality, by which a module can be selectively activated based on peripheral events, even in sleep modes where the module clock is stopped. Power monitoring is supported by on-chip Power-On Reset (POR), Brown-Out Detector (BOD), and Supply Monitor (SM). The device features several oscillators, such as Digital Frequency Locked Loop (DFLL), Oscillator 0 (OSC0), and system RC oscillator (RCSYS). Either of these oscillators can be used as source for the system clock. The DFLL is a programmable internal oscillator from 20 to 150MHz. It can be tuned to a high accuracy if an accurate oscillator is running, e.g. the 32KHz crystal oscillator.

The Watchdog Timer (WDT) will reset the device unless it is periodically serviced by the software. This allows the device to recover from a condition that has caused the system to be unstable.

The Asynchronous Timer (AST) combined with the 32KHz crystal oscillator supports powerful real-time clock capabilities, with a maximum timeout of up to 136 years. The AST can operate in counter mode or calendar mode.

The Frequency Meter (FREQM) allows accurate measuring of a clock frequency by comparing it to a known reference clock.

The device includes six 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.

The Pulse Width Modulation controller (PWMA) provides 8-bit PWM channels which can be synchronized and controlled from a common timer. One PWM channel is available for each I/O pin on the device, enabling applications that require multiple PWM outputs, such as LCD backlight control. The PWM channels can operate independently, with duty cycles set independently from each other, or in interlinked mode, with multiple channels changed at the same time.

The AT32UC3L also features many communication interfaces for communication intensive applications like USART, SPI, or TWI.

A general purpose 9-channel ADC is provided, as well as eight analog comparators (AC). The ADC can operate in 10-bit mode at full speed or in enhanced mode at reduced speed, offering up to 12-bit resolution. The ADC also provides an internal temperature sensor input channel. The analog comparators can be paired to detect when the sensing voltage is within or outside the defined reference window.

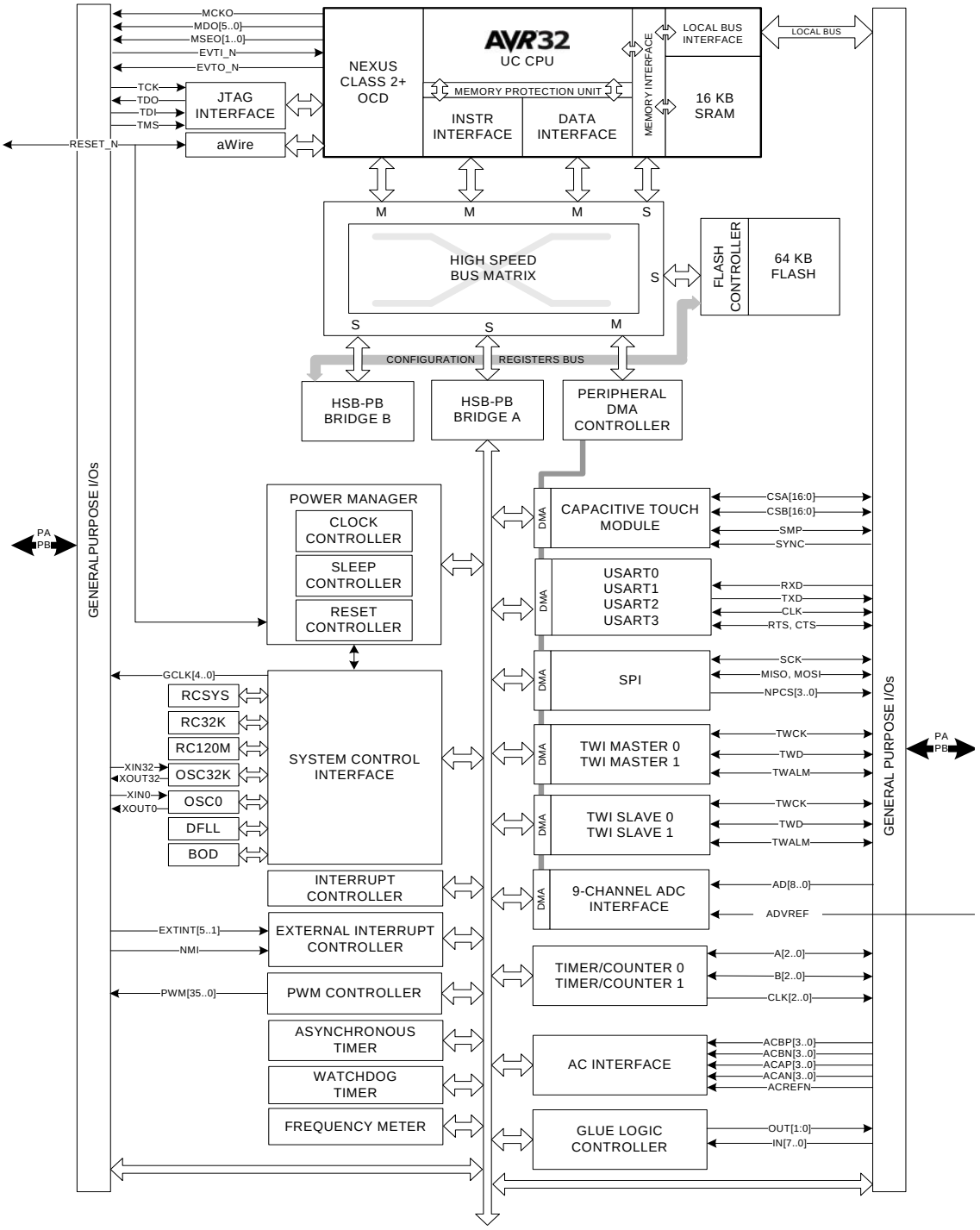
The Capacitive Touch (CAT) module senses touch on external capacitive touch sensors, using the QTouch technology. Capacitive touch sensors use no external mechanical components, unlike normal push buttons, and therefore demand less maintenance in the user application. The CAT module allows up to 17 touch sensors, or up to 18 by 8 matrix sensors to be interfaced. One touch sensor can be configured to operate autonomously without software interaction, allowing wakeup from sleep modes when activated.

The AT32UC3L 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. The NanoTrace interface enables trace feature for aWire- or JTAG-based debuggers. The single-pin aWire interface allows all features available through the JTAG interface to be accessed through the RESET pin, allowing the JTAG pins to be used for GPIO or peripherals.

2. Overview

2.1 Block Diagram

Figure 2-1. Block Diagram



2.2 Configuration Summary

Table 2-1. Configuration Summary

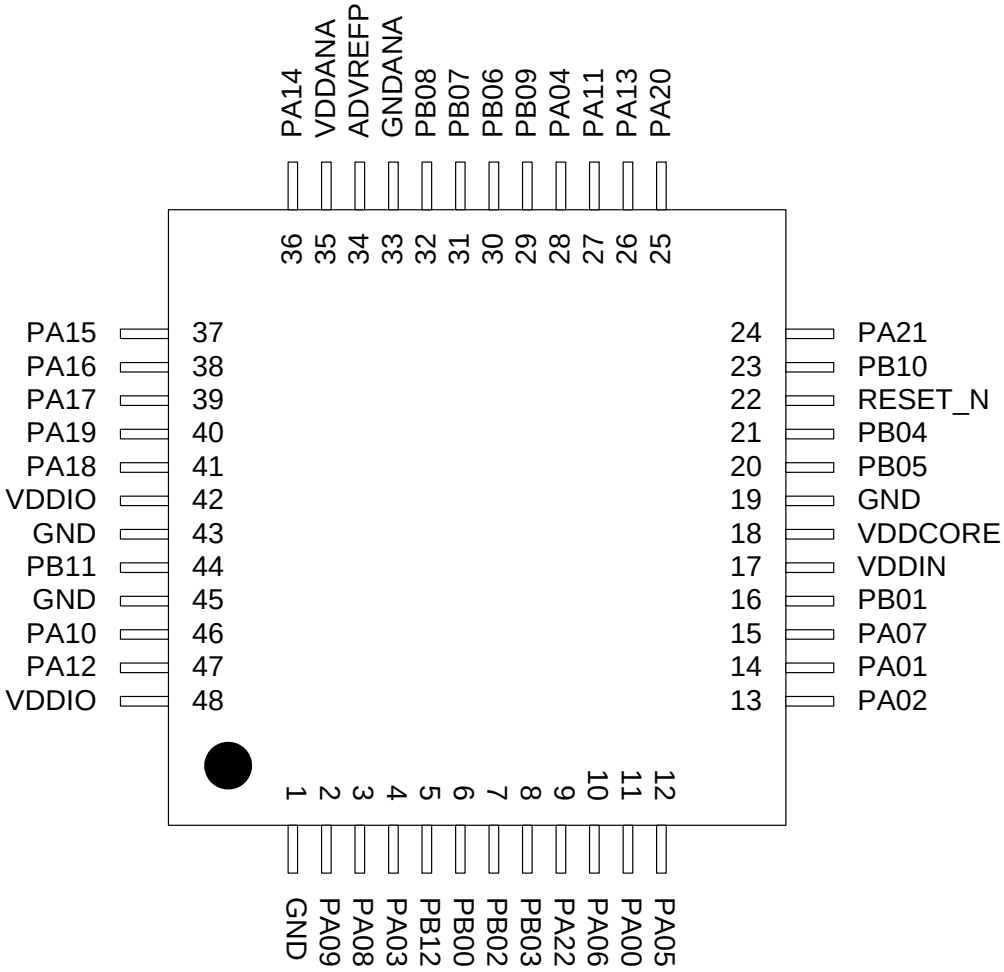
Feature	AT32UC3L0	AT32UC3L1	AT32UC3L2
Flash	64KB	32KB	16KB
SRAM	16KB	16KB	8KB
GPIO	36		
Hi-drive pins	5		
External Interrupts	8		
TWI	2		
USART	4		
Peripheral DMA Channels	12		
Peripheral Event System	1		
SPI	1		
Asynchronous Timers	1		
Timer/Counter Channels	6		
PWM channels	36		
Frequency Meter	1		
Watchdog Timer	1		
Power Manager	1		
Oscillators	Digital Frequency Locked Loop 20-150 MHz (DFLL) Crystal Oscillator 3-16 MHz (OSC0) Crystal Oscillator 32 KHz (OSC32K) RC Oscillator 120MHz (RC120M) RC Oscillator 115 kHz (RCSYS) RC Oscillator 32 kHz (RC32K)		
ADC	9 channel 10-bit		
Temperature Sensor	1		
Analog Comparators	8		
Capacitive Touch Module	1		
JTAG	1		
aWire	1		
Max Frequency	50 MHz		
Package	TQFP48/QFN48/TLLGA48		

3. Package and Pinout

3.1 Package

The device pins are multiplexed with peripheral functions as described in Section 3.2.

Figure 3-1. TQFP48/QFN48/TLLGA48 Pinout



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

QFP 48	PIN	GPIO	Supply	Pad Type	GPIO Function							
					A	B	C	D	E	F	G	H
11	PA00	0	VDDIO	Normal I/O	USART0-TXD	USART1-RTS	SPI-NPCS[2]		PWMA-PWMA[0]		SCIF-GCLK[0]	CAT-CSA[2]
14	PA01	1	VDDIO	Normal I/O	USART0-RXD	USART1-CTS	SPI-NPCS[3]	USART1-CLK	PWMA-PWMA[1]	ACIFB-ACAP[0]	TWIMS0-TWALM	CAT-CSA[1]
13	PA02	2	VDDIO	High-drive I/O	USART0-RTS	ADCIFB-TRIGGER	USART2-TXD	TC0-A0	PWMA-PWMA[2]	ACIFB-ACBP[0]	USART0-CLK	CAT-CSA[3]
4	PA03	3	VDDIO	Normal I/O	USART0-CTS	SPI-NPCS[1]	USART2-TXD	TC0-B0	PWMA-PWMA[3]	ACIFB-ACBN[3]	USART0-CLK	CAT-CSB[3]
28	PA04	4	VDDIO	Normal I/O	SPI-MISO	TWIMS0-TWCK	USART1-RXD	TC0-B1	PWMA-PWMA[4]	ACIFB-ACBP[1]		CAT-CSA[7]
12	PA05	5	VDDIO	TWI, Normal I/O	SPI-MOSI	TWIMS1-TWCK	USART1-TXD	TC0-A1	PWMA-PWMA[5]	ACIFB-ACBN[0]	TWIMS0-TWD	CAT-CSB[7]
10	PA06	6	VDDIO	High-drive I/O, 5V tolerant	SPI-SCK	USART2-TXD	USART1-CLK	TC0-B0	PWMA-PWMA[6]		SCIF-GCLK[1]	CAT-CSB[1]
15	PA07	7	VDDIO	TWI, Normal I/O	SPI-NPCS[0]	USART2-RXD	TWIMS1-TWALM	TWIMS0-TWCK	PWMA-PWMA[7]	ACIFB-ACAN[0]	EIC-EXTINT[0]	CAT-CSB[2]
3	PA08	8	VDDIO	High-drive I/O	USART1-TXD	SPI-NPCS[2]	TC0-A2	ADCIFB-ADP[0]	PWMA-PWMA[8]			CAT-CSA[4]
2	PA09	9	VDDIO	High-drive I/O	USART1-RXD	SPI-NPCS[3]	TC0-B2	ADCIFB-ADP[1]	PWMA-PWMA[9]	SCIF-GCLK[2]	EIC-EXTINT[1]	CAT-CSB[4]
46	PA10	10	VDDIO	Normal I/O	TWIMS0-TWD		TC0-A0		PWMA-PWMA[10]	ACIFB-ACAP[1]	SCIF-GCLK[2]	CAT-CSA[5]
27	PA11	11	VDDIN	Normal I/O					PWMA-PWMA[11]			
47	PA12	12	VDDIO	Normal I/O	ADCIFB-PRND	USART2-CLK	TC0-CLK1	CAT-SMP	PWMA-PWMA[12]	ACIFB-ACAN[1]	SCIF-GCLK[3]	CAT-CSB[5]
26	PA13	13	VDDIN	Normal I/O	GLOC-OUT[0]	GLOC-IN[7]	TC0-A0	SCIF-GCLK[2]	PWMA-PWMA[13]	CAT-SMP	EIC-EXTINT[2]	CAT-CSA[0]
36	PA14	14	VDDIO	Normal I/O	ADCIFB-AD[0]	TC0-CLK2	USART2-RTS	CAT-SMP	PWMA-PWMA[14]		SCIF-GCLK[4]	CAT-CSA[6]
37	PA15	15	VDDIO	Normal I/O	ADCIFB-AD[1]	TC0-CLK1		GLOC-IN[6]	PWMA-PWMA[15]	CAT-SYNC	EIC-EXTINT[3]	CAT-CSB[6]
38	PA16	16	VDDIO	Normal I/O	ADCIFB-AD[2]	TC0-CLK0		GLOC-IN[5]	PWMA-PWMA[16]	ACIFB-ACREFN	EIC-EXTINT[4]	CAT-CSA[8]
39	PA17	17	VDDIO	TWI, Normal I/O	ADCIFB-AD[3]	TC0-A1	USART2-CTS	TWIMS1-TWD	PWMA-PWMA[17]	CAT-SMP	CAT-DIS	CAT-CSB[8]
41	PA18	18	VDDIO	Normal I/O	ADCIFB-AD[4]	TC0-B1		GLOC-IN[4]	PWMA-PWMA[18]	CAT-SYNC	EIC-EXTINT[5]	CAT-CSB[0]
40	PA19	19	VDDIO	Normal I/O	ADCIFB-AD[5]		TC0-A2	TWIMS1-TWALM	PWMA-PWMA[19]		CAT-SYNC	CAT-CSA[10]
25	PA20	20	VDDIN	Normal I/O	USART2-TXD	TWIMS0-TWCK	TC0-A1	GLOC-IN[3]	PWMA-PWMA[20]	SCIF-RC32OUT		CAT-CSA[12]

