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 AVR®32 32-Bit Microcontroller
  - 210 DMIPS throughput at 150 MHz
  - 16 KB instruction cache and 16 KB data caches
  - Memory Management Unit enabling use of operating systems
  - Single-cycle RISC instruction set including SIMD and DSP instructions
  - Java Hardware Acceleration
- Pixel Co-Processor
  - Pixel Co-Processor for video acceleration through color-space conversion (YUV<->RGB), image scaling and filtering, quarter pixel motion compensation
- Multi-hierarchy bus system
  - High-performance data transfers on separate buses for increased performance
- Data Memories
  - 32KBytes SRAM
- External Memory Interface
  - SDRAM, DataFlash™, SRAM, Multi Media Card (MMC), Secure Digital (SD),
  - Compact Flash, Smart Media, NAND Flash
- Direct Memory Access Controller
  - External Memory access without CPU intervention
- Interrupt Controller
  - Individually maskable Interrupts
  - Each interrupt request has a programmable priority and autovector address
- System Functions
  - Power and Clock Manager
  - Crystal Oscillator with Phase-Lock-Loop (PLL)
  - Watchdog Timer
  - Real-time Clock
- 6 Multifunction timer/counters
  - Three external clock inputs, I/O pins, PWM, capture and various counting capabilities
- 4 Universal Synchronous/Asynchronous Receiver/Transmitters (USART)
  - 115.2 kbps IrDA Modulation and Demodulation
  - Hardware and software handshaking
- 3 Synchronous Serial Protocol controllers
  - Supports I2S, SPI and generic frame-based protocols
- Two-Wire Interface
  - Sequential Read/Write Operations, Philips' I^2C© compliant
- Liquid Crystal Display (LCD) interface
  - Supports TFT displays
  - Configurable pixel resolution supporting QCIF/QVGA/VGA/SVGA configurations.
- Image Sensor Interface
  - 12-bit Data Interface for CMOS cameras
- Universal Serial Bus (USB) 2.0 High Speed (480 Mbps) Device
  - On-chip Transceivers with physical interface
- 2 Ethernet MAC 10/100 Mbps interfaces
  - 802.3 Ethernet Media Access Controller
  - Supports Media Independent Interface (MII) and Reduced MII (RMII)
- 16-bit stereo audio bitstream DAC
  - Sample rates up to 50 kHz
- On-Chip Debug System
  - Nexus Class 3
  - Full speed, non-intrusive data and program trace
  - Runtime control and JTAG interface
- Package/Pins
  - AT32AP7000: 256-ball CTBGA 1.0 mm pitch/160 GPIO pins
- Power supplies
  - 1.65V to1.95V VDDCORE
  - 3.0V to 3.6V VDDIO

AVR®32 32-bit Microcontroller

AT32AP7000

Preliminary

32003M-AVR32-09/09

1. Part Description

The AT32AP7000 is a complete System-on-chip application processor with an AVR32 RISC processor achieving 210 DMIPS running at 150 MHz. AVR32 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 application performance.

AT32AP7000 implements a Memory Management Unit (MMU) and a flexible interrupt controller supporting modern operating systems and real-time operating systems. The processor also includes a rich set of DSP and SIMD instructions, specially designed for multimedia and telecom applications.

AT32AP7000 incorporates SRAM memories on-chip for fast and secure access. For applications requiring additional memory, external 16-bit SRAM is accessible. Additionally, an SDRAM controller provides off-chip volatile memory access as well as controllers for all industry standard off-chip non-volatile memories, like Compact Flash, MultiMedia Card (MMC), Secure Digital (SD)-card, SmartCard, NAND Flash and Atmel DataFlash™.

The Direct Memory Access controller for all the serial peripherals enables data transfer between memories without processor intervention. This reduces the processor overhead when transferring continuous and large data streams between modules in the MCU.

The Timer/Counters includes three identical 16-bit timer/counter channels. Each channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation.

AT32AP7000 also features an onboard LCD Controller, supporting single and double scan monochrome and color passive STN LCD modules and single scan active TFT LCD modules. On monochrome STN displays, up to 16 gray shades are supported using a time-based dithering algorithm and Frame Rate Control (FRC) method. This method is also used in color STN displays to generate up to 4096 colors.

The LCD Controller is programmable for supporting resolutions up to 2048 x 2048 with a pixel depth from 1 to 24 bits per pixel.

A pixel co-processor provides color space conversions for images and video, in addition to a wide variety of hardware filter support

The media-independent interface (MII) and reduced MII (RMII) 10/100 Ethernet MAC modules provides on-chip solutions for network-connected devices.

Synchronous Serial Controllers provide easy access to serial communication protocols, audio standards like I2S and frame-based protocols.

The Java hardware acceleration implementation in AVR32 allows for a very high-speed Java byte-code execution. AVR32 implements Java instructions in hardware, reusing the existing RISC data path, which allows for a near-zero hardware overhead and cost with a very high performance.

The Image Sensor Interface supports cameras with up to 12-bit data buses.

PS2 connectivity is provided for standard input devices like mice and keyboards.

AT32AP7000 integrates a class 3 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 C-compiler is closely linked to the architecture and is able to utilize code optimization features, both for size and speed.

2. Signals Description

The following table gives details on the signal name classified by peripheral. The pinout multiplexing of these signals is given in the Peripheral Muxing table in the Peripherals chapter.

