Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Features

- High Performance, Low Power Atmel® 32-bit AVR® 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 50 MHz from Flash (1 Flash Wait State)
    - Up to 36 DMIPS Running at 25 MHz from Flash (0 Flash Wait State)
  - Memory Protection Unit (MPU)
    - Secure Access Unit (SAU) providing User Defined Peripheral Protection
- 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
  - 64 Kbytes, 32 Kbytes, and 16 Kbytes Versions
  - Single-cycle Access up to 25 MHz
  - FlashVault™ Technology Allows Pre-programmed Secure Library Support for End User Applications
  - Prefetch Buffer Optimizing Instruction Execution at Maximum Speed
  - 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
  - 16 Kbytes (64 Kbytes and 32 Kbytes Flash), or 8 Kbytes (16 Kbytes 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 150 MHz 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 Interface (TWI), 400 kbit/s I^2C-compatible
- One 8-channel Analog-to-digital Converter (ADC) with up to 12 Bits Resolution
  - Internal Temperature Sensor

32-bit AVR® Microcontroller

AT32UC3L064
AT32UC3L032
AT32UC3L016

Preliminary

32099G–06/2011

- Eight Analog Comparators (AC) with Optional Window Detection
- Capacitive Touch (CAT) Module
  - Hardware Assisted Atmel® AVR® QTouch® and Atmel® AVR® QMatrix® Touch Acquisition
  - Supports QTouch and QMatrix Capture from Capacitive Touch Sensors
- QTouch Library Support
  - Capacitive Touch Buttons, Sliders, and Wheels
  - QTouch and QMatrix Acquisition
- 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 1.62-3.6 V Power Supply

1. Description

The Atmel® AVR® 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. The Secure Access Unit (SAU) is used together with the MPU to provide the required security and integrity.

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 current consumption down to 165µA/MHz, and leakage down to 9nA 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 40 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. The USART supports different communication modes, like SPI Mode and LIN Mode.

A general purpose 8-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 16 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.

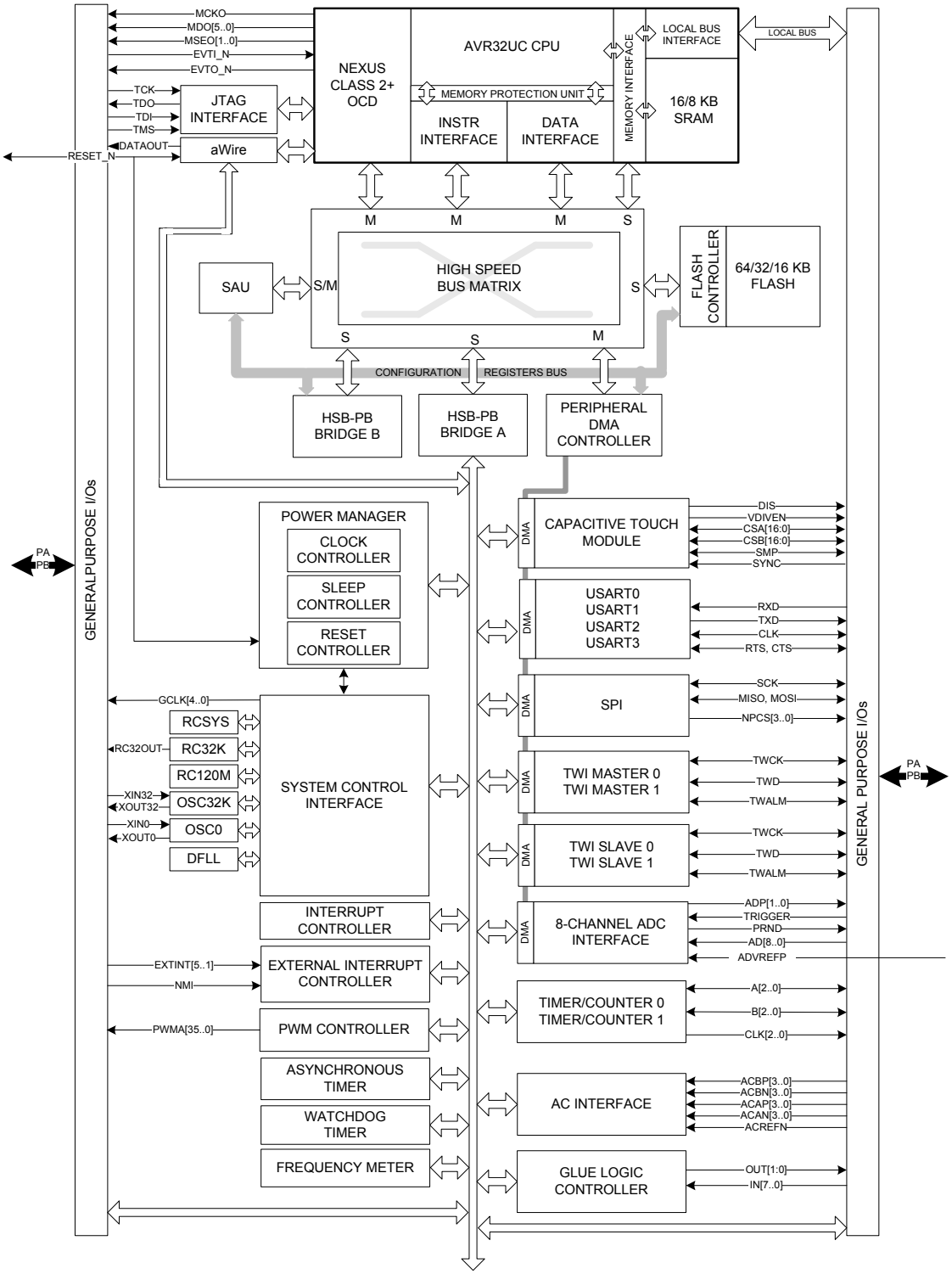
Atmel offers the QTouch library for embedding capacitive touch buttons, sliders, and wheels functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys as well as Adjacent Key Suppression® (AKS®) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop, and debug your own touch applications.