Table 3-1. GPIO Controller Function Multiplexing

24	PA21	21	VDDIN	TWI, 5V tolerant, SMBus, Normal I/O	USART2-RXD	TWIMS0-TWD	TC0-B1	ADCIFB-TRIGGER	PWMA-PWMA[21]	PWMA-PWMAOD[21]	SCIF-GCLK[0]	CAT-SMP
9	PA22	22	VDDIO	Normal I/O	USART0-CTS	USART2-CLK	TC0-B2	CAT-SMP	PWMA-PWMA[22]	ACIFB-ACBN[2]		CAT-CSB[10]
6	PB00	32	VDDIO	Normal I/O	USART3-TXD	ADCIFB-ADP[0]	SPI-NPCS[0]	TC0-A1	PWMA-PWMA[23]	ACIFB-ACAP[2]	TC1-A0	CAT-CSA[9]
16	PB01	33	VDDIO	High-drive I/O	USART3-RXD	ADCIFB-ADP[1]	SPI-SCK	TC0-B1	PWMA-PWMA[24]		TC1-A1	CAT-CSB[9]
7	PB02	34	VDDIO	Normal I/O	USART3-RTS	USART3-CLK	SPI-MISO	TC0-A2	PWMA-PWMA[25]	ACIFB-ACAN[2]	SCIF-GCLK[1]	CAT-CSB[11]
8	PB03	35	VDDIO	Normal I/O	USART3-CTS	USART3-CLK	SPI-MOSI	TC0-B2	PWMA-PWMA[26]	ACIFB-ACBP[2]	TC1-A2	CAT-CSA[11]
21	PB04	36	VDDIN	TWI, 5V tolerant, SMBus, Normal I/O	TC1-A0	USART1-RTS	USART1-CLK	TWIMS0-TWALM	PWMA-PWMA[27]	PWMA-PWMAOD[27]	TWIMS1-TWCK	CAT-CSA[14]
20	PB05	37	VDDIN	TWI, 5V tolerant, SMBus, Normal I/O	TC1-B0	USART1-CTS	USART1-CLK	TWIMS0-TWCK	PWMA-PWMA[28]	PWMA-PWMAOD[28]	SCIF-GCLK[3]	CAT-CSB[14]
30	PB06	38	VDDIO	Normal I/O	TC1-A1	USART3-TXD	ADCIFB-AD[6]	GLOC-IN[2]	PWMA-PWMA[29]	ACIFB-ACAN[3]	EIC-EXTINT[0]	CAT-CSB[13]
31	PB07	39	VDDIO	Normal I/O	TC1-B1	USART3-RXD	ADCIFB-AD[7]	GLOC-IN[1]	PWMA-PWMA[30]	ACIFB-ACAP[3]	EIC-EXTINT[1]	CAT-CSA[13]
32	PB08	40	VDDIO	Normal I/O	TC1-A2	USART3-RTS	ADCIFB-AD[8]	GLOC-IN[0]	PWMA-PWMA[31]	CAT-SYNC	EIC-EXTINT[2]	CAT-CSB[12]
29	PB09	41	VDDIO	Normal I/O	TC1-B2	USART3-CTS	USART3-CLK		PWMA-PWMA[32]	ACIFB-ACBN[1]	EIC-EXTINT[3]	CAT-CSB[15]
23	PB10	42	VDDIN	Normal I/O	TC1-CLK0	USART1-TXD	USART3-CLK	GLOC-OUT[1]	PWMA-PWMA[33]		EIC-EXTINT[4]	CAT-CSB[16]
44	PB11	43	VDDIO	Normal I/O	TC1-CLK1	USART1-RXD		ADCIFB-TRIGGER	PWMA-PWMA[34]	CAT-VDIVEN	EIC-EXTINT[5]	CAT-CSA[16]
5	PB12	44	VDDIO	Normal I/O	TC1-CLK2		TWIMS1-TWALM	CAT-SYNC	PWMA-PWMA[35]	ACIFB-ACBP[3]	SCIF-GCLK[4]	CAT-CSA[15]

See Section 3.3 for a description of the various peripheral signals.

Signals are prioritized according to the function priority listed in Table 3-2 on page 10 if multiple functions are enabled simultaneously.

Refer to "Electrical Characteristics" on page 40 for a description of the electrical properties of the pad types used.

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.

Table 3-2. Peripheral Functions

Function	Description
A	GPIO peripheral selection A
B	GPIO peripheral selection B
C	GPIO peripheral selection C
D	GPIO peripheral selection D
E	GPIO peripheral selection E
F	GPIO peripheral selection F
G	GPIO peripheral selection G
H	GPIO peripheral selection H

3.2.3 JTAG Port Connections

If the JTAG is enabled, the JTAG will take control over a number of pins, irrespective of the I/O Controller configuration.

Table 3-3. JTAG Pinout

48TQFP/QFN pin	Pin name	JTAG pin
11	PA00	TCK
14	PA01	TMS
13	PA02	TDO
4	PA03	TDI

3.2.4 Nexus OCD AUX Port Connections

If the OCD trace system is enabled, the trace system will take control over a number of pins, respectively of the I/O Controller configuration. Two 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 3-4. Nexus OCD AUX Port Connections

Pin	AXS=1	AXS=0
EVTI_N	PA05	PB08
MDO[5]	PA10	PB00
MDO[4]	PA18	PB04
MDO[3]	PA17	PB05
MDO[2]	PA16	PB03
MDO[1]	PA15	PB02
MDO[0]	PA14	PB09

Table 3-4. Nexus OCD AUX Port Connections

Pin	AXS=1	AXS=0
EVTO_N	PA04	PA04
MCKO	PA06	PB01
MSEO[1]	PA07	PB11
MSEO[0]	PA11	PB12

3.2.5 Oscillator Pinout

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

Table 3-5. Oscillator Pinout

48TQFP/QFN/TLLGA	Pin	Oscillator Function
3	PA08	XIN0
46	PA10	XIN32
26	PA13	XIN32_2
2	PA09	XOUT0
47	PA12	XOUT32
25	PA20	XOUT32_2

3.2.6 Other Functions

The functions listed in Table 3-6 are not mapped to the normal GPIO functions.The aWire DATA pin will only be active after the aWire is enabled. The aWire DATAOUT pin will only be actice after the aWire is enabled and the full duplex command has been sent. The WAKE_N pin is always enabled. Please refer to Section 3.5.4 on page 20 for constraints on the WAKE_N pin.

Table 3-6. Other Functions

48TQFP/TQFN/TLLGA	Pin	Function
27	PA11	WAKE_N
22	RESET_N	aWire DATA
11	PA00	aWire DATAOUT

3.3 Signal Descriptions

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

Table 3-7. Signal Descriptions List

Signal Name	Function	Type	Active Level	Comments
Analog Comparator Interface - ACIFB				
ACAN3 - ACAN0	Negative inputs for comparators "A"	Analog		
ACAP3 - ACAP0	Positive inputs for comparators "A"	Analog		
ACBN3 - ACBN0	Negative inputs for comparators "B"	Analog		
ACBP3 - ACBP0	Positive inputs for comparators "B"	Analog		
ACREFN	Common negative reference	Analog		
ADC Interface - ADCIFB				
AD8 - AD0	Analog Signal	Analog		
ADP1 - ADP0	Drive Pin for touch screen	Output		
PRND	Pseudorandom output signal	Output		
TRIGGER	External trigger	Input		
aWire - AW				
DATA	aWire data	I/O		
DATAOUT	aWire data output for full duplex mode	I/O		
Capacitive Touch Module - CAT				
CSA16 - CSA0	Capacitive Sense A	I/O		
CSB16 - CSB0	Capacitive Sense B	I/O		
SMP	SMP signal	Output		
SYNC	Synchronize signal	Input		
VDIVEN	Voltage divider enable	Output		
External Interrupt Controller - EIC				
NMI	Non-Maskable Interrupt	Input		
EXTINT5 - EXTINT1	External interrupt	Input		
Glue Logic Controller - GLOC				
IN7 - IN0	Inputs to lookup tables	Input		
OUT1 - OUT0	Outputs from lookup tables	Output		
JTAG module - JTAG				
TCK	Test Clock	Input		
TDI	Test Data In	Input		
TDO	Test Data Out	Output		
TMS	Test Mode Select	Input		
Power Manager - PM				

Table 3-7. Signal Descriptions List

RESET_N	Reset	Input	Low	
Basic Pulse Width Modulation Controller - PWMA				
PWMA35 - PWMA0	PWMA channel waveforms	Output		
PWMAOD35 - PWMAOD0	PWMA channel waveforms, open drain mode	Output		Not all channels support open drain mode
System Control Interface - SCIF				
GCLK4 - GCLK0	Generic Clock Output	Output		
RC32OUT	RC32K output at startup	Output		
XIN0	Crystal 0 Input	Analog/ Digital		
XIN32	Crystal 32 Input (primary location)	Analog/ Digital		
XIN32_2	Crystal 32 Input (secondary location)	Analog/ Digital		
XOUT0	Crystal 0 Output	Analog		
XOUT32	Crystal 32 Output (primary location)	Analog		
XOUT32_2	Crystal 32 Output (secondary location)	Analog		
Serial Peripheral Interface - SPI				
MISO	Master In Slave Out	I/O		
MOSI	Master Out Slave In	I/O		
NPCS3 - NPCS0	SPI Peripheral Chip Select	I/O	Low	
SCK	Clock	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		
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 - TWIMS0, TWIMS1				
TWALM	SMBus SMBALERT	I/O	Low	
TWCK	Two-wire Serial Clock	I/O		
TWD	Two-wire Serial Data	I/O		
Universal Synchronous/Asynchronous Receiver/Transmitter - USART0, USART1, USART2, USART3				
CLK	Clock	I/O		

Table 3-7. Signal Descriptions List

CTS	Clear To Send	Input	Low	
RTS	Request To Send	Output	Low	
RXD	Receive Data	Input		
TXD	Transmit Data	Output		

Table 3-8. Signal Description List, continued

Signal Name	Function	Type	Active Level	Comments
Power				
VDDCORE	Core Power Supply / Voltage Regulator Output	Power Input/Output		1.62 V to 1.98 V
VDDIO	I/O Power Supply	Power Input		1.62 V to 3.6 V. VDDIO should always be equal to or lower than VDDIN.
VDDANA	Analog Power Supply	Power Input		1.62 V to 1.98 V
ADVREFP	Analog Reference Voltage	Power Input		TBD to 1.98 V
VDDIN	Voltage Regulator Input	Power Input		1.62 V to 3.6V (1)
GNDANA	Analog Ground	Ground		
GND	Ground	Ground		
Auxiliary Port - AUX				
MCKO	Trace Data Output Clock	Output		
MDO5 - MDO0	Trace Data Output	Output		
MSEO1 - MSEO0	Trace Frame Control	Output		
EVTI_N	Event In	Input	Low	
EVTO_N	Event Out	Output	Low	
General Purpose I/O pin - GPIOA, GPIOB				
PA22 - PA0	Parallel I/O Controller GPIOA	I/O		
PB12 - PB0	Parallel I/O Controller GPIOB	I/O		

1. See Section 3.5

3.4 I/O Line Considerations

3.4.1 JTAG Pins

The JTAG is enabled if TCK is low while the RESET_N pin is released. The TCK, TMS, and TDI pins have pull-up resistors when JTAG is enabled. TDO pin is an output, driven at VDDIO, and has no pull-up resistor. These JTAG pins can be used as GPIO pins and muxed with peripherals when the JTAG is disabled.

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 detector, the RESET_N pin can be left unconnected in case no reset from the system needs to be applied to the product.

The RESET_N pin is also used for the aWire debug protocol. When the pin is used for debugging, it must not be driven by the application.

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 GPIO pins.