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
Power				
AVDDPLL	PLL Power Supply	Power		1.65 to 1.95 V
AVDDUSB	USB Power Supply	Power		1.65 to 1.95 V
AVDDOSC	Oscillator Power Supply	Power		1.65 to 1.95 V
VDDCORE	Core Power Supply	Power		1.65 to 1.95 V
VDDIO	I/O Power Supply	Power		3.0 to 3.6V
AGNDPLL	PLL Ground	Ground		
AGNDUSB	USB Ground	Ground		
AGNDOSC	Oscillator Ground	Ground		
GND	Ground	Ground		
Clocks, Oscillators, and PLL's				
XIN0, XIN1, XIN32	Crystal 0, 1, 32 Input	Analog		
XOUT0, XOUT1, XOUT32	Crystal 0, 1, 32 Output	Analog		
PLL0, PLL1	PLL 0,1 Filter Pin	Analog		
JTAG				
TCK	Test Clock	Input		
TDI	Test Data In	Input		
TDO	Test Data Out	Output		
TMS	Test Mode Select	Input		
TRST_N	Test Reset	Input	Low	
Auxiliary Port - AUX				
MCKO	Trace Data Output Clock	Output		
MDO0 - MDO5	Trace Data Output	Output		
MSEO0 - MSEO1	Trace Frame Control	Output		
EVTI_N	Event In	Input	Low	

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
EVTO_N	Event Out	Output	Low	
Power Manager - PM				
GCLK0 - GCLK4	Generic Clock Pins	Output		
OSCEN_N	Oscillator Enable	Input	Low	
RESET_N	Reset Pin	Input	Low	
WAKE_N	Wake Pin	Input	Low	
External Interrupt Controller - EIC				
EXTINT0 - EXTINT3	External Interrupt Pins	Input		
NMI_N	Non-Maskable Interrupt Pin	Input	Low	
AC97 Controller - AC97C				
SCLK	AC97 Clock Signal	Input		
SDI	AC97 Receive Signal	Output		
SDO	AC97 Transmit Signal	Output		
SYNC	AC97 Frame Synchronization Signal	Input		
Audio Bitstream DAC - ABDAC				
DATA0 - DATA1	D/A Data Out	Output		
DATAN0 - DATAN1	D/A Inverted Data Out	Output		
Ethernet MAC - MACB0, MACB1				
COL	Collision Detect	Input		
CRS	Carrier Sense and Data Valid	Input		
MDC	Management Data Clock	Output		
MDIO	Management Data Input/Output	I/O		
RXD0 - RXD3	Receive Data	Input		
RX_CLK	Receive Clock	Input		
RX_DV	Receive Data Valid	Input		
RX_ER	Receive Coding Error	Input		
SPEED	Speed	Output		
TXD0 - TXD3	Transmit Data	Output		

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
TX_CLK	Transmit Clock or Reference Clock	Input		
TX_EN	Transmit Enable	Output		
TX_ER	Transmit Coding Error	Output		
External Bus Interface - EBI				
PX0 - PX53	I/O Controlled by EBI	I/O		
ADDR0 - ADDR25	Address Bus	Output		
CAS	Column Signal	Output	Low	
CFCE1	Compact Flash 1 Chip Enable	Output	Low	
CFCE2	Compact Flash 2 Chip Enable	Output	Low	
CFRNW	Compact Flash Read Not Write	Output		
DATA0 - DATA31	Data Bus	I/O		
NANDOE	NAND Flash Output Enable	Output	Low	
NANDWE	NAND Flash Write Enable	Output	Low	
NCS0 - NCS5	Chip Select	Output	Low	
NRD	Read Signal	Output	Low	
NWAIT	External Wait Signal	Input	Low	
NWE0	Write Enable 0	Output	Low	
NWE1	Write Enable 1	Output	Low	
NWE3	Write Enable 3	Output	Low	
RAS	Row Signal	Output	Low	
SDA10	SDRAM Address 10 Line	Output		
SDCK	SDRAM Clock	Output		
SDCKE	SDRAM Clock Enable	Output		
SDWE	SDRAM Write Enable	Output	Low	
Image Sensor Interface - ISI				
DATA0 - DATA11	Image Sensor Data	Input		
HSYNC	Horizontal Synchronization	Input		
PCLK	Image Sensor Data Clock	Input		

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
VSYNC	Vertical Synchronization	Input		
LCD Controller - LCDC				
CC	LCD Contrast Control	Output		
DATA0 - DATA23	LCD Data Bus	Input		
DVAL	LCD Data Valid	Output		
GPL0 - GPL7	LCD General Purpose Lines	Output		
HSYNC	LCD Horizontal Synchronization	Output		
MODE	LCD Mode	Output		
PCLK	LCD Clock	Output		
PWR	LCD Power	Output		
VSYNC	LCD Vertical Synchronization	Output		
MultiMedia Card Interface - MCI				
CLK	Multimedia Card Clock	Output		
CMD0 - CMD1	Multimedia Card Command	I/O		
DATA0 - DATA7	Multimedia Card Data	I/O		
Parallel Input/Output - PIOA, PIOB, PIOC, PIOD, PIOE				
PA0 - PA31	Parallel I/O Controller PIOA	I/O		
PB0 - PB30	Parallel I/O Controller PIOB	I/O		
PC0 - PC31	Parallel I/O Controller PIOC	I/O		
PD0 - PD17	Parallel I/O Controller PIOD	I/O		
PE0 - PE26	Parallel I/O Controller PIOE	I/O		
PS2 Interface - PSIF				
CLOCK0 - CLOCK1	PS2 Clock	Input		
DATA0 - DATA1	PS2 Data	I/O		
Serial Peripheral Interface - SPI0, SPI1				
MISO	Master In Slave Out	I/O		
MOSI	Master Out Slave In	I/O		
NPCS0 - NPCS3	SPI Peripheral Chip Select	I/O	Low	

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
SCK	Clock	Output		
Synchronous Serial Controller - SSC0, SSC1, SSC2				
RX_CLOCK	SSC Receive Clock	I/O		
RX_DATA	SSC Receive Data	Input		
RX_FRAME_SYNC	SSC Receive Frame Sync	I/O		
TX_CLOCK	SSC Transmit Clock	I/O		
TX_DATA	SSC Transmit Data	Output		
TX_FRAME_SYNC	SSC Transmit Frame Sync	I/O		
DMA Controller - DMACA				
DMARQ0 - DMARQ3	DMA Requests	Input		
Timer/Counter - TIMER0, TIMER1				
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 - TWI				
SCL	Serial Clock	I/O		
SDA	Serial Data	I/O		
Universal Synchronous Asynchronous Receiver Transmitter - USART0, USART1, USART2, USART3				
CLK	Clock	I/O		
CTS	Clear To Send	Input		
RTS	Request To Send	Output		
RXD	Receive Data	Input		

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level	Comments
TXD	Transmit Data	Output		
Pulse Width Modulator - PWM				
PWM0 - PWM3	PWM Output Pins	Output		
USB Interface - USBA				
HSDM	High Speed USB Interface Data -	Analog		
FSDM	Full Speed USB Interface Data -	Analog		
HSDP	High Speed USB Interface Data +	Analog		
FSDP	Full Speed USB Interface Data +	Analog		
VBG	USB bandgap	Analog		Connected to a 6810 Ohm ± 0.5% resistor to gound and a 10 pF capacitor to ground.