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	AT32UC3L064	AT32UC3L032	AT32UC3L016
Flash	64KB	32KB	16KB
SRAM	16KB	16KB	8KB
GPIO	36		
High-drive pins	5		
External Interrupts	6		
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		
Secure Access Unit	1		
Glue Logic Controller	1		
Oscillators	Digital Frequency Locked Loop 40-150MHz (DFLL) Crystal Oscillator 3-16MHz (OSC0) Crystal Oscillator 32KHz (OSC32K) RC Oscillator 120MHz (RC120M) RC Oscillator 115kHz (RCSYS) RC Oscillator 32kHz (RC32K)		
ADC	8-channel 12-bit		
Temperature Sensor	1		
Analog Comparators	8		
Capacitive Touch Module	1		
JTAG	1		
aWire	1		
Max Frequency	50MHz		
Packages	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 Pinout

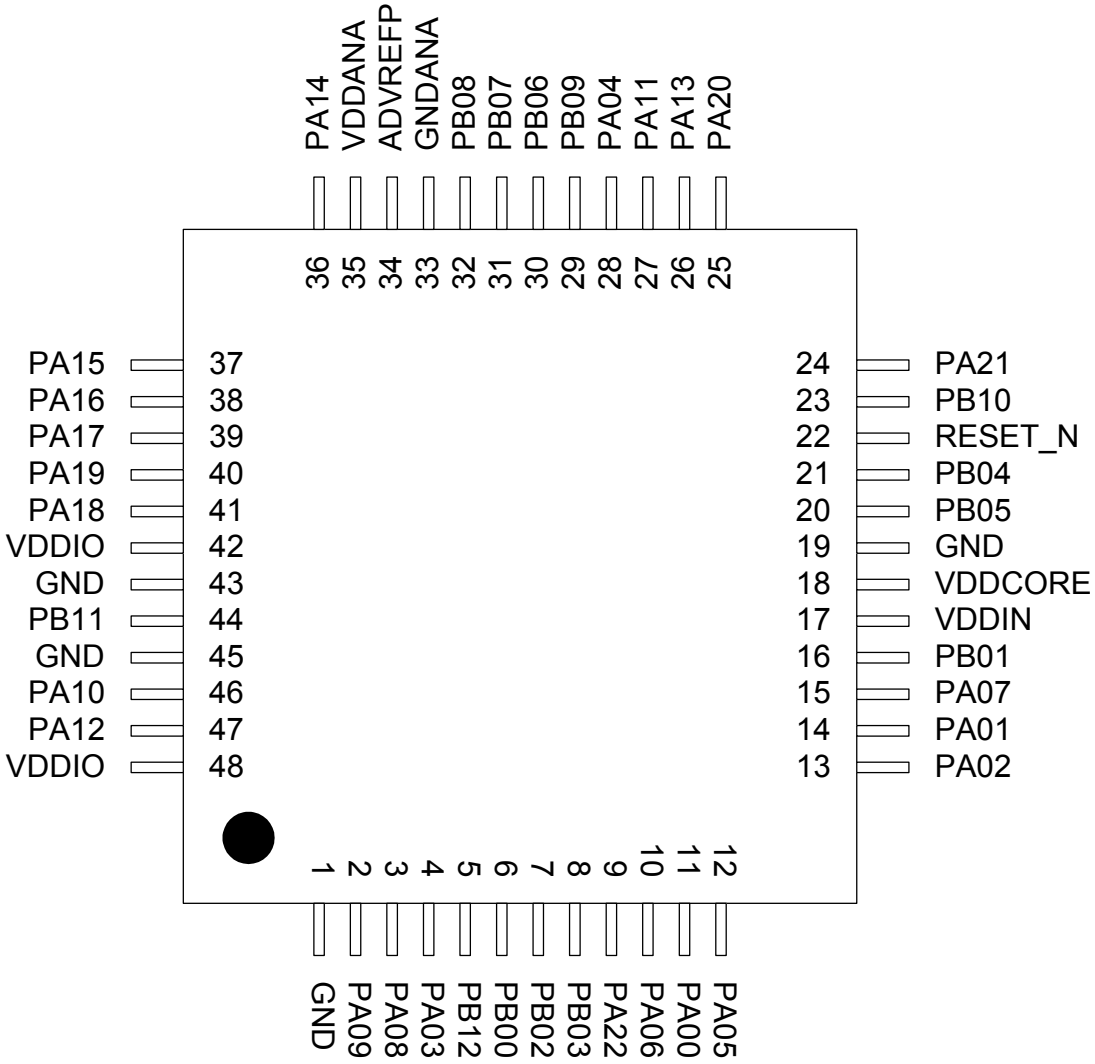
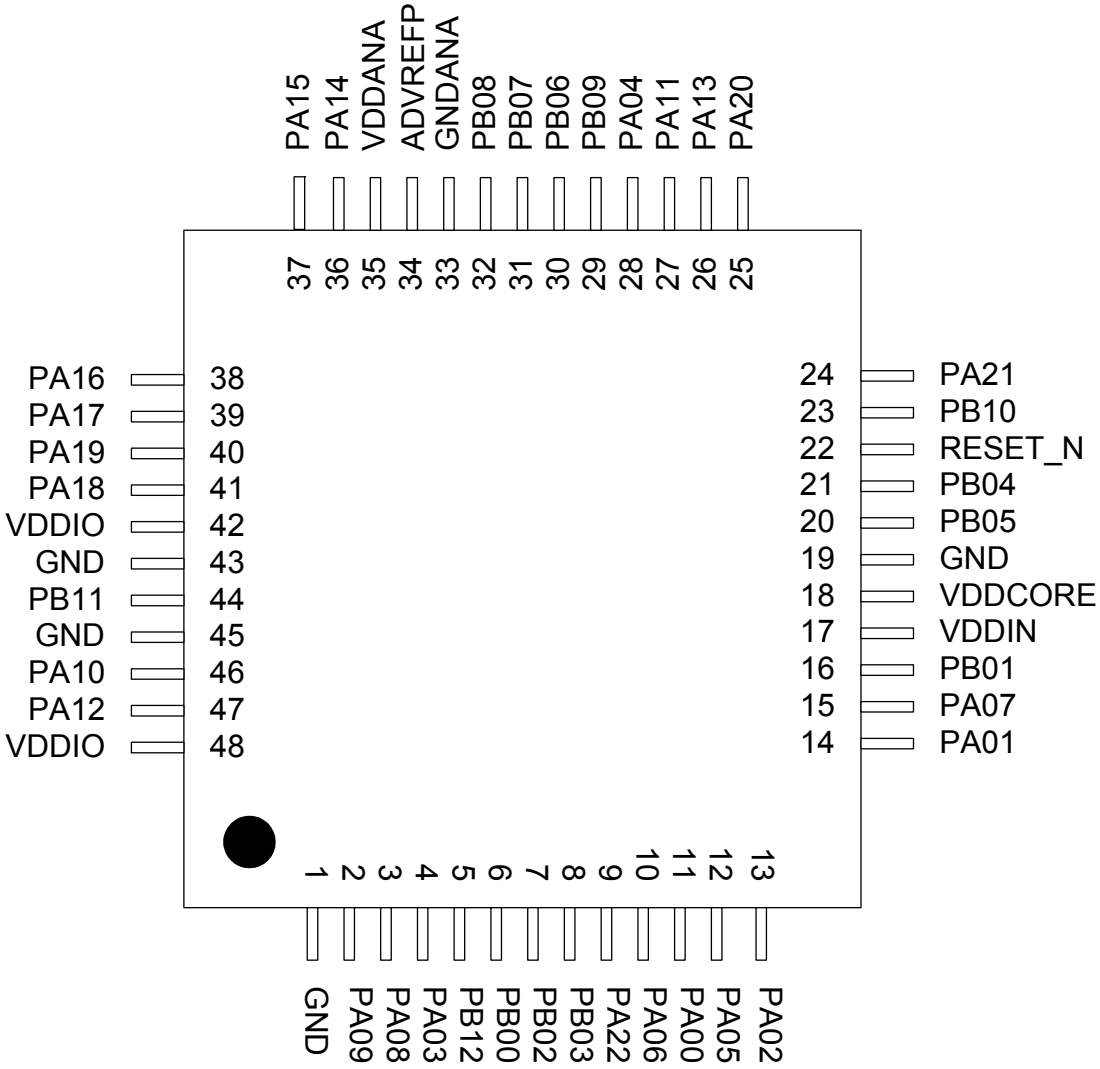