3.4.4 GPIO Pins

All the I/O lines integrate a pull-up resistor. Programming of this pull-up resistor is performed independently for each I/O line through the GPIO Controllers. After reset, I/O lines default as inputs with pull-up resistors disabled, except PA00.

3.4.5 ADC Input Pins

These pins are regular I/O pins powered from the VDDIO. However, when these pins are used for ADC inputs, the voltage applied to the pin must not exceed 1.98V. Internal circuitry ensures that the pin cannot be used as an analog input pin when the I/O drives to VDD. When the pins are not used for ADC inputs, the pins may be driven to the full I/O voltage range.

3.5 Power Considerations

3.5.1 Power Supplies

The AT32UC3L has several types of power supply pins:

- VDDIO: Powers I/O lines. Voltage is 1.8 to 3.3V nominal.
- VDDIN: Powers I/O lines and the internal regulator. Voltage is 1.8 to 3.3V nominal.
- VDDANA: Powers the ADC. Voltage is 1.8V nominal.
- VDDCORE: Powers the core, memories, and peripherals. Voltage is 1.8V nominal.

The ground pins GND are common to VDDCORE and VDDIO. The ground pin for VDDANA is GNDANA.

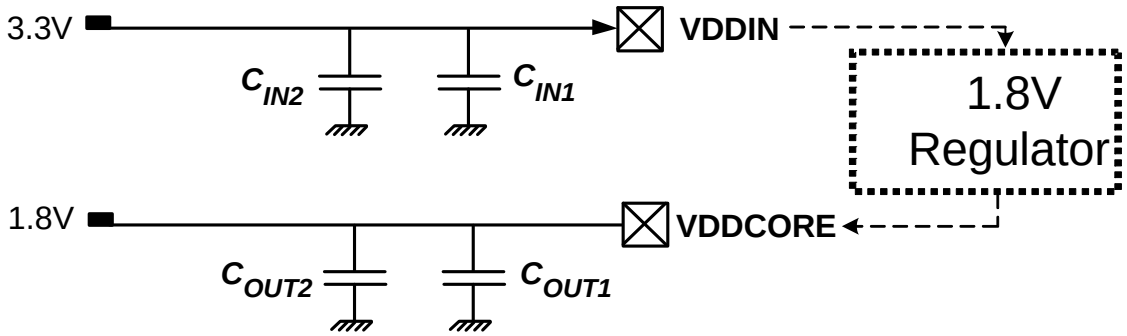
Refer to ”Electrical Characteristics” on page 40 for power consumption on the various supply pins.

3.5.2 Voltage Regulator

The AT32UC3L embeds a voltage regulator that converts from 3.3V nominal to 1.8V with a load of up to 60 mA. The regulator supplies the output voltage on VDDCORE. VDDCORE should be externally connected to the 1.8V domains. See Section 3.5.3 for regulator connection figures.

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 parallell between VDDCORE and GND as close to the chip as possible. Please refer to Section 7.9.1 for decoupling capacitors values and regulator characteristics.

Figure 3-2. Supply Decoupling



3.5.3 Regulator Connection

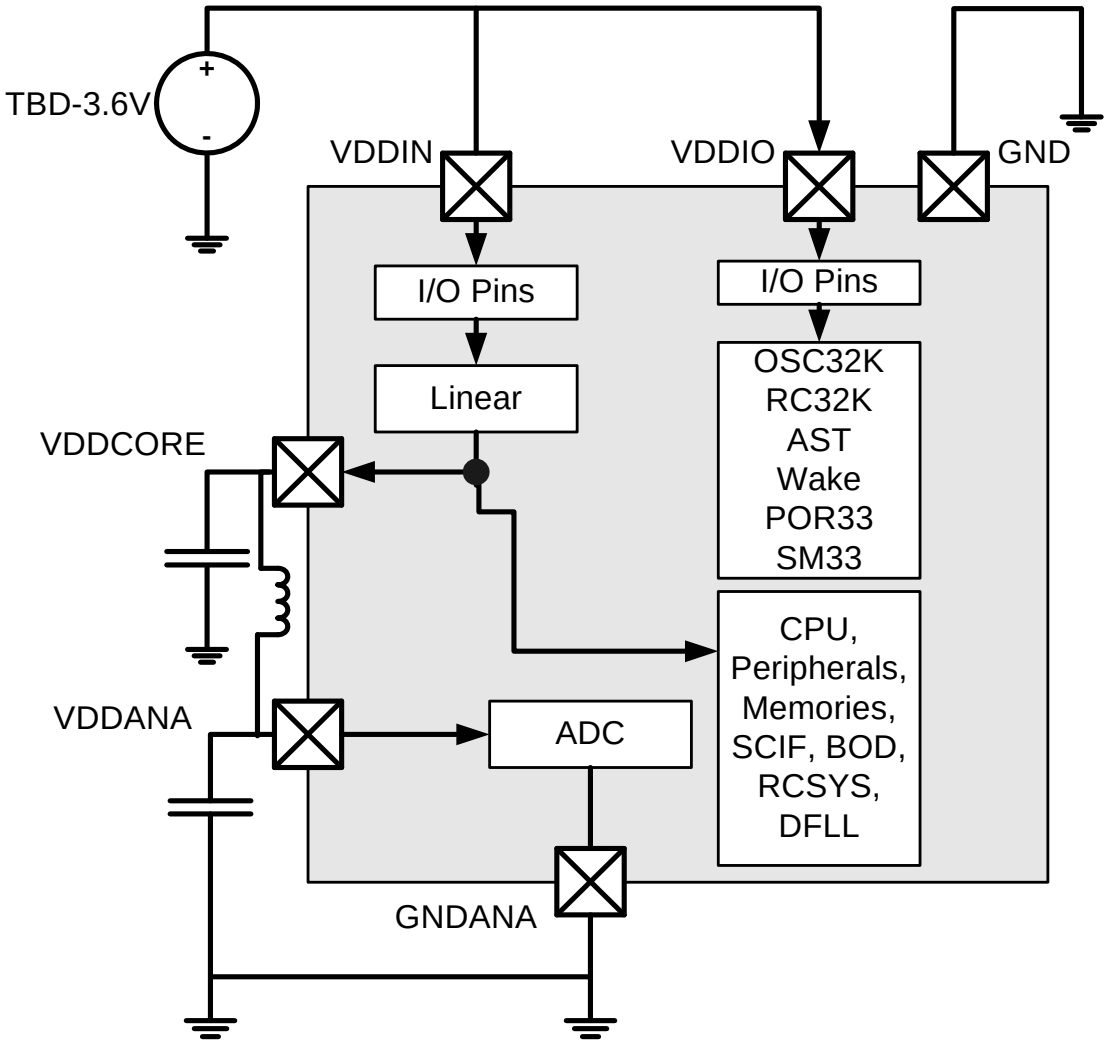
The AT32UC3L supports three power supply configurations:

- 3.3V single supply mode
- 1.8V single supply mode
- 3.3V supply mode, with 1.8V regulated I/O lines

3.5.3.1 3.3V Single Supply Mode

In 3.3V single supply mode the internal regulator is connected to the 3.3V source (VDDIN pin) and its output feeds VDDCORE. Figure 3-3 shows the power schematics to be used for 3.3V single supply mode. All I/O lines will be powered by the same power (VDDIN=VDDIO).

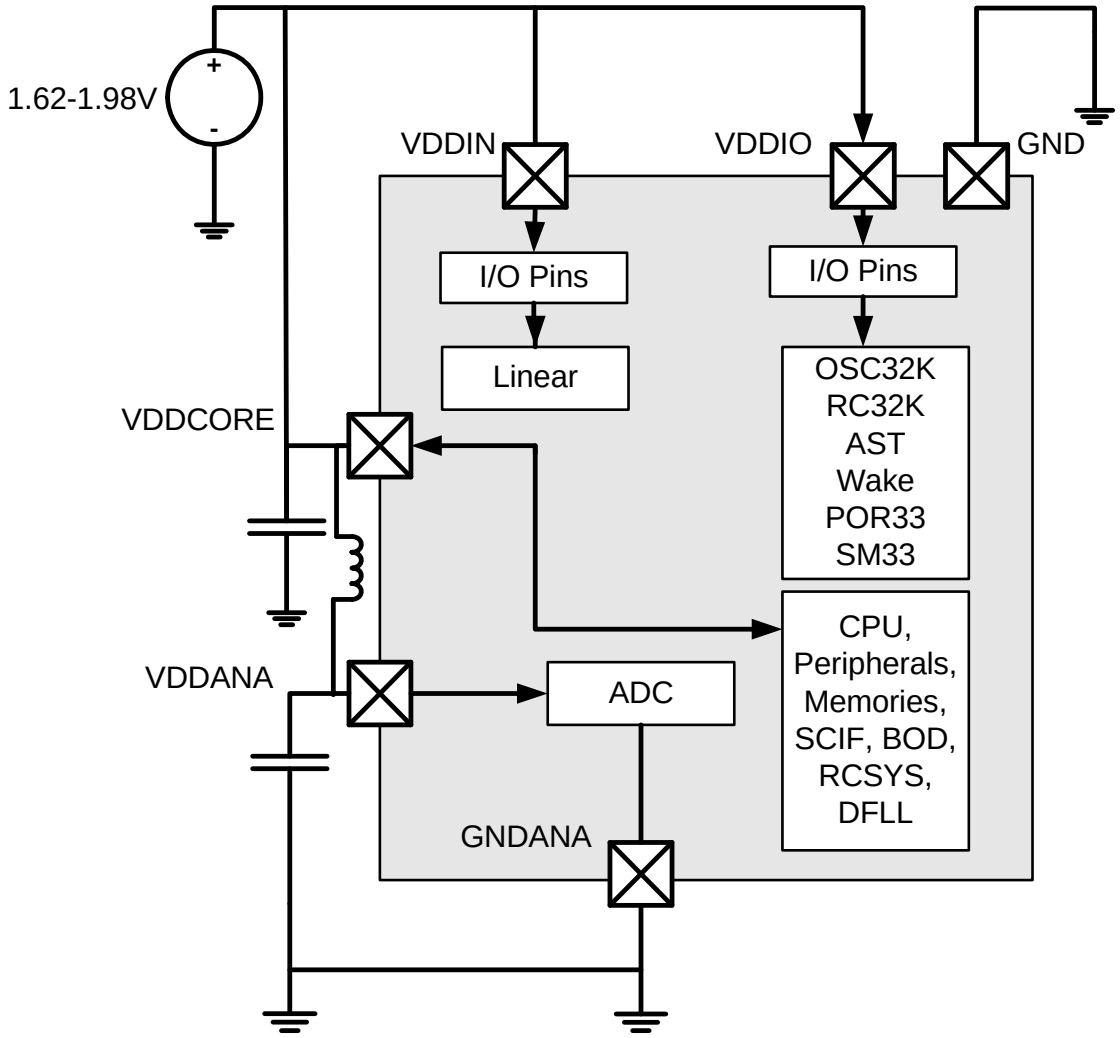
Figure 3-3. 3.3V Single Power Supply mode



3.5.3.2 *1.8V Single Supply Mode*

In 1.8V single supply mode the internal regulator is not used, and VDDIO and VDDCORE are powered by a single 1.8V supply as shown in Figure 3-4. All I/O lines will be powered by the same power (VDDIN = VDDIO = VDDCORE).

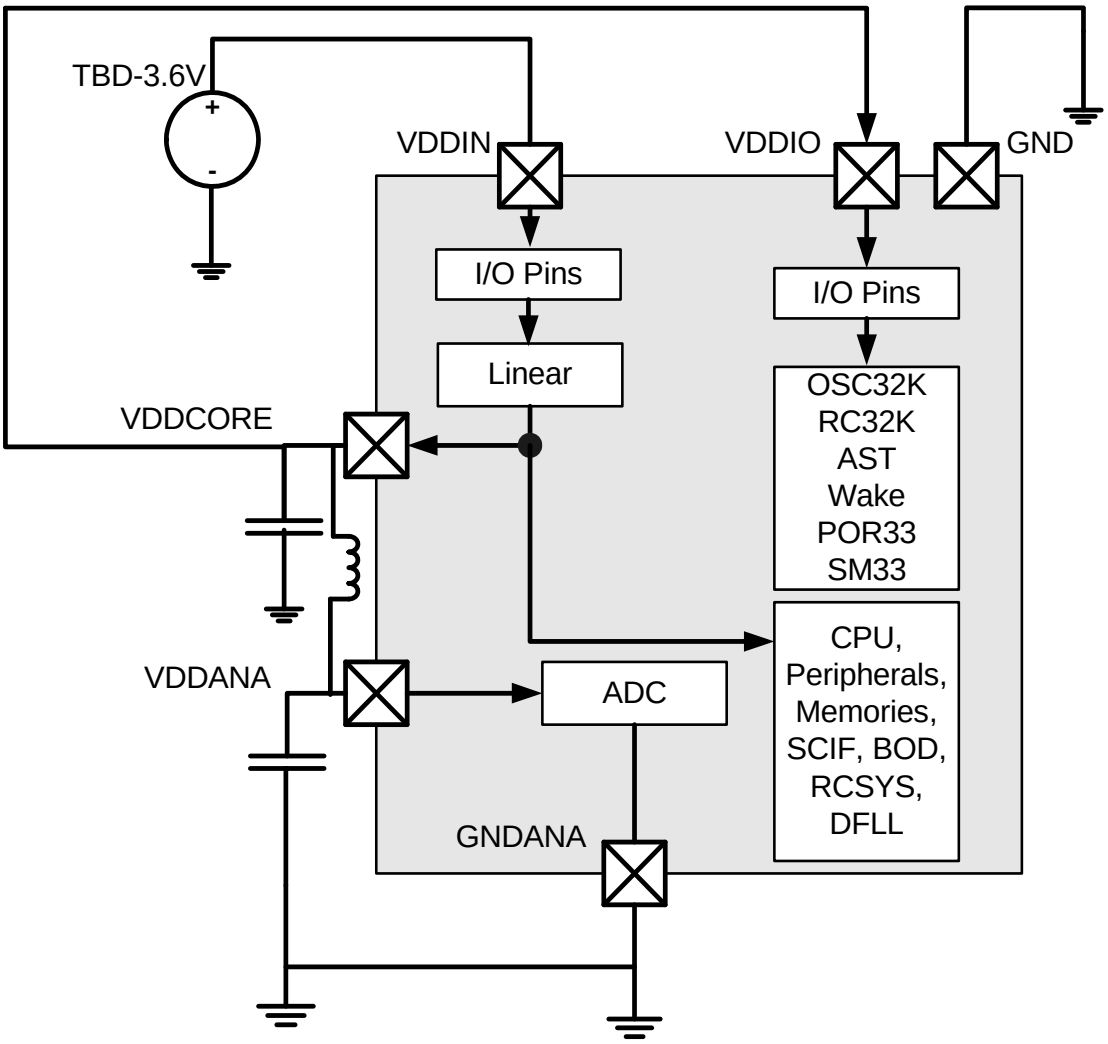
Figure 3-4. 1.8V Single Power Supply Mode.