3. Power Considerations

3.1 Power Supplies

The AT32AP7000 has several types of power supply pins:

- **VDDCORE pins: Power the core, memories, and peripherals. Voltage is 1.8V nominal.**
- **VDDIO pins: Power I/O lines. Voltage is 3.3V nominal.**
- **VDDPLL pin: Powers the PLL. Voltage is 1.8V nominal.**
- **VDDUSB pin: Powers the USB. Voltage is 1.8V nominal.**
- **VDDOSC pin: Powers the oscillators. Voltage is 1.8V nominal.**

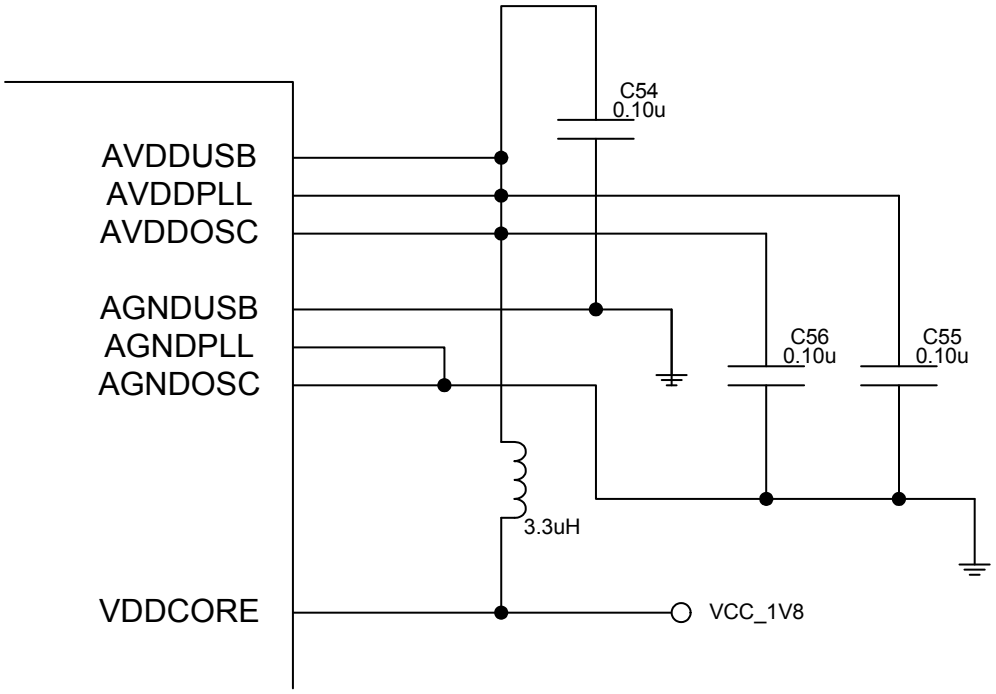
The ground pins GND are common to VDDCORE and VDDIO. The ground pin for VDDPLL is GNDPLL, and the GND pin for VDDOSC is GNDOSC.

See ”Electrical Characteristics” on page 930 for power consumption on the various supply pins.

3.2 Power Supply Connections

Special considerations should be made when connecting the power and ground pins on a PCB. Figure 3-1 shows how this should be done.