Figure 3-2. TLLGA48 Pinout



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

48-pin	PIN	GPIO	Supply	Pin 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	Normal I/O (TWI)	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	Normal I/O (TWI)	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	Normal I/O (TWI)		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]

Table 3-1. GPIO Controller Function Multiplexing

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		TC0 A1	GLOC IN[3]	PWMA PWMA[20]	SCIF RC32OUT		CAT CSA[12]
24	PA21	21	VDDIN	Normal I/O (TWI, 5V tolerant SMBus)	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	Normal I/O (TWI, 5V tolerant SMBus)	TC1 A0	USART1 RTS	USART1 CLK	TWIMS0 TWALM	PWMA PWMA[27]	PWMA PWMAOD[27]	TWIMS1 TWCK	CAT CSA[14]
20	PB05	37	VDDIN	Normal I/O (TWI, 5V tolerant SMBus)	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.

Refer to "Electrical Characteristics" on page 41 for a description of the electrical properties of the pin types used.

3.2.1.1 TWI, 5V Tolerant, and SMBUS Pins

Some Normal I/O pins have TWI, 5V Tolerant, and SMBUS features. These features are available only when TWI functions or the PWMAOD function of the PWMA are selected on these pins.

Refer to the "TWI Pin Characteristics(1)" on page 49 for a description of the electrical properties of the TWI, 5V Tolerant, and SMBUS pins.

3.2.2 Peripheral Functions

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

Table 3-2. Peripheral Functions

Function	Description
GPIO Controller Function multiplexing	GPIO and GPIO peripheral selection A to H
Nexus OCD AUX port connections	OCD trace system
aWire DATAOUT	aWire output in two-pin mode
JTAG port connections	JTAG debug port
Oscillators	OSC0, OSC32

3.2.3 JTAG Port Connections

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

Table 3-3. JTAG Pinout

48-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, irrespectively 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

Table 3-4. Nexus OCD AUX Port Connections

Pin	AXS=1	AXS=0
MDO[2]	PA16	PB03
MDO[1]	PA15	PB02
MDO[0]	PA14	PB09
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

48-pin	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 active after the aWire is enabled and the 2_PIN_MODE command has been sent. The WAKE_N pin is always enabled. Please refer to Section 6.1.4 on page 40 for constraints on the WAKE_N pin.