3.5.3.3 *3.3V Supply Mode with 1.8V Regulated I/O Lines*

In this mode, the internal regulator is connected to the 3.3V source and its output is connected to both VDDCORE and VDDIO as shown in Figure 3-5. This configuration is required in order to use Shutdown mode.

Figure 3-5. 3.3V Power with 1.8V Regulated I/O Lines



In this mode, some I/O lines are powered by VDDIN while others I/O lines are powered by VDDIO. Refer to Table 3-1 on page 8 for description of power supply for each I/O line.

Important note: As the regulator has a maximum output current of 60mA, this mode can only be used in applications where the maximum I/O current is known and compatible with the core and peripheral power consumption. Typically, great care must be used to ensure that only a few I/O lines are toggling at the same time and drive very small loads.

3.5.4 Power-up Sequence

3.5.4.1 Maximum Rise Rate

To avoid risk of latch-up, the rise rate of the power supplies must not exceed the values described in Table 7-3 on page 41.

Recommended order for power supplies is also described in this table.

3.5.4.2 Minimum Rise Rate

The integrated Power-Reset circuitry monitoring the VDDIN powering supply requires a minimum rise rate for the VDDIN power supply.

See Table 7-3 on page 41 for the minimum rise rate value.

If the application can not ensure that the minimum rise rate condition for the VDDIN power supply is met, one of the following configuration can be used:

- A logic “0” value is applied during power-up on pin PA11 until VDDIN rises above 1.2V.
- A logic “0” value is applied during power-up on pin RESET_N until VDDIN rises above 1.2V.

4. Processor and Architecture

Rev: 2.1.0.0

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 extention with saturating arithmetic, and a wide variety of multiply instructions**
- **3-stage pipeline allowing 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**
- **Secure State for supporting FlashVault™ technology**

4.2 AVR32 Architecture

AVR32 is a new, 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 microarchitectures, 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

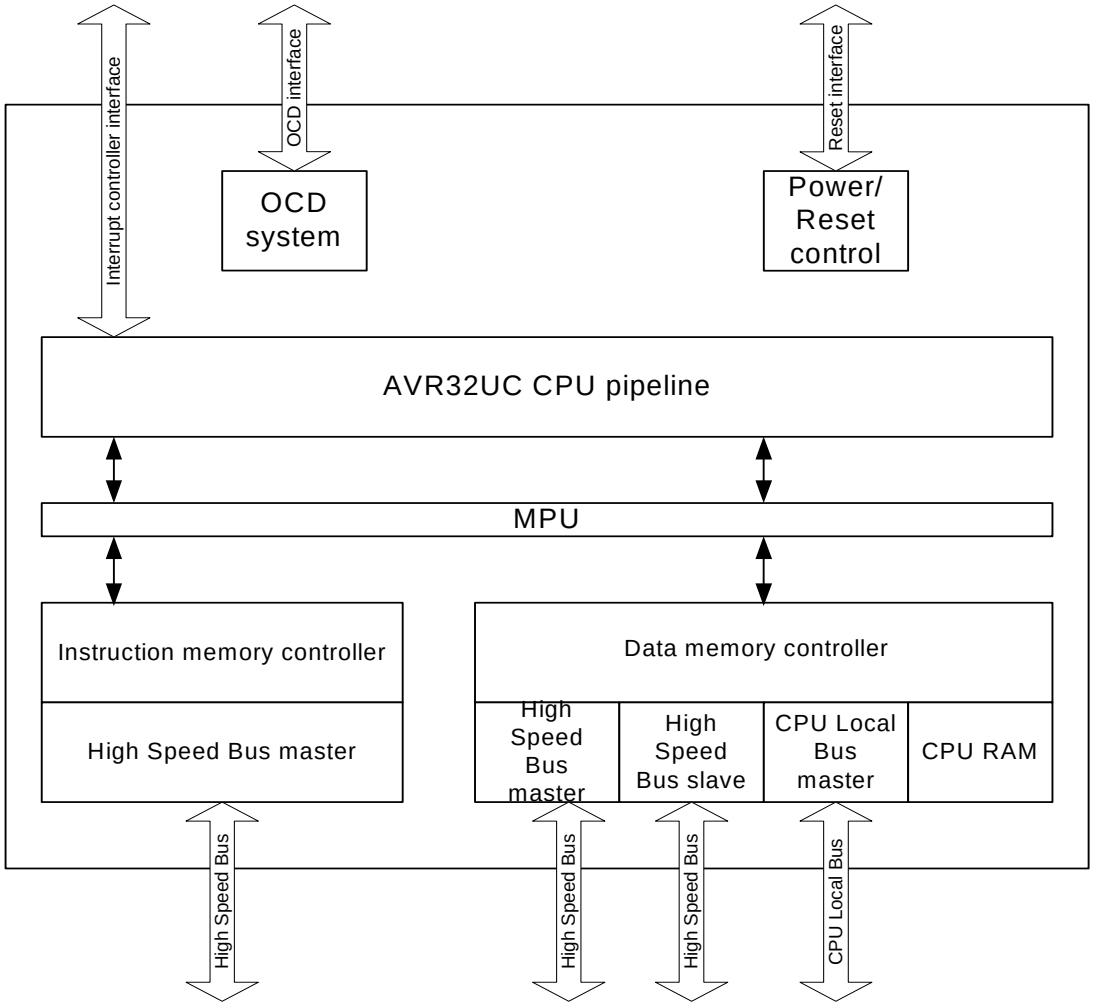
The AVR32UC CPU targets low- and medium-performance applications, and provides an advanced On-Chip Debug (OCD) system, no caches, and a Memory Protection Unit (MPU). Java acceleration hardware is not implemented.

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 I/O controller ports. This local bus has to be enabled by writing a one to 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 CPU Local Bus section in the Memories chapter.

Figure 4-1 on page 23 displays the contents of AVR32UC.

Figure 4-1. Overview of the AVR32UC CPU



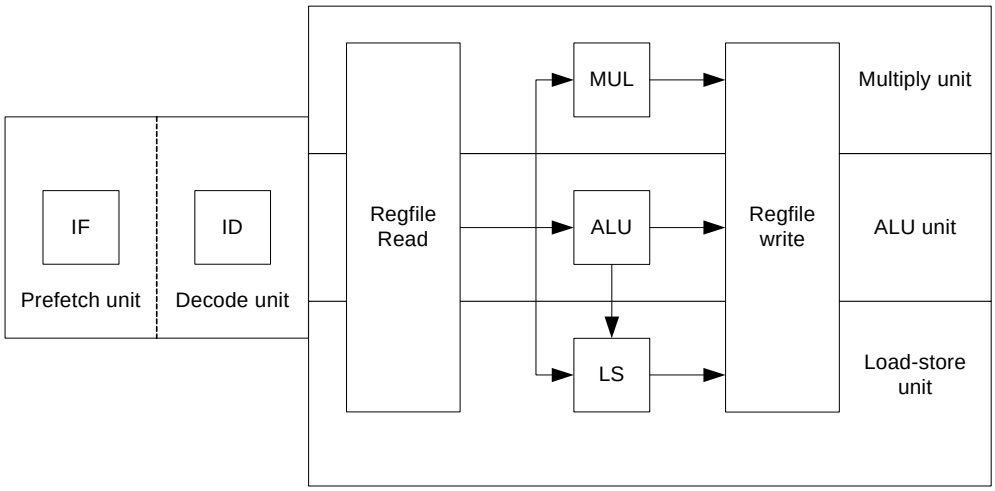
4.3.1 Pipeline Overview

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.

4.3.2.1 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.

4.3.2.2 Java Support

AVR32UC does not provide Java hardware acceleration.

4.3.2.3 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.

4.3.2.4 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

4.3.2.5 *Unimplemented Instructions*

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

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

4.3.2.6 *CPU and Architecture Revision*

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

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

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

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	Bit 31 Bit 0	Bit 31 Bit 0	Bit 31 Bit 0
PC	PC	PC	PC	PC	PC	PC	PC	PC
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_SEC
R12	R12	R12	R12	R12	R12	R12	R12	R12
R11	R11	R11	R11	R11	R11	R11	R11	R11
R10	R10	R10	R10	R10	R10	R10	R10	R10
R9	R9	R9	R9	R9	R9	R9	R9	R9
R8	R8	R8	R8	R8	R8	R8	R8	R8
R7	R7	R7	R7	R7	R7	R7	R7	R7
R6	R6	R6	R6	R6	R6	R6	R6	R6
R5	R5	R5	R5	R5	R5	R5	R5	R5
R4	R4	R4	R4	R4	R4	R4	R4	R4
R3	R3	R3	R3	R3	R3	R3	R3	R3
R2	R2	R2	R2	R2	R2	R2	R2	R2
R1	R1	R1	R1	R1	R1	R1	R1	R1
R0	R0	R0	R0	R0	R0	R0	R0	R0
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 and Figure 4-5. 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

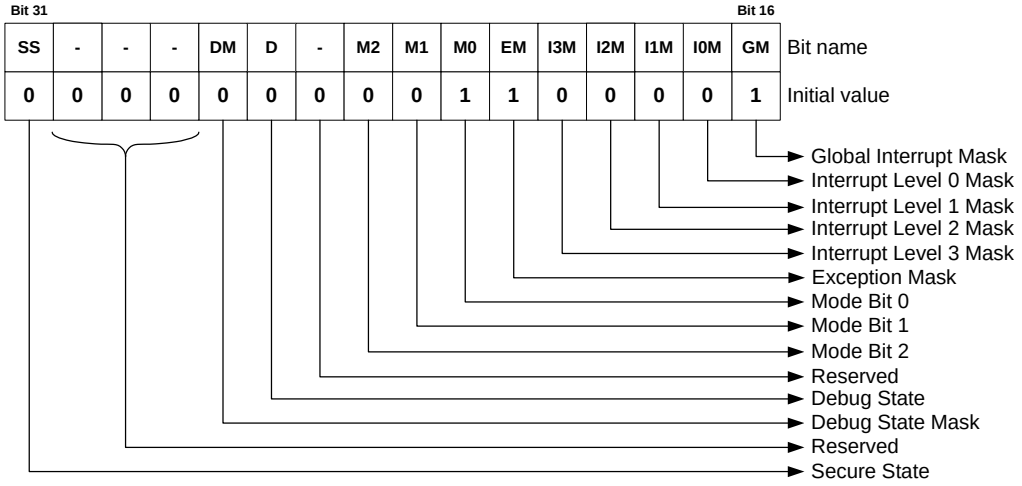
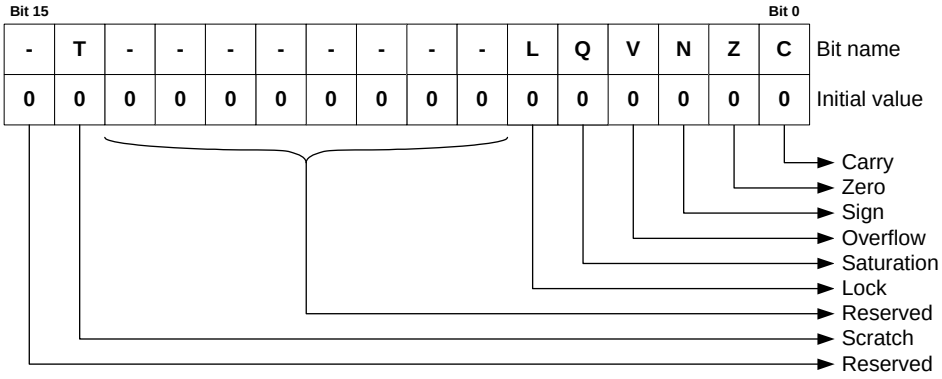


Figure 4-5. The Status Register Low Halfword



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.

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.

4.4.3.3 Secure State

The AVR32 can be set in a secure state, that allows a part of the code to execute in a state with higher security levels. The rest of the code can not access resources reserved for this secure code. Secure State is used to implement FlashVault technology. Refer to the *AVR32UC Technical Reference Manual* for details.