Figure 3-1. Connecting analog power supplies



3.3 Package and PinoutAVR32AP7000

Figure 3-2. 256 CTBGA Pinout

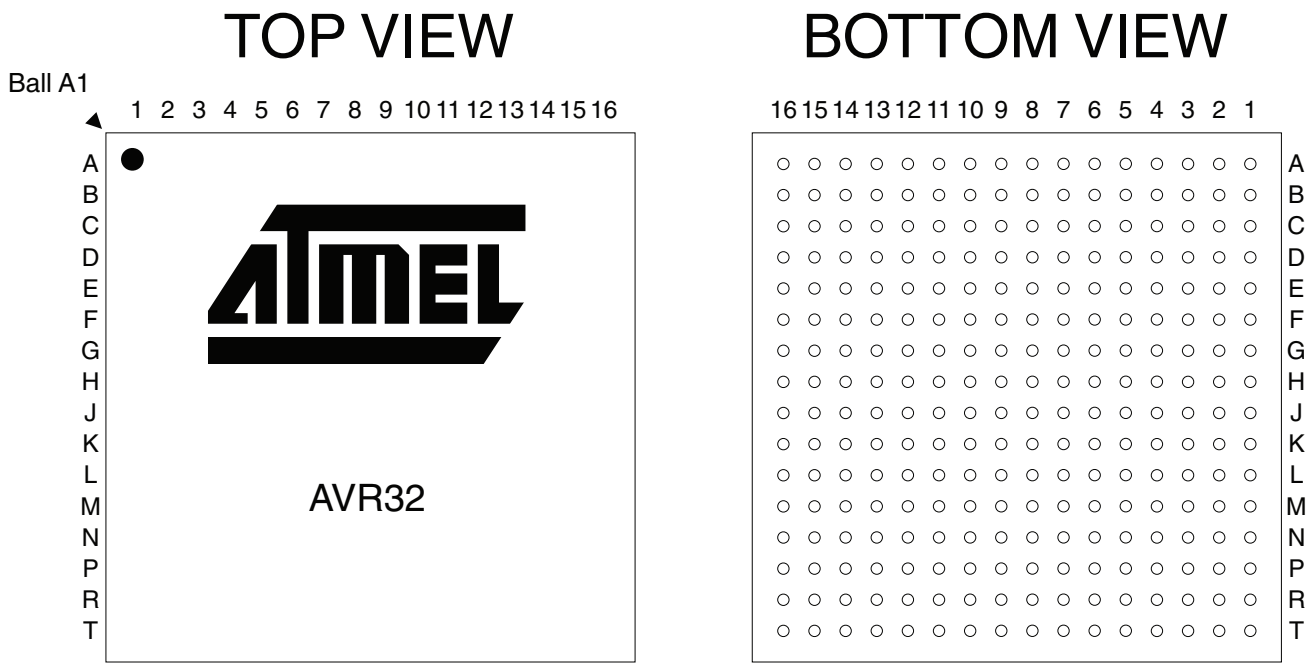


Table 3-1. CTBGA256 Package Pinout A1..T8

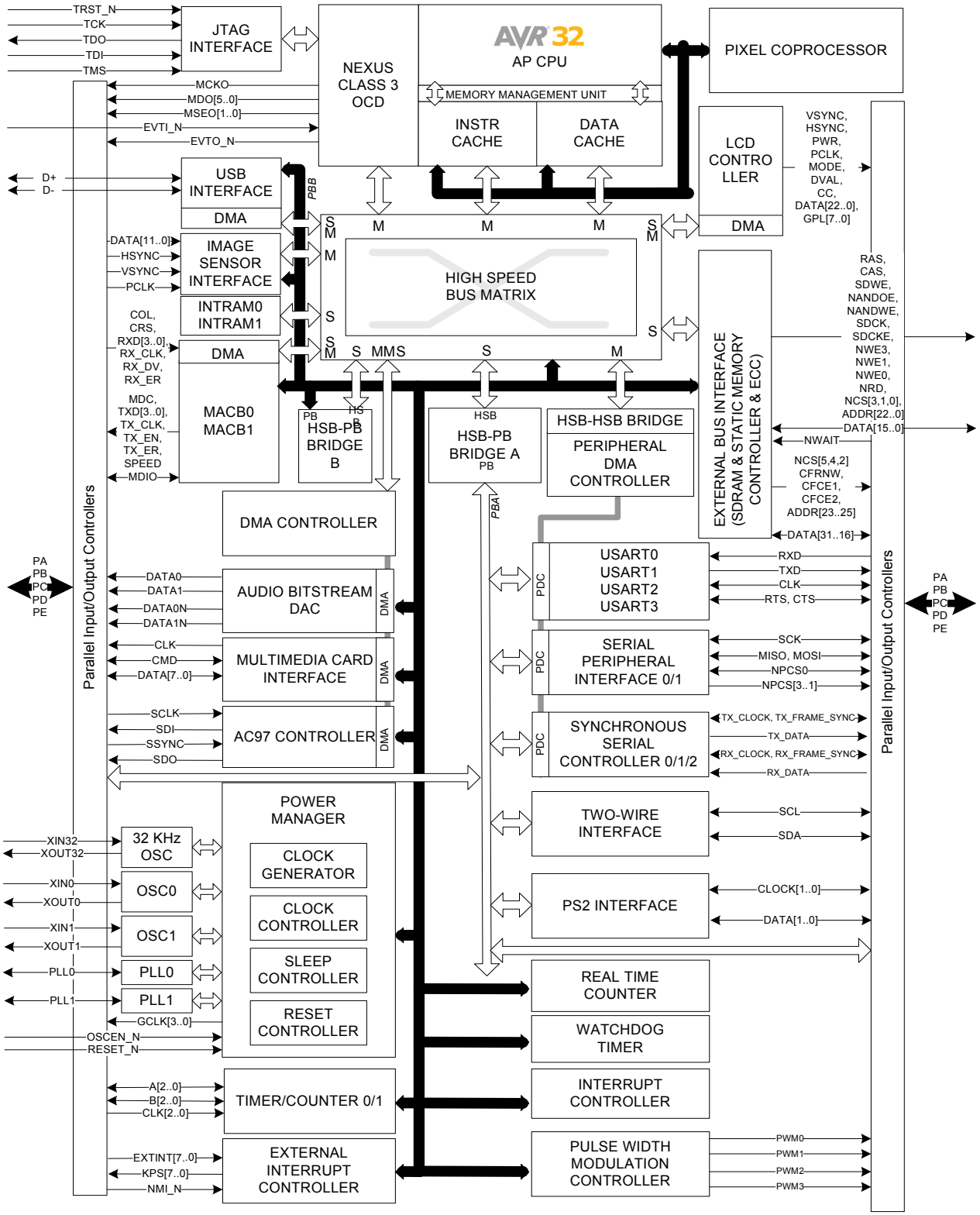
	1	2	3	4	5	6	7	8
A	VDDIO	PE15	PE13	PE11	PE07	PE02	AGNDPLL	OSCEN_N
B	GNDIO	PE16	PE12	PE09	PE04	PLL0	AVDDOSC	PC30
C	PD01	PD00	PE14	PE10	PE06	PE00	PLL1	PC31
D	PE17	PE18	PD02	PE08	PE03	GND	AGNDOSC	PC29
E	PX48	PX50	PX49	PX47	PE05	PE01	XOUT32	PC28
F	PX32	PX00	PX33	VDDIO	PX51	AVDDPLL	XIN0	PC27
G	PX04	VDDCORE	PX05	PX03	PX02	PX01	XOUT0	PC26
H	PD06	VDDIO	PD07	PD05	PD04	PD03	GND	XIN32
J	TRST_N	TMS	TDI	TCK	TDO	PD09	PD08	EVTI_N
K	PA05	PA01	PA02	PA00	RESET_N	PA03	PA04	HSDP
L	PA09	PB25	VDDIO	PA08	GND	PB24	AGNDUSB	VDDCORE
M	PA14	PA11	PA13	PA10	PA12	VDDIO	VDDIO	GND
N	PA18	PA16	PA17	PA15	PD14	GND	FSDM	VBG
P	PA20	PA19	PA21	PD11	PD16	XOUT1	GND	PA25
R	PA22	PD10	PA23	PD13	PD17	AVDDUSB	HSDM	PA26
T	VDDIO	GND	PA24	PD12	PD15	XIN1	FSDP	VDDIO