Table 3-6. Other Functions

48-pin	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 resistive touch screen	Output		
PRND	Pseudorandom output signal	Output		
TRIGGER	External trigger	Input		
aWire - AW				
DATA	aWire data	I/O		
DATAOUT	aWire data output for 2-pin mode	I/O		
Capacitive Touch Module - CAT				
CSA16 - CSA0	Capacitive Sense A	I/O		
CSB16 - CSB0	Capacitive Sense B	I/O		
DIS	Discharge current control	Analog		
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		

Table 3-7. Signal Descriptions List

TMS	Test Mode Select	Input		
Power Manager - PM				
RESET_N	Reset	Input	Low	
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		

Table 3-7. Signal Descriptions List

Universal Synchronous/Asynchronous Receiver/Transmitter - USART0, USART1, USART2, USART3				
CLK	Clock	I/O		
CTS	Clear To Send	Input	Low	
RTS	Request To Send	Output	Low	
RXD	Receive Data	Input		
TXD	Transmit Data	Output		

Note: 1. ADCIFB: AD3 does not exist.

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		1.62V to 1.98V
VDDIO	I/O Power Supply	Power Input		1.62V to 3.6V. VDDIO should always be equal to or lower than VDDIN.
VDDANA	Analog Power Supply	Power Input		1.62V to 1.98V
ADVREFP	Analog Reference Voltage	Power Input		1.62V to 1.98V
VDDIN	Voltage Regulator Input	Power Input		1.62V 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				
PA22 - PA00	Parallel I/O Controller I/O Port 0	I/O		
PB12 - PB00	Parallel I/O Controller I/O Port 1	I/O		

1. See Section 6.1 on page 36

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. The TCK pin always has pull-up enabled during reset. The TDO pin is an output, driven at VDDIO, and has no pull-up resistor. The JTAG pins can be used as GPIO pins and multiplexed with peripherals when the JTAG is disabled. Please refer to Section 3.2.3 on page 11 for the JTAG port connections.

3.4.2 PA00

Note that PA00 is multiplexed with TCK. PA00 GPIO function must only be used as output in the application.

3.4.3 RESET_N Pin

The RESET_N pin is a schmitt input and integrates a permanent pull-up resistor to VDDIN. 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 external circuitry.

3.4.4 TWI Pins PA21/PB04/PB05

When these pins are used for TWI, the pins are open-drain outputs with slew-rate limitation and inputs with spike filtering. When used as GPIO pins or used for other peripherals, the pins have the characteristics indicated in the Electrical Characteristics section. Selected pins are also SMBus compliant (refer to Section 3.2 on page 9). As required by the SMBus specification, these pins provide no leakage path to ground when the AT32UC3L is powered down. This allows other devices on the SMBus to continue communicating even though the AT32UC3L is not powered.

After reset a TWI function is selected on these pins instead of the GPIO. Please refer to the GPIO Module Configuration chapter for details.

3.4.5 TWI Pins PA05/PA07/PA17

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

After reset a TWI function is selected on these pins instead of the GPIO. Please refer to the GPIO Module Configuration chapter for details.

3.4.6 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. PA20 selects SCIF-RC32OUT (GPIO Function F) as default enabled after reset.

3.4.7 High-Drive Pins

The five pins PA02, PA06, PA08, PA09, and PB01 have high-drive output capabilities. Refer to Section 7. on page 41 for electrical characteristics.

3.4.8 RC32OUT Pin

3.4.8.1 Clock output at startup

After power-up, the clock generated by the 32kHz RC oscillator (RC32K) will be output on 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 PA20, but will be disabled if 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 written to zero (refer to the Power Manager chapter)

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

Once the RC32K output on PA20 is disabled it can never be enabled again.

3.4.8.2 XOUT32_2 function

PA20 selects RC32OUT as default enabled after reset. This function is not automatically disabled when the user enables the XOUT32_2 function on PA20. This disturbs the oscillator and may result in the wrong frequency. To avoid this, RC32OUT must be disabled when XOUT32_2 is enabled.

3.4.9 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.