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	JECR	Unused in AVR32UC
22	88	JOSP	Unused in AVR32UC
23	92	JAVA_LV0	Unused in AVR32UC

Table 4-3. System Registers (Continued)

Reg #	Address	Name	Function
24	96	JAVA_LV1	Unused in AVR32UC
25	100	JAVA_LV2	Unused in AVR32UC
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

Table 4-3. System Registers (Continued)

Reg #	Address	Name	Function
90	360	MPUPSR2	MPU Privilege Select Register region 2
91	364	MPUPSR3	MPU Privilege Select Register region 3
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	412	SS_STATUS	Secure State Status Register
104	416	SS_ADRF	Secure State Address Flash Register
105	420	SS_ADRR	Secure State Address RAM Register
106	424	SS_ADR0	Secure State Address 0 Register
107	428	SS_ADR1	Secure State Address 1 Register
108	432	SS_SP_SYS	Secure State Stack Pointer System Register
109	436	SS_SP_APP	Secure State Stack Pointer Application Register
110	440	SS_RAR	Secure State Return Address Register
111	444	SS_RSR	Secure State Return Status Register
112-191	448-764	Reserved	Reserved for future use
192-255	768-1020	IMPL	IMPLEMENTATION DEFINED

4.5 Exceptions and Interrupts

In the AVR32 architecture, events are used as a common term for exceptions and interrupts. AVR32UC incorporates a powerful event handling scheme. The different event sources, like Illegal Op-code and interrupt requests, have different priority levels, ensuring a well-defined behavior when multiple events are received simultaneously. Additionally, pending events of a higher priority class may preempt handling of ongoing events of a lower priority class.

When an event occurs, the execution of the instruction stream is halted, and execution is passed to an event handler at an address specified in Table 4-4 on page 34. 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 interrupt sources have autovectored 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 interrupt requests, yielding a uniform event handling scheme.

An interrupt controller does the priority handling of the 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 IRAM 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 responsability 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 on page 34, 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 0x80000000. 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 interrupt requests have entry points located at an offset relative to EVBA. This autovector offset is specified by an 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 on page 34. 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 on page 34. 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	0x80000000	Reset	External input	Undefined
2	Provided by OCD system	OCD Stop CPU	OCD system	First non-completed instruction
3	EVBA+0x00	Unrecoverable exception	Internal	PC of offending instruction
4	EVBA+0x04	TLB multiple hit	MPU	PC of offending instruction
5	EVBA+0x08	Bus error data fetch	Data bus	First non-completed instruction
6	EVBA+0x0C	Bus error instruction fetch	Data bus	First non-completed instruction
7	EVBA+0x10	NMI	External input	First non-completed instruction
8	Autovectored	Interrupt 3 request	External input	First non-completed instruction
9	Autovectored	Interrupt 2 request	External input	First non-completed instruction
10	Autovectored	Interrupt 1 request	External input	First non-completed instruction
11	Autovectored	Interrupt 0 request	External input	First non-completed instruction
12	EVBA+0x14	Instruction Address	CPU	PC of offending instruction
13	EVBA+0x50	ITLB Miss	MPU	PC of offending instruction
14	EVBA+0x18	ITLB Protection	MPU	PC of offending instruction
15	EVBA+0x1C	Breakpoint	OCD system	First non-completed instruction
16	EVBA+0x20	Illegal Opcode	Instruction	PC of offending instruction
17	EVBA+0x24	Unimplemented instruction	Instruction	PC of offending instruction
18	EVBA+0x28	Privilege violation	Instruction	PC of offending instruction
19	EVBA+0x2C	Floating-point	UNUSED	
20	EVBA+0x30	Coprocessor absent	Instruction	PC of offending instruction
21	EVBA+0x100	Supervisor call	Instruction	PC(Supervisor Call) +2
22	EVBA+0x34	Data Address (Read)	CPU	PC of offending instruction
23	EVBA+0x38	Data Address (Write)	CPU	PC of offending instruction
24	EVBA+0x60	DTLB Miss (Read)	MPU	PC of offending instruction
25	EVBA+0x70	DTLB Miss (Write)	MPU	PC of offending instruction
26	EVBA+0x3C	DTLB Protection (Read)	MPU	PC of offending instruction
27	EVBA+0x40	DTLB Protection (Write)	MPU	PC of offending instruction
28	EVBA+0x44	DTLB Modified	UNUSED	

5. Memories

5.1 Embedded Memories

- **Internal High-Speed Flash**
  - **64Kbytes (AT32UC3L064)**
  - **32Kbytes (AT32UC3L032)**
  - **16Kbytes (AT32UC3L016)**
    - **0 Wait State Access at up to 25 MHz in Worst Case Conditions**
    - **1 Wait State Access at up to 50 MHz 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 8% compared to 0 wait state operation**
    - **100 000 Write Cycles, 15-year Data Retention Capability**
    - **4ms Page Programming Time, 8 ms Chip Erase Time**
    - **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, Single-cycle access at full speed**
  - **16Kbytes (AT32UC3L064, AT32UC3L032)**
  - **8Kbytes (AT32UC3L016)**

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 AVR32 Architecture Manual. The 32-bit physical address space is mapped as follows:

Table 5-1. AT32UC3L Physical Memory Map

Device	Start Address	Size		
		AT32UC3L064	AT32UC3L032	AT32UC3L016
Embedded SRAM	0x00000000	16 Kbytes	16 Kbytes	8 Kbytes
Embedded Flash	0x80000000	64 Kbytes	32 Kbytes	16 Kbytes
HSB-PB Bridge B	0xFFFE0000	64 Kbytes	64 kBytes	64 Kbytes
HSB-PB Bridge A	0xFFFF0000	64 Kbytes	64 Kbytes	64 Kbytes

Table 5-2. Flash Memory Parameters

Part Number	Flash Size (*FLASH_PW*)	Number of pages (*FLASH_P*)	Page size (*FLASH_W*)
AT32UC3L064	64 Kbytes	256	64 words
AT32UC3L032	32 Kbytes	128	64 words
AT32UC3L016	16 Kbytes	64	64 words

5.3 Peripheral Address Map

Table 5-3. Peripheral Address Mapping

Address		Peripheral Name	Bus
0xFFFE0000	FLASHCDW	Flash Controller - FLASHCDW	
0xFFFE0400	HMATRIX	HSB Matrix - HMATRIX	
0xFFFE0800	SAU	Secure Access Unit - SAU	
0xFFFF0000	PDCA	Peripheral DMA Controller - PDCA	
0xFFFF1000	INTC	Interrupt controller - INTC	
0xFFFF1400	PM	Power Manager - PM	
0xFFFF1800	SCIF	System Control Interface - SCIF	
0xFFFF1C00	AST	Asynchronous Timer - AST	
0xFFFF2000	WDT	Watchdog Timer - WDT	
0xFFFF2400	EIC	External Interrupt Controller - EIC	
0xFFFF2800	FREQM	Frequency Meter - FREQM	
0xFFFF2C00	GPIO	General Purpose Input/Output Controller - GPIO	
0xFFFF3000	USART0	Universal Synchronous/Asynchronous Receiver/Transmitter - USART0	
0xFFFF3400	USART1	Universal Synchronous/Asynchronous Receiver/Transmitter - USART1	
0xFFFF3800	USART2	Universal Synchronous/Asynchronous Receiver/Transmitter - USART2	
0xFFFF3C00	USART3	Universal Synchronous/Asynchronous Receiver/Transmitter - USART3	
0xFFFF4000	SPI	Serial Peripheral Interface - SPI	
0xFFFF4400	TWIM0	Two-wire Master Interface - TWIM0	

Table 5-3. Peripheral Address Mapping

0xFFFF4800	TWIM1	Two-wire Master Interface - TWIM1
0xFFFF4C00	TWIS0	Two-wire Slave Interface - TWIS0
0xFFFF5000	TWIS1	Two-wire Slave Interface - TWIS1
0xFFFF5400	PWMA	Basic Pulse Width Modulation Controller - PWMA
0xFFFF5800	TC0	Timer/Counter - TC0
0xFFFF5C00	TC1	Timer/Counter - TC1
0xFFFF6000	ADCIFB	ADC Interface - ADCIFB
0xFFFF6400	ACIFB	Analog Comparator Interface - ACIFB
0xFFFF6800	CAT	Capacitive Touch Module - CAT
0xFFFF6C00	GLOC	Glue Logic Controller - GLOC
0xFFFF7000	AW	aWire - AW

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 5-4. Local Bus Mapped GPIO Registers

Port	Register	Mode	Local Bus Address	Access
A	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
B	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

6. Boot Sequence

This chapter summarizes the boot sequence of the AT32UC3L. The behavior after power-up is controlled by the Power Manager. For specific details, refer to the Power Manager chapter.

6.1 Starting of Clocks

After power-up, the device will be held in a reset state by the Power-On Reset circuitry for a short time to allow the power to stabilize throughout the device. After reset, the device will use the System RC Oscillator (RCSYS) as clock source. Please refer to Table 7-20 on page 48 for the frequency for this oscillotor.

On system start-up, the DFLL is disabled. All clocks to all modules are running. No clocks have a divided frequency; all parts of the system receive a clock with the same frequency as the System 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 0x80000000. This address points to the first address in the internal Flash.

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

6.3 RC32K Clock Output at Startup

After power-up, the clock generated by the 32kHz RC oscillator (RC32K) will be output on I/O line PA20, even when the device is still reset by the Power-On Reset Circuitry.

This clock can be used by the system to start other devices or to clock a switching regulator to rise the power supply voltage up to an acceptable value.

The clock will be available on I/O line PA20 until one of the following conditions are true:

- PA20 is configured to use a GPIO function other than F (SCIF-RC32OUT)
- PA20 is configured as a General Purpose Input/Output (GPIO)
- The bit FRC32 in the Power Manager PPCR register is cleared (see Power Manager chapter)

The maximum amplitude of the clock signal will be defined by VDDIN.

7. Electrical Characteristics

7.1 Disclaimer

All values in this chapter are preliminary and subject to change without further notice.

7.2 Absolute Maximum Ratings*

Table 7-1. Absolute Maximum Ratings

Operating temperature..... -40°C to +85°C
Storage temperature..... -60°C to +150°C
Voltage on all pins (except those noted below) .....-0.3V to V_{VDDIO}+0.3V
Voltage on PA11, PA13, PA 20.....-0.3V to V_{VDDIN}+0.3V
Voltage on 5V tolerant pins with respect to ground .... -0.3V to 5.5V
DC current per I/O pin..... TBD mA
DC current V_{CC} and GND pins..... TBD mA
Maximum operating voltage (VDDCORE) 1.98V
Maximum operating voltage (VDDIO, VDDIN)..... 3.6V

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

7.3 Supply Characteristics

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

Table 7-2. Supply Characteristics

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

Table 7-3. Supply Rise Rates and Order

Symbol	Parameter	Rise Rate			
		Min	Max	Unit	Comment
V_{VDDIO}	DC supply peripheral I/Os	0	2.5	V/µs	
V_{VDDIN}	DC supply peripheral I/Os and internal regulator	0.002(1)	2.5	V/µs	
$V_{VDDCORE}$	DC supply core	0	2.5	V/µs	Rise before or at the same time as VDDIO
V_{VDDANA}	Analog supply voltage	0	2.5	V/µs	Rise together with VDDCORE

Note: 1. Slower rise time requires external power-on circuit.

7.4 Clock Characteristics

These parameters are given in the following conditions:

$V_{VDDCORE}$ = 1.62 to 1.98V

Temperature = -40°C to 85°C

Table 7-4. Clock Frequencies

Symbol	Parameter	Conditions	Min	Max	Units
f_{CPU}	CPU clock frequency			50	MHz
f_{PBA}	PBA clock frequency			50	MHz
f_{PBB}	PBB clock frequency			50	MHz

7.5 Power Consumption

The values in Table 7-5 are measured values of power consumption with operating conditions as follows:

- V_{DDIO} = 1.8V
- V_{DDCORE} =1.8V
- T_A = 25°C
- I/Os are inactive with internal pull-up

Figure 7-1. Measurement Schematic

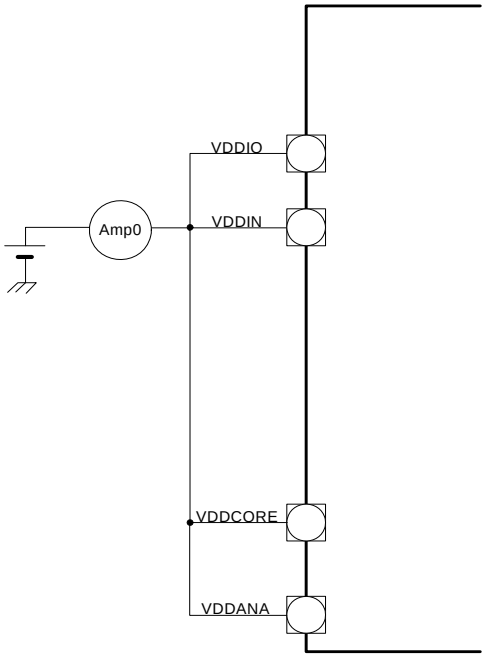


Table 7-5. Power Consumption for Different Modes

Mode	Conditions	Measured on	Consumption Typ	Unit
Active	Active mode(1)	Amp0	300	µA/MHz
Idle	Idle (2)	Amp0	150	µA/MHz
Frozen	Frozen sleep mode(2)	Amp0	90	µA/MHz
Standby	Standby sleep mode(3)	Amp0	70	µA/MHz
Stop	Stop sleep mode(4)	Amp0	30	µA
DeepStop	DeepStop sleep mode(4)	Amp0	20	µA
Static	Static sleep mode with RTC(4)	Amp0	7	µA
Static	Static sleep mode(5)	Amp0	5	µA
Shutdown	Shutdown sleep mode with RTC(6)	Amp0	1.5	µA
Shutdown	Shutdown sleep mode (7)	Amp0	0.1	µA

Note:
1. CPU performing recursive Fibonacci algorithm running from flash. Main clock source is DFLL. XIN0 stopped. XIN32: External clock. DFLL running. No peripheral clocks masked, peripharal clocks divided by 8. GPIOs on internal pull-up.
2. Main clock source is DFLL. XIN0 stopped. XIN32: External clock. DFLL running. No peripheral clocks masked. GPIOs on internal pull-up.
3. Main clock source is DFLL. XIN0 stopped. XIN32: External clock. DFLL running. GPIOs on internal pull-up.
4. XIN0 stopped. XIN32: External clock. DFLL stopped. GPIOs on internal pull-up.
5. XIN0 stopped. XIN32 stopped. DFLL stopped. GPIOs on internal pull-up.
6. XIN0 stopped. XIN32: External clock. DFLL stopped. GPIOs on internal pull-up.
7. XIN0 stopped. XIN32 stopped. GPIOs on internal pull-up.