Table 3-2. CTBGA256 Package Pinout A9..T16

	9	10	11	12	13	14	15	16
A	PC23	PA06	PB21	PB16	PB13	PB11	GND	VDDIO
B	PC25	PC19	PB23	PB18	PB14	PB10	PC17	PC16
C	PC24	PA07	PB22	PB17	PB12	PB09	PB07	PB08
D	PC22	PC18	PB20	PB15	PB03	PB05	PB04	PB06
E	VDDIO	GND	PB19	PB00	PX46	PB01	VDDIO	PB02
F	PC21	VDDCORE	GND	PX44	PX42	PX43	PX40	PX45
G	PC20	PC15	PC14	PC10	PC11	PC13	PC12	VDDCORE
H	PC09	PC05	PC06	PE26	VDDIO	PC07	PX39	PC08
J	PB27	PX27	PX28	PX29	PX30	VDDCORE	GND	PX31
K	PA27	GND	PX22	PX23	PX24	PX26	VDDIO	PX25
L	PA28	VDDIO	PE24	PX38	PX18	PX20	PX21	PX19
M	PA29	PB28	PE20	PX08	PX34	PX36	PX37	PX35
N	PA30	PX53	PE22	PX06	PX11	PX15	PX17	PX16
P	WAKE_N	PX41	PE21	PX09	PB30	PC02	PX13	PX14
R	PA31	PX52	PE23	PX07	PB29	PC00	PC04	GND
T	PB26	PE25	PE19	PX10	PX12	PC01	PC03	VDDIO

4. Blockdiagram

Figure 4-1. Blockdiagram



4.0.1 AVR32AP CPU

- 32-bit load/store AVR32B RISC architecture.
  - Up to 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.
  - SIMD extention for media applications.
- 7 stage pipeline allows one instruction per clock cycle for most instructions.
  - Java Hardware Acceleration.
  - Byte, half-word, word and double word memory access.
  - Unaligned memory access.
  - Shadowed interrupt context for INT3 and multiple interrupt priority levels.
  - Dynamic branch prediction and return address stack for fast change-of-flow.
  - Coprocessor interface.
- Full MMU allows for operating systems with memory protection.
- 16Kbyte Instruction and 16Kbyte data caches.
  - Virtually indexed, physically tagged.
  - 4-way associative.
  - Write-through or write-back.
- Nexus Class 3 On-Chip Debug system.
  - Low-cost NanoTrace supported.

4.0.2 Pixel Coprocessor (PICO)

- Coprocessor coupled to the AVR32 CPU Core through the TCB Bus.
  - Coprocessor number one on the TCB bus.
- Three parallel Vector Multiplication Units (VMU) where each unit can:
  - Multiply three pixel components with three coefficients.
  - Add the products from the multiplications together.
  - Accumulate the result or add an offset to the sum of the products.
- Can be used for accelerating:
  - Image Color Space Conversion.
    - Configurable Conversion Coefficients.
    - Supports packed and planar input and output formats.
    - Supports subsampled input color spaces (i.e 4:2:2, 4:2:0).
  - Image filtering/scaling.
    - Configurable Filter Coefficients.
    - Throughput of one sample per cycle for a 9-tap FIR filter.
    - Can use the built-in accumulator to extend the FIR filter to more than 9-taps.
    - Can be used for bilinear/bicubic interpolations.
  - MPEG-4/H.264 Quarter Pixel Motion Compensation.
- Flexible input Pixel Selector.
  - Can operate on numerous different image storage formats.
- Flexible Output Pixel Inserter.
  - Scales and saturates the results back to 8-bit pixel values.
  - Supports packed and planar output formats.

- Configurable coefficients with flexible fixed-point representation.

4.0.3 Debug and Test system

- IEEE1149.1 compliant JTAG and boundary scan
- Direct memory access and programming capabilities through JTAG interface
- Extensive On-Chip Debug features in compliance with IEEE-ISTO 5001-2003 (Nexus 2.0) Class 3
- Auxiliary port for high-speed trace information
- Hardware support for 6 Program and 2 data breakpoints
- Unlimited number of software breakpoints supported
- Advanced Program, Data, Ownership, and Watchpoint trace supported

4.0.4 DMA Controller

- 2 HSB Master Interfaces
- 3 Channels
- Software and Hardware Handshaking Interfaces
  - 11 Hardware Handshaking Interfaces
- Memory/Non-Memory Peripherals to Memory/Non-Memory Peripherals Transfer
- Single-block DMA Transfer
- Multi-block DMA Transfer
  - Linked Lists
  - Auto-Reloading
  - Contiguous Blocks
- DMA Controller is Always the Flow Controller
- Additional Features
  - Scatter and Gather Operations
  - Channel Locking
  - Bus Locking
  - FIFO Mode
  - Pseudo Fly-by Operation

4.0.5 Peripheral DMA Controller

- Transfers from/to peripheral to/from any memory space without intervention of the processor.
- Next Pointer Support, forbids strong real-time constraints on buffer management.
- Eighteen channels
  - Two for each USART
  - Two for each Serial Synchronous Controller
  - Two for each Serial Peripheral Interface

4.0.6 Bus system

- HSB bus matrix with 10 Masters and 8 Slaves handled
  - Handles Requests from the CPU Icache, CPU Dcache, HSB bridge, HISI, USB 2.0 Controller, LCD Controller, Ethernet Controller 0, Ethernet Controller 1, DMA Controller 0, DMA Controller 1, and to internal SRAM 0, internal SRAM 1, PB A, PB B, EBI and, USB.

  - Round-Robin Arbitration (three modes supported: no default master, last accessed default master, fixed default master)
  - Burst Breaking with Slot Cycle Limit
  - One Address Decoder Provided per Master
- 2 Peripheral buses allowing each bus to run on different bus speeds.
  - PB A intended to run on low clock speeds, with peripherals connected to the PDC.
  - PB B intended to run on higher clock speeds, with peripherals connected to the DMACA.
- HSB-HSB Bridge providing a low-speed HSB bus running at the same speed as PBA
  - Allows PDC transfers between a low-speed PB bus and a bus matrix of higher clock speeds

An overview of the bus system is given in Figure 4-1 on page 13. All modules connected to the same bus use the same clock, but the clock to each module can be individually shut off by the Power Manager. The figure identifies the number of master and slave interfaces of each module connected to the HSB bus, and which DMA controller is connected to which peripheral.