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 extension 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

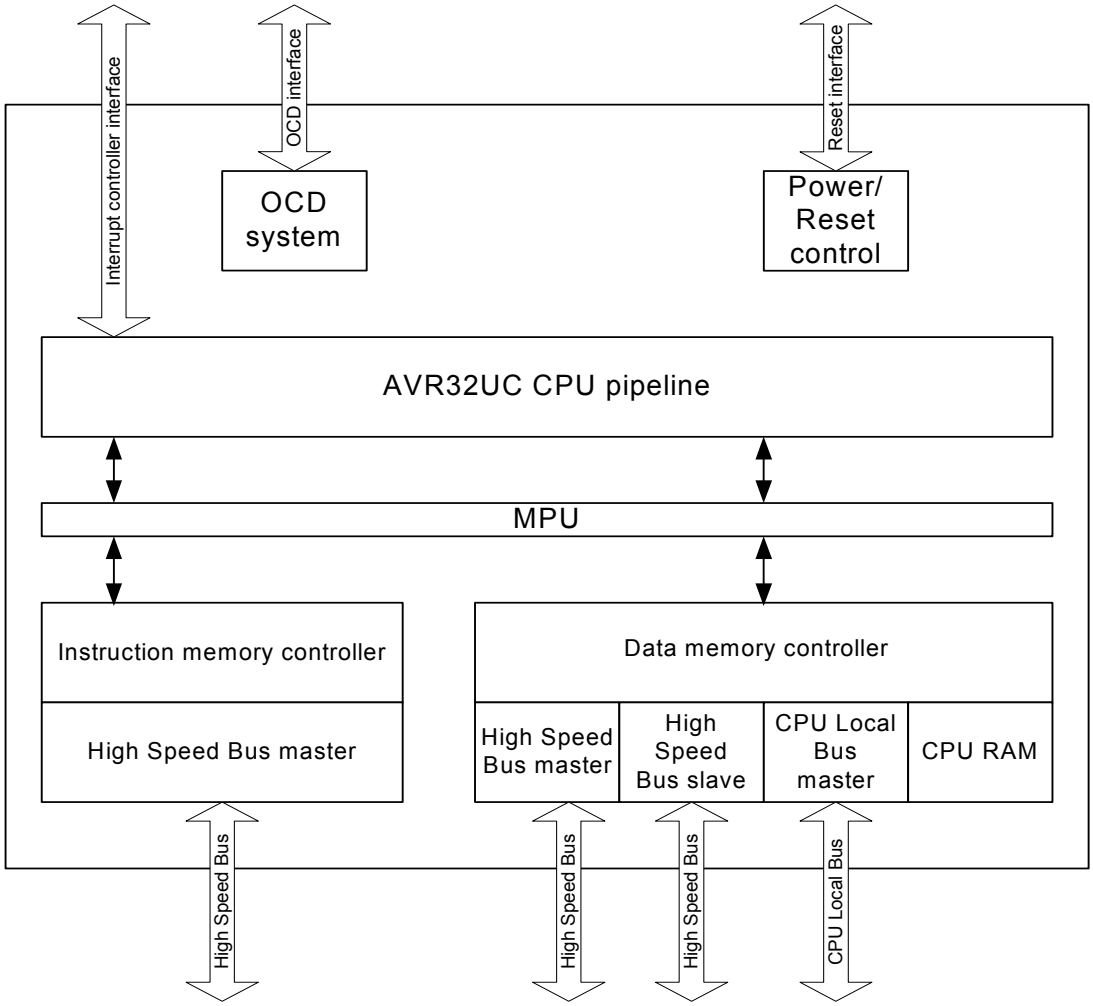
The AVR32UC CPU targets low- and medium-performance applications, and provides an advanced On-Chip Debug (OCD) system, no caches, and a Memory Protection Unit (MPU). Java acceleration hardware is not implemented.

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 20 displays the contents of AVR32UC.