Table 7-6. Power Consumption by Peripheral in Active Mode

Peripheral	Consumption Typ	Unit
ACIFB	TBD	µA/MHz
ADCIFB	TBD	
AST	TBD	
AW	TBD	
CAT	TBD	
EIC	TBD	
FLASHCDW	TBD	
FREQM	TBD	
GPIO	TBD	
HMATRIX	TBD	
INTC	TBD	
PDCA	TBD	
PM	TBD	
PWMA	TBD	
SAU	TBD	
SCIF	TBD	
SPI	TBD	
TC	TBD	
TWIM	TBD	
TWIS	TBD	
USART	TBD	
WDT	TBD	

7.6 I/O Pad Characteristics

Table 7-7. Normal I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance			105k		Ohm
V_{IL}	Input low-level voltage		-0.3		+0.8	V
V_{IH}	Input high-level voltage		TBD		V_{VDDIO}+0.3	V
V_{OL}	Output low-level voltage				0.4	V
V_{OH}	Output high-level voltage		V_{VDDIO}-0.4			V
I_{OL}	Output low-level current				2	mA
I_{OH}	Output high-level current				2	mA
I_{LEAK}	Input leakage current	Pull-up resistors disabled			1	µA
C_{IN}	Input capacitance			3(1)		pF

Note: 1. IBIS simulated values

Table 7-8. High-drive I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up resistance			105k		Ohm
V_{IL}	Input low-level voltage		-0.3		+0.8	V
V_{IH}	Input high-level voltage		TBD		V_{VDDIO}+0.3	V
V_{OL}	Output low-level voltage				0.4	V
V_{OH}	Output high-level voltage		V_{VDDIO}-0.4			V
I_{OL}	Output low-level current				4	mA
I_{OH}	Output high-level current				4	mA
I_{LEAK}	Input leakage current	Pull-up resistors disabled			1	µA
C_{IN}	Input capacitance			5(1)		pF

Note: 1. IBIS simulated values

Table 7-9. 5V Tolerant I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up Resistance			TBD		Ohm
V_{IL}	Input Low-level Voltage		-0.3		+0.8	V
V_{IH}	Input High-level Voltage		TBD		5.5V	V
V_{OL}	Output Low-level Voltage				0.4	V
V_{OH}	Output High-level Voltage		V_{VDDIO}-0.4			V
I_{OL}	Output Low-level Current				TBD	mA

Table 7-9. 5V Tolerant I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
I_{OH}	Output High-level Current				TBD	mA
I_{LEAK}	Input Leakage Current	Pull-up resistors disabled			TBD	µA
C_{IN}	Input Capacitance			TBD		pF

Table 7-10. TWI Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up Resistance			TBD		Ohm
V_{IL}	Input Low-level Voltage		-0.3		+0.8	V
V_{IH}	Input High-level Voltage		TBD		5.5V	V
V_{OL}	Output Low-level Voltage				0.4	V
V_{OH}	Output High-level Voltage		V_{VDDIO}-0.4			V
I_{OL}	Output Low-level Current				TBD	mA
I_{OH}	Output High-level Current				TBD	mA
I_{LEAK}	Input Leakage Current	Pull-up resistors disabled			TBD	µA
C_{IN}	Input Capacitance			TBD		pF
	Slew Rate			TBD		V/µs

Table 7-11. SMBus Compliant Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up Resistance			TBD		Ohm
V_{IL}	Input Low-level Voltage		-0.3		+0.8	V
V_{IH}	Input High-level Voltage		TBD		5.5V	V
V_{OL}	Output Low-level Voltage				0.4	V
	Input Voltage Range					
V_{OH}	Output High-level Voltage		V_{VDDIO}-0.4			V
I_{OL}	Output Low-level Current				TBD	mA
I_{OH}	Output High-level Current				TBD	mA
I_{LEAK}	Input Leakage Current	Pull-up resistors disabled			TBD	µA
C_{IN}	Input Capacitance			TBD		pF
	Slew Rate			TBD		V/µs

Table 7-12. Oscillator I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
R_{PULLUP}	Pull-up Resistance			30k		Ohm
V_{IL}	Input Low-level Voltage		-0.3		+0.8	V
V_{IH}	Input High-level Voltage		TBD		5.5V	V

Table 7-12. Oscillator I/O Pad Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{OL}	Output Low-level Voltage				0.4	V
V_{OH}	Output High-level Voltage		V_{VDDIO}-0.4			V
I_{OL}	Output Low-level Current				TBD	mA
I_{OH}	Output High-level Current				TBD	mA
I_{LEAK}	Input Leakage Current	Pull-up resistors disabled			TBD	µA
C_{IN}	Input Capacitance			TBD		pF

7.7 Oscillator Characteristics

7.7.1 Oscillator 0 Characteristics

7.7.1.1 Digital Clock Characteristics

The following table describes the characteristics for the oscillator when a digital clock is applied on XIN.

Table 7-13. Digital Clock Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
f_{CPXIN}	XIN clock frequency				50	MHz
t_{CHXIN}	XIN clock duty cycle		40		60	%
C_{IN}	XIN input capacitance			TBD		pF
R_{IN}	Optional pull-down resistor			TBD		kΩ

7.7.1.2 Crystal Oscillator Characteristics

The following table describes the characteristics for the oscillator when a crystal is connected between XIN and XOUT.

Table 7-14. Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$1/(t_{CPMAIN})$	Crystal oscillator frequency		3		16	MHz
C_{L1}, C_{L2}	Internal load capacitance ($C_{L1} = C_{L2}$)			TBD		pF
C_L	Equivalent load capacitance			TBD		pF
t_{ST}	Startup time			TBD		ms
I_{OSC}	Current consumption	Active mode @3MHz. Gain = G0		TBD		µA
		Active mode @16MHz. Gain = G3				

7.7.2 32 KHz Crystal Oscillator Characteristics

Table 7-16. 32 KHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$1/(t_{CP32KHz})$	Crystal oscillator frequency			32 768		Hz
t_{ST}	Startup time	R_S = TBD kΩ, C_L = TBD pF(1)		TBD		ms
C_L	Equivalent load capacitance		TBD		TBD	pF
I_{OSC}	Current consumption	Active mode		1.5		µA

Note: 1. R_S is the equivalent series resistance, C_L is the equivalent load capacitance.

7.7.3 DFLL Characteristics

Table 7-17. Digital Frequency Locked Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency		20		150	MHz
f_{IN}	Input frequency		0.02		16	MHz
I_{DFLL}	Current consumption	Active mode		TBD		µA/MHz
$t_{STARTUP}$	Startup time			TBD		cycles
t_{LOCK}	Lock time	f_{IN} = 32KHz, f_{OUT} = 50MHz		TBD		ms

7.7.4 RC120M Characteristics

Table 7-18. Internal 120MHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency			120		MHz
I_{RC120M}	Current consumption	Active mode		TBD		µA
$t_{STARTUP}$	Startup time			TBD		cycles

7.7.5 RC32K

Table 7-19. 32kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency		20	32	44	kHz
I_{RC32K}	Current consumption	Active mode		TBD		µA
$t_{STARTUP}$	Startup time			TBD		cycles

7.7.6 RCSYS

Table 7-20. System RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency			115		kHz
I_{RCSYS}	Current consumption	Active mode		TBD		µA
$t_{STARTUP}$	Startup time			TBD		cycles
	Tuning resolution			TBD		%

7.8 Flash Characteristics

Table 7-21 gives the device maximum operating frequency depending on the number of flash wait states and the flash read mode. The FSW bit in the FLASHCDW FSR register controls the number of wait states used when accessing the flash memory.

Table 7-21. Maximum Operating Frequency

Flash Wait States	Read Mode	Maximum Operating Frequency
1	High speed read mode	50MHz
0		25MHz
1	Normal read mode	30MHz
0		15MHz

7.9 Analog Characteristics

7.9.1 Regulator Characteristics

7.9.1.1 Electrical Characteristics

Table 7-22. Electrical Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
I_{OUT}	Maximum DC output current with $V_{VDDIN = 3.3V}$				TBD	mA
	Maximum DC output current with $V_{VDDIN = 1.8V}$				TBD	mA
I_{SCR}	Static current of internal regulator	Normal mode		TBD		µA
		Low Power mode		TBD		µA

7.9.1.2 Decoupling Requirements

Table 7-23. Decoupling Requirements

Symbol	Parameter	Condition	Typ	Techno.	Units
C_{IN1}	Input regulator capacitor 1		TBD		nF
C_{IN2}	Input regulator capacitor 2		10		µF
C_{OUT1}	Output regulator capacitor 1		100		nF
C_{OUT2}	Output regulator capacitor 2		2.2	Tantalum 0.5<ESR<10	µF

7.9.2 POR

Table 7-24. Power-on Reset Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{POT}	POR threshold voltage (rising)			1.5		V
	POR threshold voltage (falling)			1.3		V

7.9.3 SM33

Table 7-25. SM33 Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{TH}	Voltage threshold	onsm = '1', without calibration		1.8		V

7.9.4 POR33

Table 7-26. POR33 Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{TH}	Threshold voltage rising			1.5		V

7.10 Timing Characteristics

7.10.1 RESET_N Characteristics

Table 7-27. RESET_N Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Units
t_{RESET}	RESET_N minimum pulse length		10		ns

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	TQFP48	TBD	°C/W
θ_{JC}	Junction-to-case thermal resistance		TQFP48	TBD	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	QFN48	TBD	°C/W
θ_{JC}	Junction-to-case thermal resistance		QFN48	TBD	
θ_{JA}	Junction-to-ambient thermal resistance	Still Air	TLLGA48	TBD	°C/W
θ_{JC}	Junction-to-case thermal resistance		TLLGA48	TBD	

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

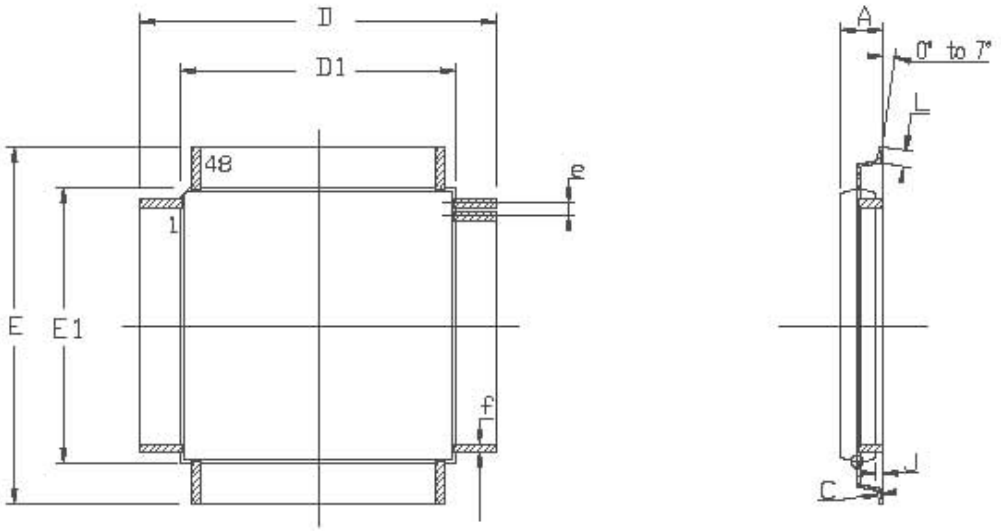
where:

- θ_{JA} = package thermal resistance, Junction-to-ambient (°C/W), provided in Table 8-1.
- θ_{JC} = package thermal resistance, Junction-to-case thermal resistance (°C/W), provided in Table 8-1.
- $\theta_{HEAT\ SINK}$ = cooling device thermal resistance (°C/W), provided in the device datasheet.
- P_D = device power consumption (W) estimated from data provided in the Section 7.5 on page 41.
- 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. TQFP-48 Package Drawing



COMMON DIMENSIONS IN MM

SYMBOL	Min	Max	NOTES
A	----	1.20	
A1	0.95	1.05	
C	0.09	0.20	
D	9.00 BSC		
D1	7.00 BSC		
E	9.00 BSC		
E1	7.00 BSC		
J	0.05	0.15	
L	0.45	0.75	
e	0.50 BSC		
f	0.17	0.27	