5. I/O Line Considerations

5.1 JTAG pins

The TMS, TDI and TCK pins have pull-up resistors. TDO is an output, driven at up to VDDIO, and have no pull-up resistor. The TRST_N pin is used to initialize the embedded JTAG TAP Controller when asserted at a low level. It is a schmitt input and integrates permanent pull-up resistor to VDDIO, so that it can be left unconnected for normal operations.

5.2 WAKE_N pin

The WAKE_N pin is a schmitt trigger input integrating a permanent pull-up resistor to VDDIO.

5.3 RESET_N pin

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

5.4 EVTI_N pin

The EVTI_N pin is a schmitt input and integrates a non-programmable pull-up resistor to VDDIO.

5.5 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 PIO pins.

5.6 PIO pins

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

6. AVR32 AP CPU

Rev.: 1.0.0.0

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

6.1 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 half-word data types without penalty in code size and performance.

Memory load and store operations are provided for byte, half-word, word and double word data with automatic sign- or zero extension of half-word and byte data.

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.

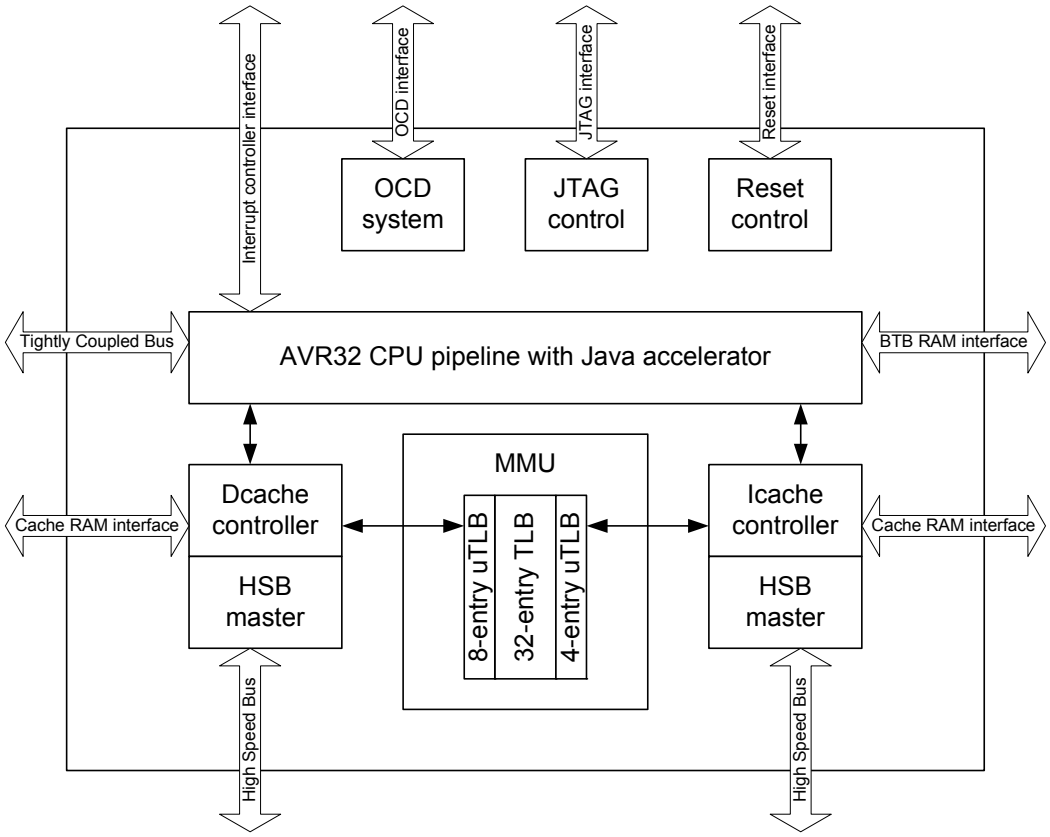
Another feature of the instruction set is that frequently used instructions, like add, have a compact format with two operands as well as an extended format with three operands. The larger format increases performance, allowing an addition and a data move in the same instruction in a single cycle. Load 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.

6.2 The AVR32 AP CPU

AVR32 AP targets high-performance applications, and provides an advanced OCD system, efficient data and instruction caches, and a full MMU. Figure 6-1 on page 19 displays the contents of AVR32 AP.

Figure 6-1. Overview of the AVR32 AP CPU

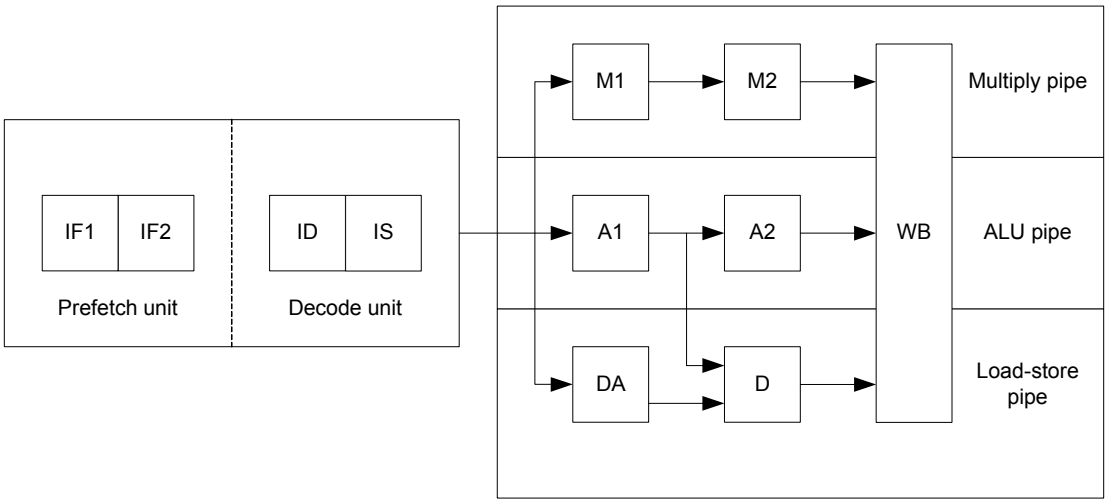


6.2.1 Pipeline Overview

AVR32 AP is a pipelined processor with seven pipeline stages. The pipeline has three subpipes, namely the Multiply pipe, the Execute pipe and the Data pipe. These pipelines may execute different instructions in parallel. Instructions are issued in order, but may complete out of order (OOO) since the subpipes may be stalled individually, and certain operations may use a subpipe for several clock cycles.

Figure 6-2 on page 20 shows an overview of the AVR32 AP pipeline stages.

Figure 6-2. The AVR32 AP Pipeline



.The follwing abbreviations are used in the figure:

- IF1, IF2 - Instruction Fetch stage 1 and 2
- ID - Instruction Decode
- IS - Instruction Issue
- A1, A2 - ALU stage 1 and 2
- M1, M2 - Multiply stage 1 and 2
- DA - Data Address calculation stage
- D - Data cache access
- WB - Writeback

6.2.2 AVR32B Microarchitecture Compliance

AVR32 AP implements an AVR32B microarchitecture. The AVR32B microarchitecture is targeted at applications where interrupt latency is important. The AVR32B therefore implements dedicated registers to hold the status register and return address for interrupts, exceptions and supervisor calls. This information does not need to be written to the stack, and latency is therefore reduced. Additionally, AVR32B allows hardware shadowing of the registers in the register file.

The *scall*, *rete* and *rets* instructions use the dedicated return status registers and return address registers in their operation. No stack accesses are performed by these instructions.

6.2.3 Java Support

AVR32 AP provides Java hardware acceleration in the form of a Java Virtual Machine hardware implementation. Refer to the *AVR32 Java Technical Reference Manual* for details.

6.2.4 Memory management

AVR32 AP implements a full MMU as specified by the AVR32 architecture. The page sizes provided are 1K, 4K, 64K and 1M. A 32-entry fully-associative common TLB is implemented, as well as a 4-entry micro-ITLB and 8-entry micro-DTLB. Instruction and data accesses perform lookups in the micro-TLBs. If the access misses in the micro-TLBs, an access in the common TLB is performed. If this access misses, a page miss exception is issued.

6.2.5 Caches and write buffer

AVR32 AP implements 16K data and 16K instruction caches. The caches are 4-way set associative. Each cache has a 32-bit System Bus master interface connecting it to the bus. The instruction cache has a 32-bit interface to the fetch pipeline stage, and the data cache has a 64-bit interface to the load-store pipeline. The caches use a least recently used allocate-on-read-miss replacement policy. The caches are virtually tagged, physically indexed, avoiding the need to flush them on task switch.

The caches provide locking on a per-line basis, allowing code and data to be permanently locked in the caches for timing-critical code. The data cache also allows prefetching of data using the *pref* instruction.

Accesses to the instruction and data caches are tagged as cacheable or uncacheable on a per-page basis by the MMU. Data cache writes are tagged as write-through or writeback on a per-page basis by the MMU.

The data cache has a 32-byte combining write buffer, to avoid stalling the CPU when writing to external memory. Writes are tagged as bufferable or unbufferable on a per-page basis by the MMU. Bufferable writes to sequential addresses are placed in the buffer, allowing for example a sequence of byte writes from the CPU to be combined into word transfers on the bus. A *sync* instruction is provided to explicitly flush the write buffer.

6.2.6 Unaligned reference handling

AVR32 AP has hardware support for performing unaligned memory accesses. This will reduce the memory footprint needed by some applications, as well as speed up other applications operating on unaligned data.

AVR32 AP is able to perform certain word-sized load and store instructions of any alignment, and word-aligned *st.d* and *ld.d*. Any other unaligned memory access will cause an MMU address exception. All coprocessor memory access instructions require word-aligned pointers. Double-word-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. Accessing an unaligned address may require several clock cycles, refer to the *AVR32 AP Technical Reference Manual* for details.

Table 6-1. Instructions with unaligned reference support

Instruction	Supported alignment
ld.w	Any
st.w	Any
lddsp	Any
lddpc	Any
stdsp	Any
ld.d	Word
st.d	Word
All coprocessor memory access instruction	Word

6.2.7 Unimplemented instructions

The following instructions are unimplemented in AVR32 AP, and will cause an Unimplemented Instruction Exception if executed:

- mems
- memc
- memt

6.2.8 Exceptions and Interrupts

AVR32 AP incorporates a powerful exception handling scheme. The different exception sources, like Illegal Op-code and external interrupt requests, have different priority levels, ensuring a well-defined behavior when multiple exceptions are received simultaneously. Additionally, pending exceptions of a higher priority class may preempt handling of ongoing exceptions of a lower priority class. Each priority class has dedicated registers to keep the return address and status register thereby removing the need to perform time-consuming memory operations to save this information.

There are four levels of external interrupt requests, all executing in their own context. The INT3 context provides dedicated shadow registers ensuring low latency for these interrupts. An interrupt controller does the priority handling of the external interrupts and provides the autovector offset to the CPU.

The addresses and priority of simultaneous events are shown in Table 6-2 on page 23.

Table 6-2. Priority and handler addresses for events

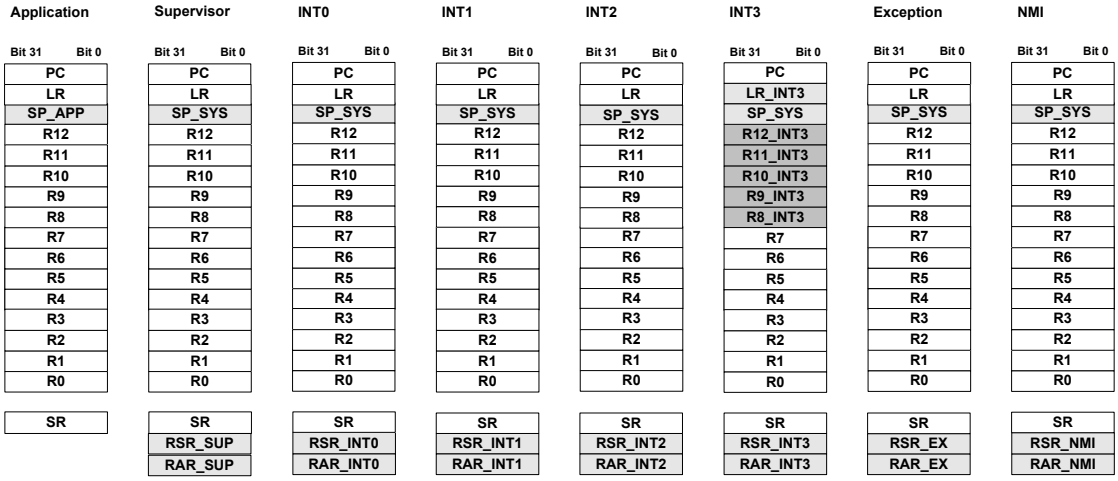
Priority	Handler Address	Name	Event source	Stored Return Address
1	0xA000_0000	Reset	External input	Undefined
2	Provided by OCD system	OCD Stop CPU	OCD system	First non-completed instruction
3	EVBA+0x00	Unrecoverable exception	Internal	PC of offending instruction
4	EVBA+0x04	TLB multiple hit	Internal signal	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	ITLB	PC of offending instruction
13	EVBA+0x50	ITLB Miss	ITLB	PC of offending instruction
14	EVBA+0x18	ITLB Protection	ITLB	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	FP Hardware	PC of offending instruction
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)	DTLB	PC of offending instruction
23	EVBA+0x38	Data Address (Write)	DTLB	PC of offending instruction
24	EVBA+0x60	DTLB Miss (Read)	DTLB	PC of offending instruction
25	EVBA+0x70	DTLB Miss (Write)	DTLB	PC of offending instruction
26	EVBA+0x3C	DTLB Protection (Read)	DTLB	PC of offending instruction
27	EVBA+0x40	DTLB Protection (Write)	DTLB	PC of offending instruction
28	EVBA+0x44	DTLB Modified	DTLB	PC of offending instruction

6.3 Programming Model

6.3.1 Register file configuration

The AVR32B architecture specifies that the exception contexts may have a different number of shadowed registers in different implementations. Figure 6-3 on page 24 shows the model used in AVR32 AP.

Figure 6-3. The AVR32 AP Register File



6.3.2 Status register configuration

The Status Register (SR) is splitted into two halfwords, one upper and one lower, see Figure 6-4 on page 24 and Figure 6-5 on page 25. 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 6-4. The Status Register High Halfword

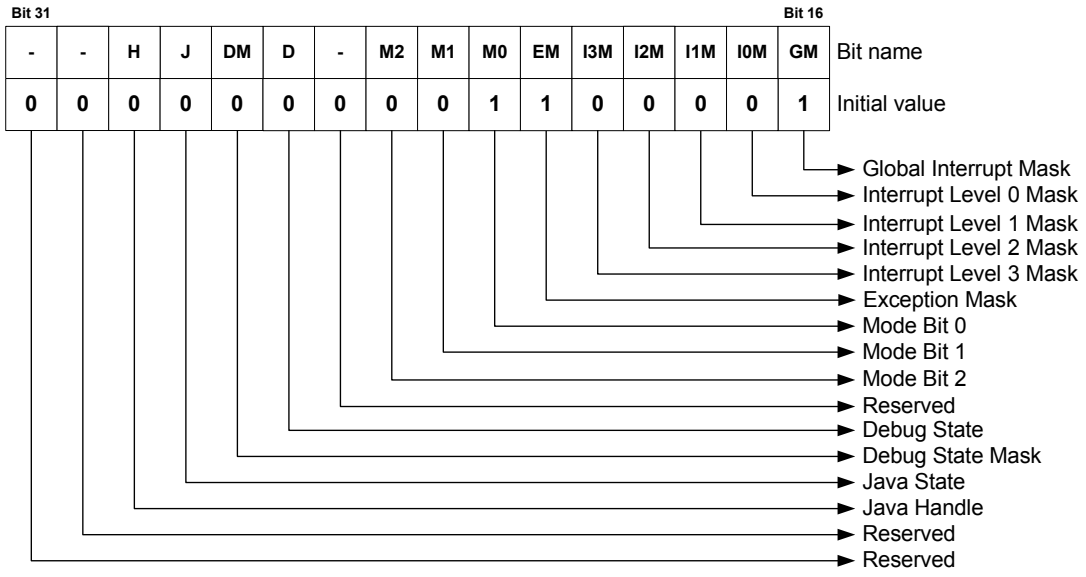
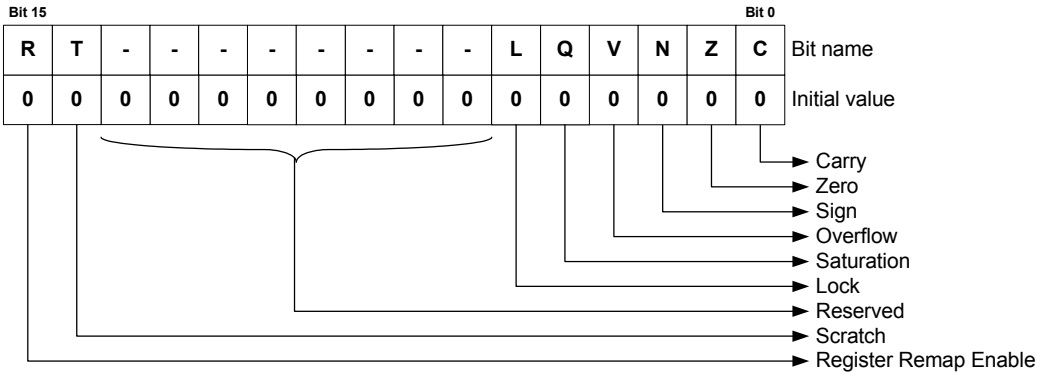


Figure 6-5. The Status Register Low Halfword



6.3.3 Processor States

6.3.3.1 Normal RISC State

The AVR32 processor supports several different execution contexts as shown in Table 6-3 on page 25.

Table 6-3. 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.

6.3.3.2 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.