Figure 4-1. Overview of the AVR32UC CPU



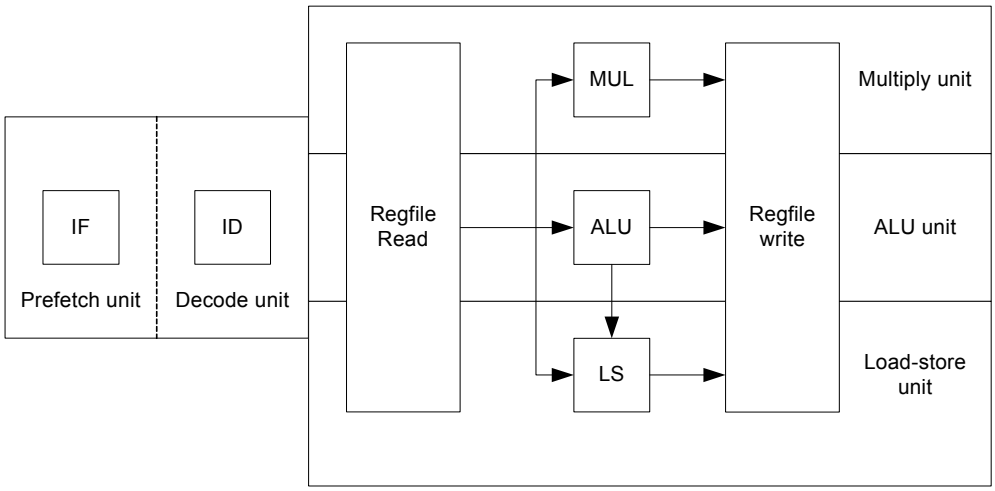
4.3.1 Pipeline Overview

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 21 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

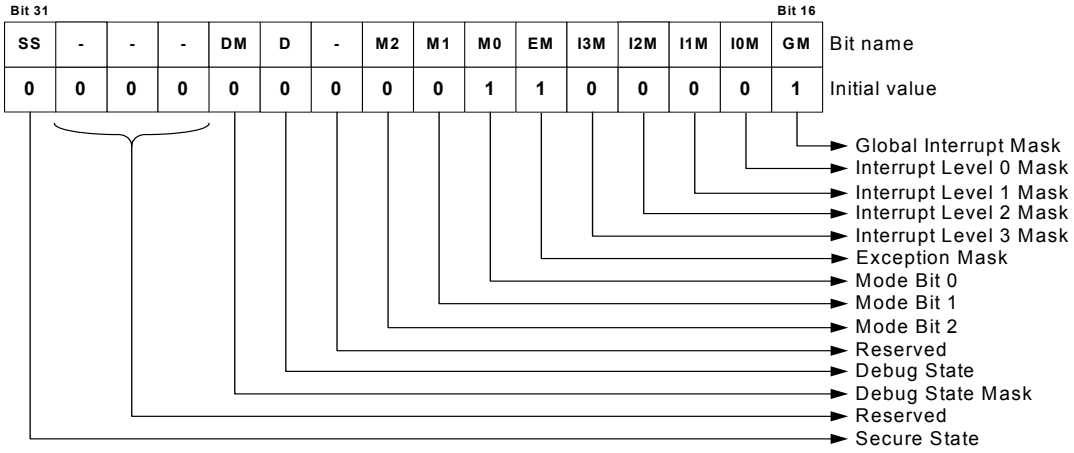
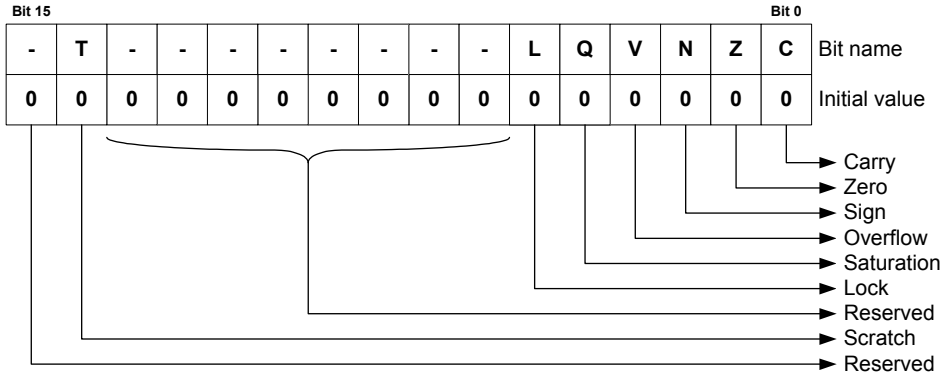


Figure 4-5. The Status Register Low Halfword



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