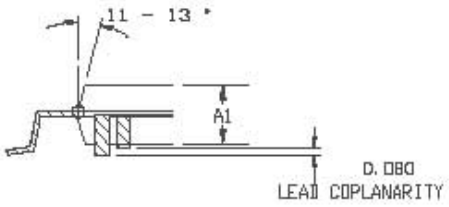


Table 8-2. Device and Package Maximum Weight

TBD	mg

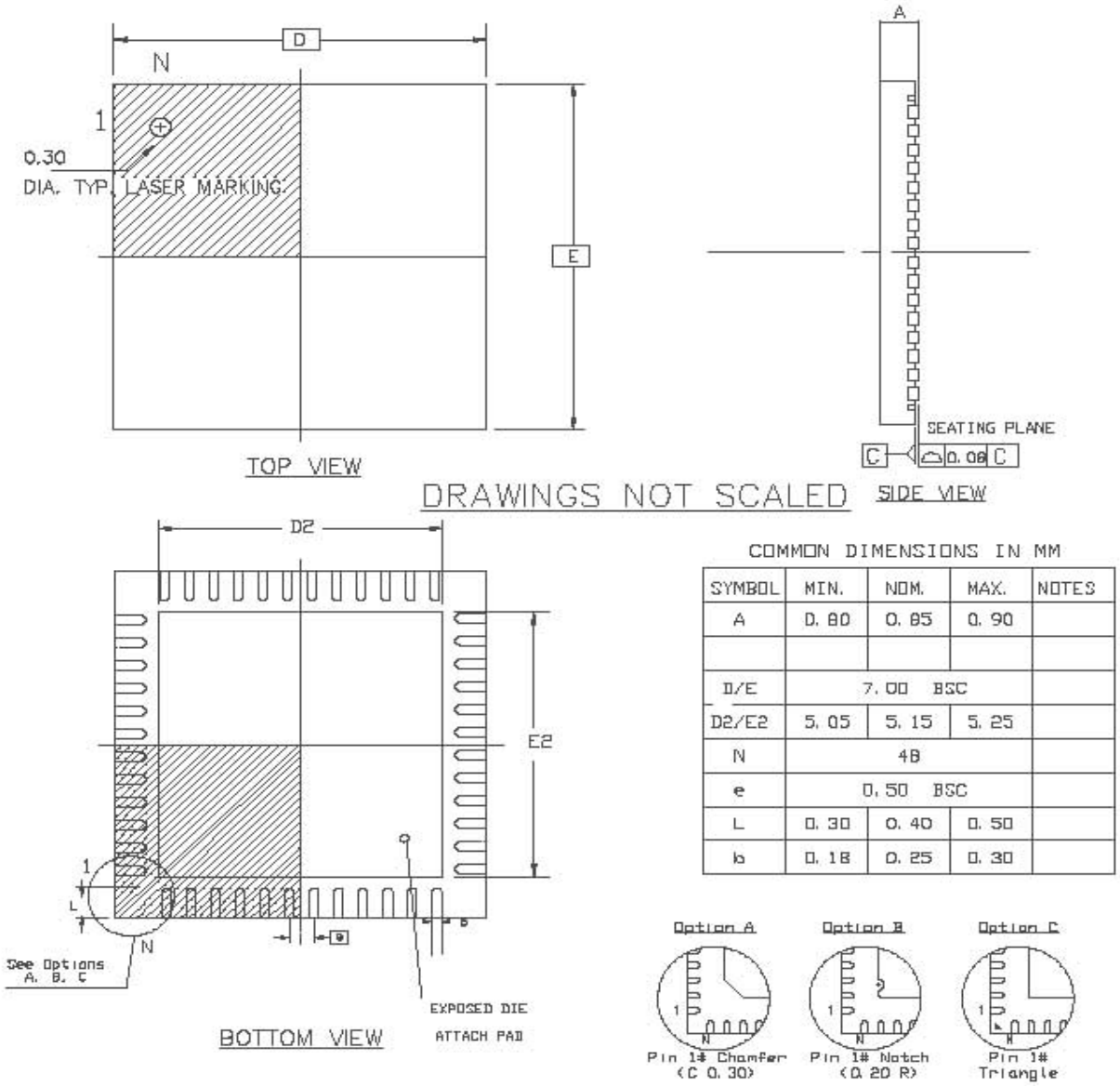
Table 8-3. Package Characteristics

Moisture Sensitivity Level	TBD

Table 8-4. Package Reference

JEDEC Drawing Reference	MS-026
JESD97 Classification	E3

Figure 8-2. QFN-48 Package Drawing



SYMBOL	MIN.	NOM.	MAX.	NOTES
A	0.80	0.85	0.90	
D/E		7.00 BSC		
D2/E2	5.05	5.15	5.25	
N		48		
e		0.50 BSC		
L	0.30	0.40	0.50	
b	0.18	0.25	0.30	

Table 8-5. Device and Package Maximum Weight

TBD	mg

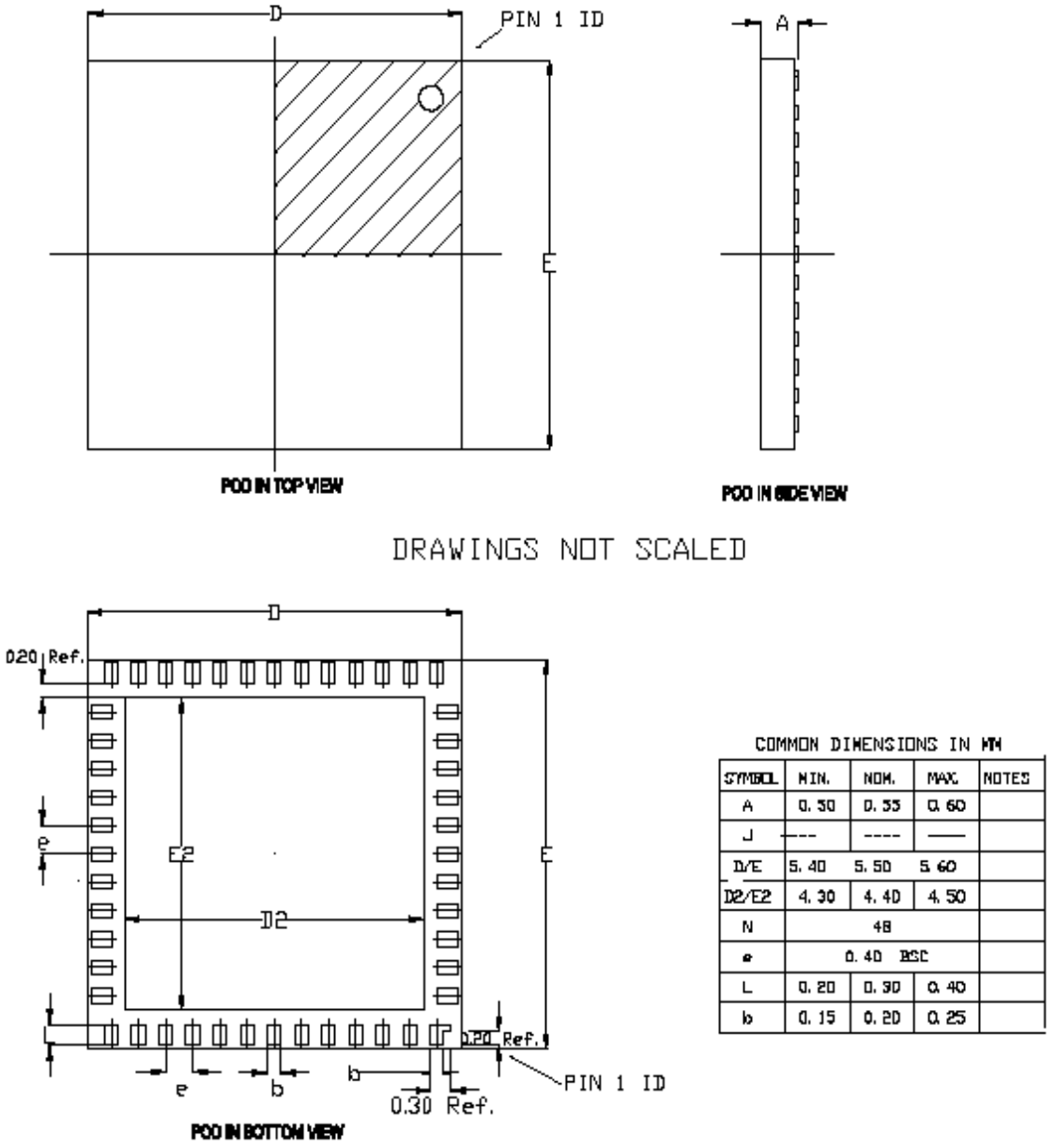
Table 8-6. Package Characteristics

Moisture Sensitivity Level	TBD

Table 8-7. Package Reference

JEDEC Drawing Reference	M0-220
JESD97 Classification	E3

Figure 8-3. TLLGA-48 Package Drawing



SYMBOL	MIN.	NOM.	MAX.	NOTES
A	0.50	0.55	0.60	
J	----	----	----	
D/E	5.40	5.50	5.60	
D2/E2	4.30	4.40	4.50	
N	48			
e	0.40 BSC			
L	0.20	0.30	0.40	
b	0.15	0.20	0.25	

Table 8-8. Device and Package Maximum Weight

TBD	mg

Table 8-9. Package Characteristics

Moisture Sensitivity Level	TBD

Table 8-10. Package Reference

JEDEC Drawing Reference	M0-220
JESD97 Classification	E3

8.3 Soldering Profile

Table 8-11 gives the recommended soldering profile from J-STD-20.

Table 8-11. Soldering Profile

Profile Feature	Green Package
Average Ramp-up Rate (217°C to Peak)	3°C/s max
Preheat Temperature 175°C ±25°C	60-120 s
Temperature Maintained Above 217°C	60-150 s
Time within 5°C of Actual Peak Temperature	30 s
Peak Temperature Range	260°C
Ramp-down Rate	6°C/s max
Time 25°C to Peak Temperature	8 minutes max

A maximum of three reflow passes is allowed per component.

9. Ordering Information

Table 9-1. Ordering Information

Device	Ordering Code	Carrier Type	Package	Package Type	Temperature Operating Range
AT32UC3L064	AT32UC3L064-AUT	Tray	TQFP 48	JESD97 Classification E3	Industrial (-40°C to 85°C)
	AT32UC3L064-AUR	Tape & Reel	TQFP 48		
	AT32UC3L064-ZAUT	Tray	QFN 48		
	AT32UC3L064-ZAUR	Tape & Reel	QFN 48		
	AT32UC3L064-D3UR	Tape & Reel	TLLGA 48		
AT32UC3L032	AT32UC3L032-AUT	Tray	TQFP 48		
	AT32UC3L032-AUR	Tape & Reel	TQFP 48		
	AT32UC3L032-ZAUT	Tray	QFN 48		
	AT32UC3L032-ZAUR	Tape & Reel	QFN 48		
	AT32UC3L032-D3UR	Tape & Reel	TLLGA 48		
AT32UC3L016	AT32UC3L016-AUT	Tray	TQFP 48		
	AT32UC3L016-AUR	Tape & Reel	TQFP 48		
	AT32UC3L016-ZAUT	Tray	QFN 48		
	AT32UC3L016-ZAUR	Tape & Reel	QFN 48		
	AT32UC3L016-D3UR	Tape & Reel	TLLGA 48		

10. Errata

10.1 Rev. C

10.1.1 SCIF

1. **A reset from Supply Monitor 33 will be registered as POR**
   A Supply Monitor 33 reset will not be detected in the Reset Cause register (RCAUSE) as BOD33, it will be detected as a Power-on Reset (POR).
   Fix/Workaround
   None.

10.1.2 SPI

1. **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.

2. **SPI Bad Serial Clock Generation on 2nd chip select when SCBR = 1, CPOL=1, and NCPHA=0**
   When multiple CS are in use, if one of the baudrates equals 1 and one of the others does not equal 1, and CPOL=1 and CPHA=0, 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 CPOL=1 and CPHA=0.

3. **SPI data transfer hangs with CSAAT=1 in CSR0 and MODFDIS=0 in MR**
   When CSAAT=1 in CSR0 and mode fault detection is enabled (MODFDIS=0 in MR), the SPI module will not start a data transfer.
   Fix/Workaround
   Disable mode fault detection by writing a one to MODFDIS in MR.

4. **Disabling SPI has no effect on the TDRE flag**
   Disabling SPI has no effect on TDRE whereas the write data command is filtered when SPI is disabled. This means that as soon as the SPI is disabled it becomes impossible to reset the TDRE flag by writing in the TDR. So if the SPI is disabled during a PDCA transfer, the PDCA will continue to write data in the TDR (as TDRE stays high) until its buffer is empty, and all data written after the disable command is lost.
   Fix/Workaround
   Disable the PDCA, 2 NOP (minimum), disable SPI. When you want to continue the transfer: Enable SPI, enable PDCA.

10.2 Rev. B

10.2.1 Processor and Architecture

1. **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 described in general as not safe in the UC technical reference manual, it is safe in this very specific case.
2. Execute the RETE instruction.

10.2.2 FLASHCDW

1. **Chip erase**
When performing chip erase, the device may report that it is protected (IR=0x11) and that chiperase failed, even if the chip erase was succesful.
Fix/workaround
Perform a reset before any further read and programming.

2. **Fuse programming**
Programming of fuses does not work.
Fix/workaround
Do not program fuses. All fuses will be erased during chiperase command.

3. **Wait 500 ns before reading from the flash after switching read mode**
After switching between normal read mode and high-speed read mode, the application must wait at least 500 ns before attempting any access to the flash.
Fix/workaround
Two workarounds exist:
1. Make sure that the appropriate instructions are executed from RAM, and that a waiting-loop is executed from RAM waiting 500ns or more before executing from flash.
2. Execute from flash with a clock with period longer than 500 ns. This guarantees that no new read access is attempted before the flash has had time to settle in the new read mode.

4. **VERSION register reads 0x100**
The VERSION register reads 0x100 instead of 0x102.
Fix/Workaround
None.

10.2.3 HMATRIX

1. **In the HMATRIX PRAS and PRBS registers MxPR fields are only two bits**
In the HMATRIX PRAS and PRBS registers 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.

10.2.4 PDCA

1. **PCONTROL.CHxRES is nonfunctional**
PCONTROL.CHxRES is nonfunctional. Counters are reset at power-on, and cannot be reset by software.
Fix/Workaround
SW needs to keep history of performance counters.

2. **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.

3. **VERSION register reads 0x120**
The VERSION register reads 0x120 instead of 0x122.

Fix/Workaround
None.

10.2.5 GPIO

1. **GPIO interrupt flag can not be cleared when interrupts are disabled**
   The GPIO interrupt flag can not be cleared unless the interrupt is enabled for the pin.
   Fix/workaround
   Enable interrupt for the corresponding pin, then clear the interrupt flag.

2. **VERSION register reads 0x210**
   The VERSION register reads 0x210 instead of 0x211.
   Fix/Workaround
   None.

10.2.6 PM

1. **OCP and high frequency clock sources**
   OCP does not work if the main clock source is a high frequency clock. If the frequency of the source exceeds the maximum frequency of the CRIPOSC the OCP will generate an interrupt and switch clock source to the slow clock upon enabling the OCP, even if the CPU clock is divided to a legal frequency.
   Fix/Workaround
   Do not use clock sources with frequencies higher that the maximum CPU frequency while using the OCP.

2. **CONFIG register reads 0x4F**
   The CONFIG register reads 0x4F instead of 0x43.
   Fix/Workaround
   None.

3. **PB writes via debugger in sleep modes are blocked during sleepwalking**
   During sleepwalking, PB writes performed by a debugger will be discarded by all PB modules except the module that is requesting the clock.
   Fix/workaround
   None.

4. **VERSION register reads 0x400**
   The VERSION register reads 0x400 instead of 0x411.
   Fix/Workaround
   None.

5. **WCAUSE register should not be used**
   The WCAUSE register should not be used.
   Fix/Workaround
   None.

6. **Clock failure detector does not work**
   In some cases the clock failure detector will not detect if the CPU clock stops. In this case the CPU will halt operation.
   Fix/Workaround
   None.

10.2.7 SCIF

1. **A reset from Supply Monitor 33 will be registered as POR**

A Supply Monitor 33 reset will not be detected in the Reset Cause register (RCAUSE) as BOD33, it will be detected as a Power-on Reset (POR).
Fix/Workaround
None.

2. **The DFLL should be slowed down before disabled**
The frequency of the DFLL should be set to minimum before disabled.
Fix/Workaround
Before disabling the DFLL the value of the COARSE register should be set to zero.

3. **Writing to SCIF ICR masks new interrupts received in the same clock cycle**
Writing to SCIF ICR masks any new SCIF interrupt received in the same clock cycle, regardless of write value.
Fix/Workaround:
For every interrupt except BODDET, SM33DET, and VREGOK the CLKSR register can be read to detect new interrupts. BODDET, SM33DET and VREGOK interrupts will not be generated if they occur when writing SCIF ICR.

4. **FINE value for DFLL is not correct when dithering is disabled**
In open loop mode, the FINE value used by the DFLL DAC is offseted by two compared to the value written to the DFLL0CONF.FINE field. I. e. the value to the DFLL DAC is DFLL0CONF.FINE-0x002. If DFLL0CONF.FINE is written to 0x000, 0x001 or 0x002 the value to the DFLL DAC will be 0x1FE, 0x1FF or 0x000 respectively.
Fix/workaround
Write the desired value added by two to the DFLL0CONF.FINE field.

5. **BODVERSION register reads 0x100**
The BODVERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

7. **BRIFA is non-functional**
BRIFA is non-functional.
Fix/Workaround
None.

8. **VREGCR DEEPMODEDISABLE bit is not readable**
VREGCR DEEPMODEDISABLE bit is not readable.
Fix/workaround
None.

9. **DFLL step size should be 7 or lower below 30 MHz**
If max step size is above 7, the DFLL might not lock at the correct frequency if the target frequency is below 30 MHz.
Fix/Workaround
If the target frequency is below 30 MHz, use max step size (DFLL0MAXSTEP.MAXSTEP) of 7 or lower.

10. **Generic clock sources are kept running in sleep modes**
If a clock is used as a source for a generic clock when going to a sleep mode where clock sources are stopped, the source of the generic clock will be kept running. Please refer to the Power Manager chapter for details about sleep modes.
Fix/Workaround

Disable generic clocks before going to sleep modes where clock sources are stopped to save power.

11. **DFLL clock is unstable with a fast reference clock**
The DFLL clock can be unstable when a fast clock is used as reference clock in closed loop mode.
Fix/Workaround
Use the 32 KHz crystal oscillator clock or a clock with similar frequency as DFLLIF reference clock.

12. **DFLLIF indicates coarse lock too early**
The DFLLIF might indicate coarse lock too early, the DFLL will lose coarse lock and regain it later.
Fix/Workaround
Use max step size (DFLL0MAXSTEP.MAXSTEP) of 4 or higher.

13. **DFLLIF dithering does not work**
The DFLLIF dithering does not work.
Fix/Workaround
None.

14. **SCIF VERSION register reads 0x100**
The VERSION register reads 0x100 instead of 0x102.
Fix/Workaround
None.

15. **DFLLVERSION register reads 0x200**
The DFLLVERSION register reads 0x200 instead of 0x201.
Fix/Workaround
None.

16. **RCCRVERSION register reads 0x100**
The RCCRVERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

17. **OSC32VERSION register reads 0x100**
The OSC32VERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

18. **VREGVERSION register reads 0x100**
The VREGVERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

19. **RC120MVERSION register reads 0x100**
The RC120MVERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

10.2.8 WDT

1. **Clearing of the WDT in window mode**

In window mode, if the WDT is cleared 2^{TBAN} CLK_WDT cycles after entering the window. The counter will be cleared, but will not exit the window. If this occurs, the SR.WINDOW bit will not be cleared after clearing the WDT.
Fix/Workaround
Check SR.WINDOW immediately after clearing the WDT. If set then clear the WDT once more.

2. **VERSION register reads 0x400**
The VERSION register reads 0x400 instead of 0x402.
Fix/Workaround
None.

10.2.9 SPI

1. **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.

2. **SPI Bad Serial Clock Generation on 2nd chip select when SCBR = 1, CPOL=1, and NCPHA=0**
When multiple CS are in use, if one of the baudrates equals 1 and one of the others does not equal 1, and CPOL=1 and CPHA=0, 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 CPOL=1 and CPHA=0.

3. **SPI data transfer hangs with CSAAT=1 in CSR0 and MODFDIS=0 in MR**
When CSAAT=1 in CSR0 and mode fault detection is enabled (MODFDIS=0 in MR), the SPI module will not start a data transfer.
Fix/Workaround
Disable mode fault detection by writing a one to MODFDIS in MR.

4. **Disabling SPI has no effect on the TDRE flag**
Disabling SPI has no effect on TDRE whereas the write data command is filtered when SPI is disabled. This means that as soon as the SPI is disabled it becomes impossible to reset the TDRE flag by writing in the TDR. So if the SPI is disabled during a PDCA transfer, the PDCA will continue to write data in the TDR (as TDRE stays high) until its buffer is empty, and all data written after the disable command is lost.
Fix/Workaround
Disable the PDCA, 2 NOP (minimum), disable SPI. When you want to continue the transfer: Enable SPI, enable PDCA.

10.2.10 TWI

1. **TWIM Version Register is zero**
TWIM Version Register (VR) reads zero instead of 0x101.
Fix/Workaround
none.

2. **TWIS Version Register is zero**
TWIS Version Register (VR) reads zero instead of 0x112.
Fix/Workaround
None.

3. **TWIS CR.STREN does not work in deep sleep modes**

When the device is in Stop, DeepStop or Static sleep modes, address reception will not wake device if both CR.SOAM and CR.STREN are set.
Fix/workaround
Do not set both CR.STREN and CR.SOAM if the device needs to wake from deep sleep.

4. **TWI pads are not SMBUS compatible**
The TWI pads draws current when the pins are supplied with 3.3V and the part is left unpowered.
Fix/workaround
None.

5. **PA21, PB04, and PB05 are not 5V tolerant**
Pins PA21, PB04, and PB05 are only 3.3 V tolerant.
Fix/workaround
None.

6. **PB04 SMBALERT function should not be used**
The SMBALERT function from TWIMS0 should not be selected on pin PB04.
Fix/workaround
None.

7. **TWI0.TWCK on PB05 is non-functional**
TWI0.TWCK on PB05 is non-functional.
Fix/workaround
Use TWI0.TWCK on other pins.

8. **TWIM STOP bit in IMR always read as zero**
The STOP bit in IMR always reads as zero.
Fix/workaround
None.

10.2.11 PWMA

1. **PARAMETER register reads 0x2424**
The PARAMETER register reads 0x2424 instead of 0x24.
Fix/Workaround
None.

2. **Open Drain mode does not work**
The open drain mode does not work.
Fix/workaround
None.

3. **VERSION register reads 0x100**
The VERSION register reads 0x100 instead of 0x101.
Fix/Workaround
None.

4. **Writing to the duty cycle registers when the timebase counter overflows can give undefined result**
The duty cycle registers will be corrupted if written when the timebase counter overflows. If the duty cycle registers are written exactly when the timebase counter overflows at TOP, the duty cycle registers may become corrupted.
Fix/workaround

Write to the duty cycle registers only directly after the Timebase Overflow bit in the status register is set.

10.2.12 SAU

1. **Idle bit reads as zero**
   The idle bit reads as zero.
   Fix/workaround
   None.

2. **Open mode is not functional**
   The open mode is not functional.
   Fix/workaround
   None.

3. **VERSION register reads 0x100**
   The VERSION register reads 0x100 instead of 0x110.
   Fix/Workaround
   None.

10.2.13 ADCIFB

1. **Pendetect in sleep modes without CLK_ADCIFB will not wake the system**
   The pendetect will not wake the system from a sleep mode if the clock for the ADCIFB (CLK_ADCIFB) is turned off.
   Fix/Workaround
   Use a sleep mode where CLK_ADCIFB is not turned off to wake the part using pendetect.

2. **8-bit mode is not working**
   Do not use the 8-bit mode of the ADCIFB.
   Fix/Workaround
   Use the 10-bit mode and shift right by 2 bits.

3. **ADC channels six to eight is non-functional**
   ADC channels six to eight is non-functional.
   Fix/Workaround
   None.

4. **VERSION register reads 0x100**
   The VERSION register reads 0x100 instead of 0x101.
   Fix/Workaround
   None.

10.2.14 ACIFB

1. **Negative offset**
   The static offset of the analog comparator is appriximately -50mV.
   Fix/Workaround
   None.

2. **Generic clock sources in sleep modes**
   The ACIFB should not use RC32K, or CLK_1K as generic clock source if the chip uses sleep modes.
   Fix/Workaround
   None.

3. **VERSION register reads 0x200**
   The VERSION register reads 0x200 instead of 0x212.
   Fix/Workaround
   None.

4. **CONFW.WEVSRC and CONFW.WEVEN are not correctly described in the user interface**
   CONFW.WEVSRC is only two bits instead of three bits wide. Only values 0, 1, and 2 can be written to this register. CONFW.WEVEN is in bit position 10 instead of 11.
   Fix/workaround
   Only write values 0, 1, and 2 to CONFW.WEVSRC. When reading CONFW.WEVSRC, disregard the third bit. Read/write bit 10 to access CONFW.WEVEN.

10.2.15 USART

1. **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.

10.2.16 TC

1. **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.

10.2.17 CAT

1. **Switch off discharge current when reaching 0V**
   The discharge current will switch off when reaching MGCFG1.MAX, not when reaching 0V.
   Fix/workaround
   None.

2. **CAT external capacitors are not clamped to ground when CAT is idle**
   The CAT module does not clamp the external capacitors to ground when it is idle. The capacitors are left floating, so they could accumulate small amounts of charge.
   Fix/workaround
   None.

3. **CAT DISHIFT field is stuck at zero**
   The DISHIFT field in the MGCFG1, TGACFG1, TGBCFG1, and ATCFG1 registers is stuck at zero and cannot be written to a different value. Capacitor discharge time will be determined only by the DILEN field.
   Fix/workaround
   None.

4. **CAT ACCTRL bit is stuck at zero**

The ACCTRL bit in the MGCFG2 register is stuck at zero and cannot be written to one. The analog comparators will be constantly enabled.
Fix/workaround
None.

5. **CAT CONSEN field is stuck at zero**
The CONSEN field in the MGCFG2 register is stuck at zero and cannot be written to a different value. The CAT consensus filter does not function properly, so termination of QMatrix data acquisition is controlled only by the MAX field in MGCFG1.
Fix/workaround
None.

6. **VERSION register reads 0x100**
The VERSION register reads 0x100 instead of 0x200.
Fix/Workaround
None.

10.2.18 aWire

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

2. **SAB multiaccess reads are not working**
Reading more than one word, halfword, or byte in one command is not workingcorrectly.
Fix/workaround
Split the access into several single word, halfword, or byte accesses.

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

4. **aWire enable does not work in static mode**
aWire enable does not work in static mode.
Fix/workaround
None.

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

10.2.19 GLOC

1. **GLOC is non-functional**
Gloc is non-functional.
Fix/workaround
None.

10.2.20 I/O pins

1. **PB10 is not 3.3V tolerant**
PB10 should be grounded on the PCB and left unused.
Fix/workaround

None.

2. **Analog multiplexing consumes extra power**
   Current consumption on VDDIO increases when the voltage on analog inputs is close to VDDIO/2.
   Fix/workaround
   None.

3. **PA02, PB01, PB04, PB05, RESET_N have half of the pullup strength**
   Pins PA02, PB01, PB04, PB05, RESET_N have half of the specified pullup strength.
   Fix/workaround
   None.

4. **OCD MCKO and MDO[3] are swapped in the AUX1 mapping**
   When using the OCD AUX1 mapping of trace signals MDO[3] is located on pin PB05 and MCKO is located on PB01.
   Fix/workaround
   Swap pins PB01 and PB05 if using OCD AUX1.

10.2.21 Chip

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

2. **Shutdown mode is not functional**
   Do not enter shutdown mode.
   Fix/workaround
   None.

3. **Static mode cannot be entered if the WDT is using OSC32**
   If the WDT is using OSC32 as clock source and the user tries to enter the static sleep mode, the DeepStop sleep mode will be entered instead.
   Fix/workaround
   None.

4. **VDDIN current consumption increase above 1.8V**
   When VDDIN increases above 1.8 V, current on VDDIN increases with up to 40 uA.
   Fix/workaround
   None.

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. A – 06/09

1. Initial revision.

Table of Contents

Headquarters

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

International

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

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

Product Contact

Web Site
www.atmel.com

Technical Support
avr32@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature