EnCore™ M3 Embedded Processor Reference Manual

P/N 5001663A Revision A

Notice Page

NOTICE

DISCLAIMER

Ampro Computers, Incorporated makes no representations or warranties with respect to the contents of this manual or of the associated Ampro products, and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Ampro shall under no circumstances be liable for incidental or consequential damages or related expenses resulting from the use of this product, even if it has been notified of the possibility of such damages. Ampro reserves the right to revise this publication from time to time without obligation to notify any person of such revisions. If errors are found, please contact Ampro at the address listed below on the Notice page of this document.

TRADEMARKS

The Ampro logo is a registered trademark, and Ampro, CoreModule, EnCore, Little Board, and MiniModule are trademarks of Ampro Computers, Inc. All other marks are the property of their respective companies.

REVISION HISTORY

Revision	Reason for Change	Date
A	Initial Release	Oct/02

Ampro Computers, Incorporated
5215 Hellyer Avenue
San Jose, CA 95138-1007
Tel. 408 360-0200
Fax 408 360-0222
www.ampro.com

Audience Assumptions

This reference manual is for the person who designs computer related equipment, including but not limited to hardware and software design and implementation of the same. Ampro Computers, Inc. assumes you are qualified in designing and implementing your hardware designs and its related software into your prototype computer equipment.

Contents

Chapter 1 About this Manual 1
Purpose of this Manual 1
Reference Material 1
Related Ampro Products 2
Chapter 2 Product Overview 5
EnCore Concept 5
Product Description 6
Module Features 7
Block Diagram 8
Major Integrated Circuits 9
Connectors, Jumpers, and LED Definitions 10
Module Connectors 10
Jumper Settings 11
LED Indicators 12
Specifications 12
Physical Specifications 12
Mechanical Specifications 13
Power Specifications 14
Environmental Specifications 14
EnCore M3 Software 14
Chapter 3 Hardware 15
Overview 15
CPU 16
Memory 17
SDRAM 17
Flash Memory 17
PCI Bus Interface (J3) 18
Input/Output Interfaces 22
IDE Interface (J1) 23
Floppy Disk, Parallel, and Serial Interfaces (J8) 25
Floppy Disk Controller 27
Parallel Port Interface 27
Serial Ports 27
Utility Interface (J9) 27
System Management Bus (SMBus) 29
Universal Serial Bus (USB) 29
Audio Controller 29
PC Speaker 29
Infrared Port (IrDA) 29
Keyboard and PS/2 Mouse Controller 30
Ethernet (LAN) Interface 30
Miscellaneous Features 33
Time of Day Clock 33
EJTAG (J12) 33
LED Signals 33
Power Interface (J4) 34

Chapter 4 Software 35
Overview 35
YAMON 35
Development Environment 35
YAMON Overview 37
Operating System Support 48
Programming Considerations 49
Interrupt Configuration 49
PCI Configuration 50
TLB Configuration 52
Floppy/Parallel Configuration 52
Processor Speed Control 57
Flash Management 58
GPIO Jumper (W5) 60
Appendix A Technical Support 61
Appendix B Special Interface Cables 63
Index 65

List of Figures

Figure 2-1. Typical Product Design Flow 6
Figure 2-2. EnCore M3 Block Diagram 8
Figure 2-3. EnCore Module (top view) 9
Figure 2-4. EnCore Module (bottom view) 10
Figure 2-5. Jumpers, Connectors, and LEDs (top view) 11
Figure 2-6. Mechanical Dimensions (thorough board view) 13
Figure 4-1. Typical Host/Target Development Environment 35
Figure 4-2. Interrupt Structure 49
Figure B-1. Ethernet Port 2 Interface Cable 63
Figure B-2. Serial Debug Port Cable 63

List of Tables

Table 2-1. Major Integrated Circuits 9
Table 2-2. Connector Summary 10
Table 2-3. User-Definable GPIO Jumper Settings (W5) 11
Table 2-4. GPIO LED Indicators 12
Table 2-5. Ethernet Port #2 (J14) LED Indicators 12
Table 2-6. Physical Specifications 12
Table 2-7. Power Requirements 14
Table 2-8. Environmental Specifications 14
Table 3-1. PCI Bus Interconnect Pins and Signals (J3) 18
Table 3-2. I/O Interface to Chip and I/O Connector 22
Table 3-3. IDE Interface Pins and Signals (J1) 23
Table 3-4. Floppy/Parallel/Serial Interface Pins and Signals (J8) 25
Table 3-5. Utility Interface Pins and Signals (J9) 28
Table 3-6. Ethernet Interface Pins and Signals (J7) 31
Table 3-7. Ethernet Port 2 Pins and Signals (J14) 32
Table 3-8. EJTAG Interface Pins and Signals (J12) 33
Table 3-9. LED Signal Descriptions 34

Table 3-10. Power Interface Pins and Signals (J4) 34
Table 4-1. System Parameter Table 39
Table 4-2. YAMON Command Descriptions 41
Table 4-3. PCI Bus Memory Map Table 50
Table 4-4. Type 0 Configuration Cycle Addresses and Descriptions 51
Table 4-5. Type 1 Configuration Cycle Addresses and Descriptions 51
Table 4-6. Flash Memory Sections and Descriptions 59
Table 4-7. Flash Memory Map 60
Table A-1. USA Technical Support Contact Information 61

Chapter 1 About this Manual

Purpose of this Manual

This manual is for designers of systems based on the EnCore™ M3 module. This manual contains information that permits designers to create an embedded system based on specific design requirements.

Information provided in this reference manual includes:

- EnCore M3 Specifications
- Environmental requirements
- Major chips and features implemented
- EnCore M3 connector/pin numbers and definition
- Supplied software and programming considerations

Information not provided in this reference manual includes:

- Detailed Chip specification
- Internal component operation
- Internal registers or signal operations
- Bus or signal timing for industry standard buses and signals

Reference Material

The following list of reference materials may be helpful for you to complete your custom design successfully. Most of this reference material is also available on the EnCore Development Kit CD-ROM or Ampro web site in the Embedded Design Resource Center. The Embedded Design Resource Center was created for embedded system developers to share Ampro's knowledge, insight, and expertise gained from years of experience.

Specifications and Manuals

- EnCore Platform Architecture Specification
- PCI Local Bus Specification Rev. 2.2,

  For latest revision of the PCI specifications, contact the PCI Special Interest Group Office at:

  PCI Special Interest Group
  P.O. Box 14070
  Portland, OR 97214
  800-433-5177 (U.S.) or 503-797-4207 (International)
- EnCore EBX Baseboard Reference Design Manual
- EnCore EBX Baseboard Drawings
  - Schematics
  - Parts list
- EnCore M3 QuickStart Guide

NOTE	Check the Ampro web site at www.ampro.com, for the latest version of these documents.

Web sites to the **chip specifications** used in the EnCore M3 module.

- AMD Alchemy Semiconductor Au1500™ Processor on the module

  Web site: http:// www.alchemysemi.com

- VIA Technologies, Inc. Southbridge, VT82C686B, used for Southbridge and Super I/O controller

  Web site: http://www.viatech.com

- AMD flash memory chip, 29F160DB, used for the flash memory on the module.

  Web site: http://www.amd.com

Related Ampro Products

The following items are directly related to successfully using the Ampro product you have just purchased or plan to purchase. Ampro highly recommends that you purchase and utilize an EnCore QuickStart Kit simultaneously with the design of your custom logic board.

EnCore M3 Support Products

- EnCore M3 QuickStart Kit (QSK)

  The QuickStart Kit includes the EnCore M3 module, at least 32MB of SODIMM SDRAM, the EnCore EBX Baseboard, and the EnCore Development Kit (EDK) CD-ROM. Included on the EDK CD-ROM are product manuals, reference items for the EBX baseboard, such as schematics and BOMs, and software development tools, board support packages (BSP) and utilities.

- EnCore EBX Baseboard

  The EnCore EBX baseboard, included with EnCore M3 QuickStart Kit, can be utilized as a substitute for your custom design. The Ampro baseboard provides a "gold board" to compare your custom design against as well as a convenient vehicle for development and test of application software.

  The EnCore EBX baseboard provides all the connections necessary to test and debug the software and firmware of your baseboard design, with Ampro's CPU and support circuitry. All of the main I/O and peripheral device connections, including the keyboard, mouse, floppy drive, IDE hard drive, IDE CD-ROM, parallel port, serial ports, sound connections, and a standard ATX power supply connection are provided on the EnCore EBX baseboard.

- EnCore Development Kit (EDK)

  This is the basic support package provided with the EnCore M3 module. The CD-ROM provides all of the documentation in PDF format, including this reference manual, the EnCore M3 QuickStart Guide, and the EnCore EBX Baseboard Reference Manual. This CD-ROM also provides reference material for the EnCore EBX Baseboard, such as OrCad schematics and Allegro (by Cadence Systems) layout files, a bill of materials (BOM), and an Approved Vendor List (AVL). Also located on the CD-ROM is a development environment, including the monitor and Board Support Packages (BSP) for the respective operating systems, along with the required start-up code and drivers for each peripheral device on the EnCore M3 module.

Other Ampro EnCore Products

- EnCore™ PP1 – This embedded processor module is a low-power, high-performance module using a PowerPC™ based processor from Motorola. The typical power consumption of an EnCore PP1 module is less than 4.0 watts at 300MHz. In addition to the standard EnCore features, the EnCore PP1 has a third serial port and drives the PCI Bus at 33 or 66MHz.

- EnCore™ 400 – This embedded processor module is a low-cost, high-integration 486-based CPU and is based on STMicroelectronics' 133MHz STPC® Atlas™ processor. In addition to the standard EnCore features, the EnCore 400 includes a 2D graphics controller, which provides both CRT and TFT flat panel video interfaces.

- EnCore™ 500 – This embedded processor module uses a 266MHz Mobile Pentium processor for its computing functions. In addition to the standard EnCore features, the EnCore 500 includes high performance 3D graphics for CRTs and popular LCD panels.
- EnCore™ 700 – This embedded processor module is the highest performance EnCore product offered with a high performance Pentium® III processor and the Intel 815EM chipset. In addition to the standard EnCore features, the EnCore 700 provides a 2D/3D graphics controller, which includes both CRT and TFT flat panel video interfaces.

Other Ampro Products

- Little Board™ Family – These high-performance, highly integrated single-board computers use the EBX form factor (5.75 x 8.00 inches), and are available with 486, Mobile Pentium and Pentium II processors. The EBX-compliant Little Board single-board computers offer functions equivalent to a complete laptop or desktop PC system, plus several expansion cards. Built-in extras to meet the critical requirements of embedded applications include onboard solid state disk compatibility, watchdog timer, smart power monitor, and other embedded-PC BIOS enhancements.
- CoreModule™ Family – These complete embedded-PC subsystems on single PC/104 or PC/104-Plus form-factor (3.6 x 3.8 inches) modules feature 386SX, 486DX, and Mobile Pentium CPUs. Each CoreModule includes a full complement of PC core logic functions, plus disk controllers, and serial and parallel ports. Some modules include CRT and flat panel graphics controllers or an Ethernet interface. The CoreModules also come with built-in extras to meet the critical reliability requirements of embedded applications. These include onboard solid state disk compatibility, watchdog timer, smart power monitor, and other embedded-PC BIOS enhancements.
- MiniModule™ Family – This extensive line of peripheral interface modules compliant with PC/104 and PC/104-Plus can be used with Ampro CoreModule and Little Board single-board computers to configure embedded system solutions. Ampro's highly reliable MiniModule products currently support CRT and flat-panel display interfacing, networking, and PC Card expansion.

Chapter 2 Product Overview

This introduction presents general information about the EnCore Concept and the EnCore M3 Embedded Processor. After reading this chapter you should understand:

- EnCore Concept
- Development strategies using the EnCore M3 Embedded Processor
- EnCore M3 architecture
- EnCore M3 module features
- Major components
- Connectors and jumpers
- Specifications

EnCore Concept

Embedded CPU technology is undergoing a period of rapid change. Many next-generation system designs require increased performance, low power consumption, heat dissipation, and Internet ready connectivity. CPU choices have grown from a simple choice between Intel and Motorola architectures to include a wide variety of new low power RISC processors.

During this period of rapid change, designers of embedded systems face increasing pressures to bring products to market quickly. Many products that once incorporated a custom CPU design can no longer afford the time to develop and debug a custom CPU let alone port operating system software to it. Furthermore, CPU subsystem design usually plays a small part in providing any uniqueness to an embedded product. The remainder of the embedded product design adds key logic elements that provide a unique product and differentiate it from other products serving the same market. The challenge is to speed these designs to market by eliminating the need for a custom CPU design while providing the flexibility to include all critical elements, which make the embedded product unique.

EnCore modules provide a standard, off-the-shelf CPU subsystem that can be included in virtually any product. EnCore modules work like a high integration chip, plugging into your circuit board to provide the custom logic for your application. EnCore provides a simple, industry standard interface that is independent of CPU type. The EnCore interface includes the industry-standard PCI bus, I/O signals from the peripheral components on the EnCore module, power, and ground. The EnCore modules support Intel architecture (x86), MIPS and PowerPC processors, and other RISC processors. Go to the Ampro web site (www.ampro.com) for the latest processor support information.

This standard EnCore interface lets you try different processors and different processor types in your actual product environment with the ability to defer a processor choice until late in the project if you choose. The interface also lets you easily offer different versions of your product with different capabilities by either selecting different EnCore modules with the same baseboard, or by designing different baseboards for the same CPU. This same capability leads to simpler ability to upgrade by either selecting a more powerful CPU (without baseboard redesign) or enhancing the baseboard without touching the CPU subsystem or the bulk of the applications software.

EnCore's flexibility enables designers to take an accelerated, low risk path with EnCore-based designs. Unlike chip based designs, the approach with EnCore allows design to begin independent of a processor decision. Your custom logic board design can begin, and be completed without identifying which processor is to be used in your application. The system can be prototyped with any available EnCore module. Your design flow diagram might look similar to the one shown in Figure 2-1.

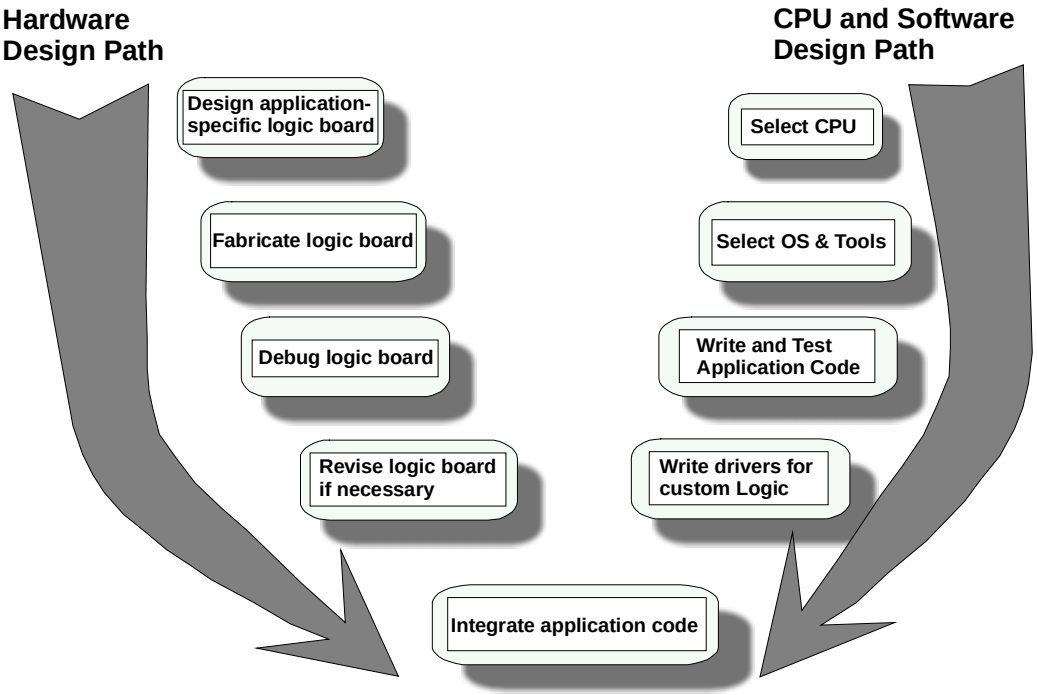

Figure 2-1. Typical Product Design Flow

Product Description

The EnCore M3 module is a low power, high-performance, high-integration MIPS32-compatible embedded processor module based upon the EnCore platform. Like all EnCore products, it is designed to mate with a custom application-specific logic board. Using a 400MHz Alchemy Au1500™ processor from AMD, rated at 480-Drystone MIPS, the typical power consumption of an entire EnCore M3 module is less than 2.5 watts.

As with all EnCore modules, the EnCore M3 provides the functionality of a complete embedded CPU subsystem in a small, 100x145mm (3.94x5.70inch) format. It supports up to 256MB SODIMM SDRAM and provides 2MB Flash, three serial ports, two USB ports, enhanced Ultra 33/66 Synchronous DMA IDE interface to two drives, floppy disk controller, PS/2 keyboard and mouse ports, IrDA port and an ECP/EPP bi-directional parallel port. It also includes two 10/100BaseT Ethernet interfaces and an AC97 audio interface.

Like all modules in the EnCore series, the EnCore M3 module is designed to interface with a host baseboard that provides application-specific logic, I/O connections, and DC power. All EnCore modules interface to the baseboard via the industry-standard PCI bus and a set of I/O signals. The small form-factor of an EnCore module gives OEM designers a great deal of flexibility in baseboard design.

EnCore modules enhance time to market for systems that seek to combine a standard 32- or 64-bit processor subsystem with applications specific logic on a custom baseboard. To speed baseboard design, Ampro offers a sample baseboard to OEM customers as a reference design for development of their own system boards. The EnCore M3 module is compatible with popular hardware and software standards assuring seamless integration with a wide range of off-the-shelf operating systems, application software and peripheral devices.

Module Features

- CPU
  - 400MHz Alchemy AU1500™MIPS32 Processor from AMD
  - Very low power; 0.50 Watts max @ 400MHz
  - Software selectable speed up to 400MHz
  - 16KB instruction cache, 16KB data cache
- Memory
  - 100MHz 3.3V SDRAM
  - Socket for one SODIMM module, 144 pin
  - 16MB to 256MB SODIMM SDRAM, 32-bit wide
  - 2MB Flash memory
- PCI Bus
  - PCI 2.2 compliant
  - 32-bit, 33/66MHz bus speed
  - PCI bus supports 4 devices on baseboard
  - Up to 3 PCI bus masters
- Input/Output Interfaces - all connections to baseboard unless stated
  - South Bridge VIA VT82C686B with Super I/O
  - IDE Interface
    - EIDE – Enhanced Ultra 33/66 Synchronous DMA IDE interface
    - Supports two IDE drives
    - Supports ATAPI extensions including DVD
  - Utility Interface
    - Serial – Two serial ports to the baseboard with full modem support and one console/debug port on the M3 module
    - Parallel/Floppy Drive– ECP/EPP bi-directional port also serves as floppy drive interface
    - USB – One USB Host controller with two ports. One port can be configures as USB slave. Supports USB v. 1.1 and Open HCI standards
    - Infrared (IrDA)
    - Mouse/Keyboard – PS/2 interface
    - Audio Interface – AC97 on module; CODEC on the baseboard
- Network Interface
  - Ethernet Port 1 – 10/100BaseT Ethernet (autosensing), interface and magnetics on the baseboard
  - Ethernet Port 2 – 10/100BaseT Ethernet (autosensing), interface and magnetics on the module
- Miscellaneous
  - Time of Day Clock – No battery back-up
  - EJTAG (MIPS Rev 2.5 connection)

- LED indicators on the module

- Power
  - +5V and +3.3V
  - Approximately 2.5 Watts @ 400MHz

Block Diagram

The following diagram shows the functional components of the module:

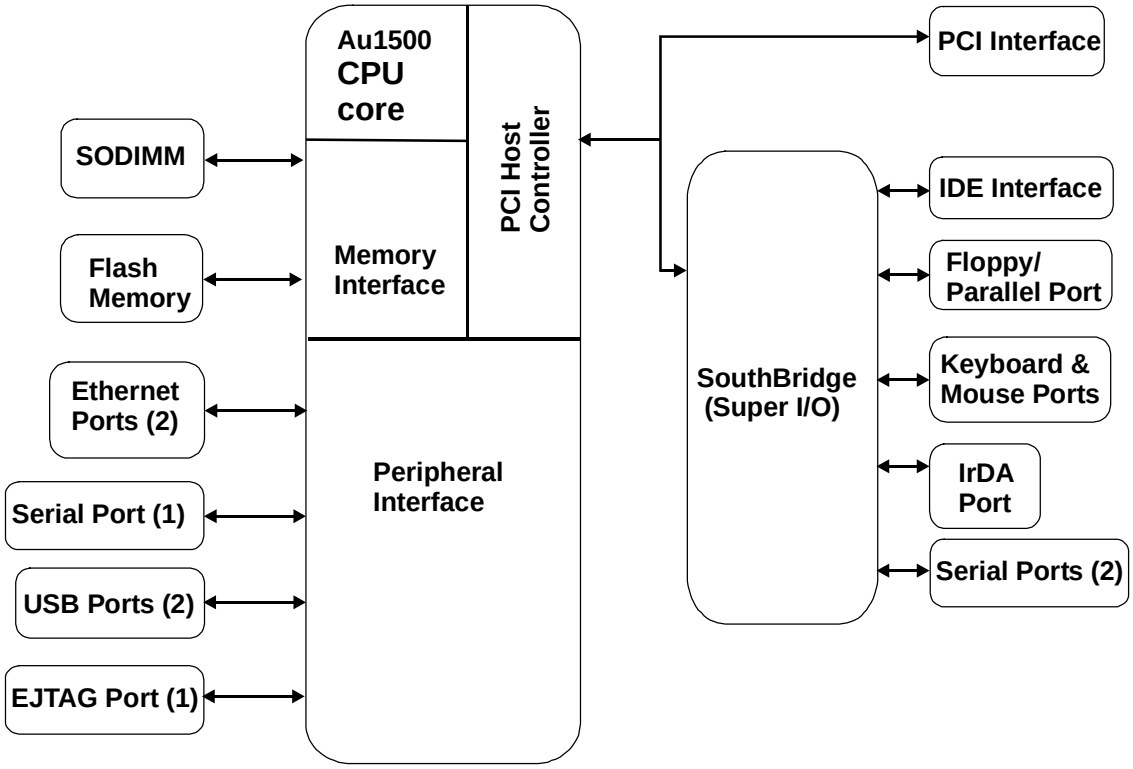

Figure 2-2. EnCore M3 Block Diagram

Major Integrated Circuits

The following tables summarize the major integrated circuits on the EnCore M3 module

Table 2-1. Major Integrated Circuits

Mfg.	Model	Description	Features
AMD-Alchemy Semiconductor	AU1500	Embedded CPU (U8) – This integrated processor combines a MIPS32 compatible microprocessor with a PCI Host Controller, high-performance memory controller, and peripheral controller.	CPU core Memory controller PCI Host Controller USB ports (2) 10/100BaseT Ethernet (2) AC97 Audio interface Serial port (1)
AMD	29F160DB	Flash memory (U2) – This chip provides the CMOS flash memory for the module.	Flash memory
VIA Technologies, Inc.	VT82C686B	Southbridge and Super I/O Controller (U13) – This chip provides all the peripheral device interfaces not provided by the CPU.	IDE Floppy disk Parallel port Keyboard Mouse Serial ports (2) IrDA

The following diagram shows the placement of the major chips on the board.

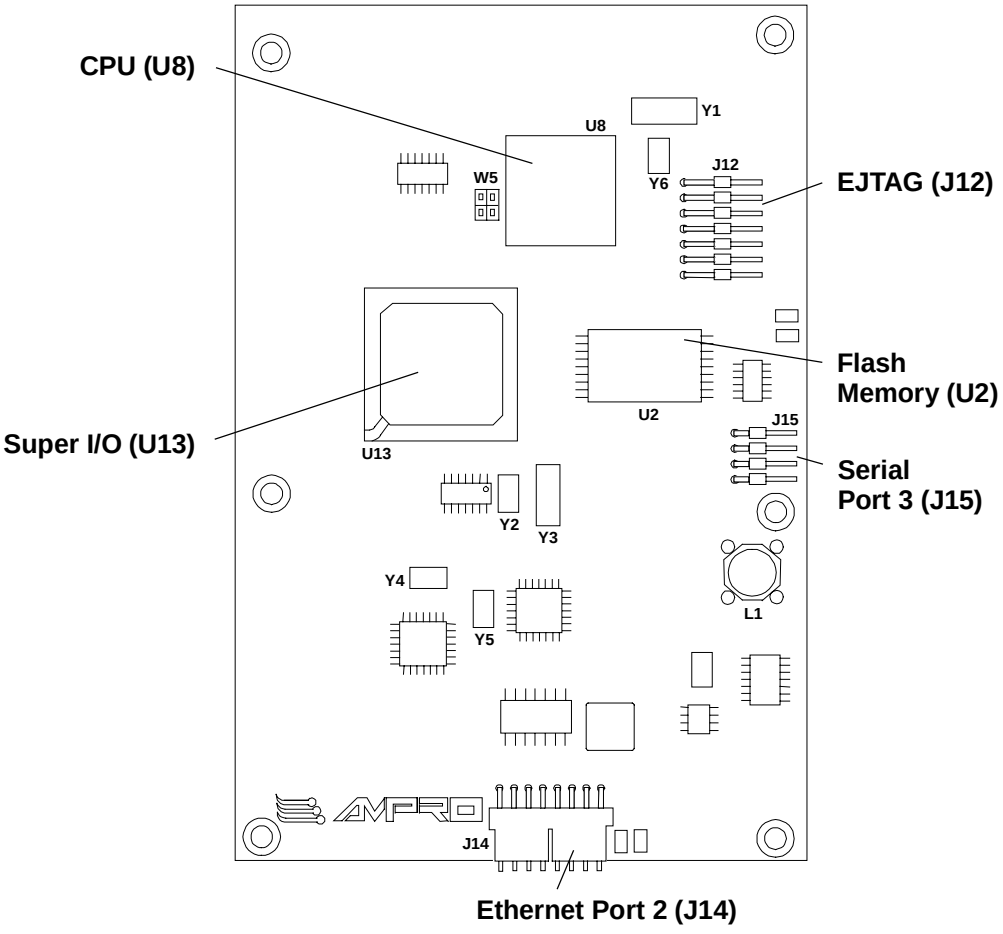

Figure 2-3. EnCore Module (top view)

Connectors, Jumpers, and LED Definitions

The following tables summarize the connectors, jumpers, and LEDs on the EnCore M3 module.

Module Connectors

Refer to Table 2-2 and Figures 2-3 and 2-4 for the connectors on the EnCore M3 module.

Table 2-2. Connector Summary

Connector	Description	Type
J1	IDE Connector	44-pin 2mm
J3	PCI Bus Connector	120-pin (30x4) 2mm
J4	Power Connector	10-pin 0.1mm
J7	Primary Ethernet Interconnect	10-pin 2mm
J8	Floppy/Parallel/Serial Connector	44-pin 2mm
J9	Utility Connector	44-pin 2mm
J12	EJTAG Connector (See Figure 2-3)	14-pin right angle connector
J14	Ethernet Port #2 (See Figure 2-3)	8-pin 2mm right angle connector
J15	Serial Port #3 (See Figure 2-3)	4-pin right angle connector

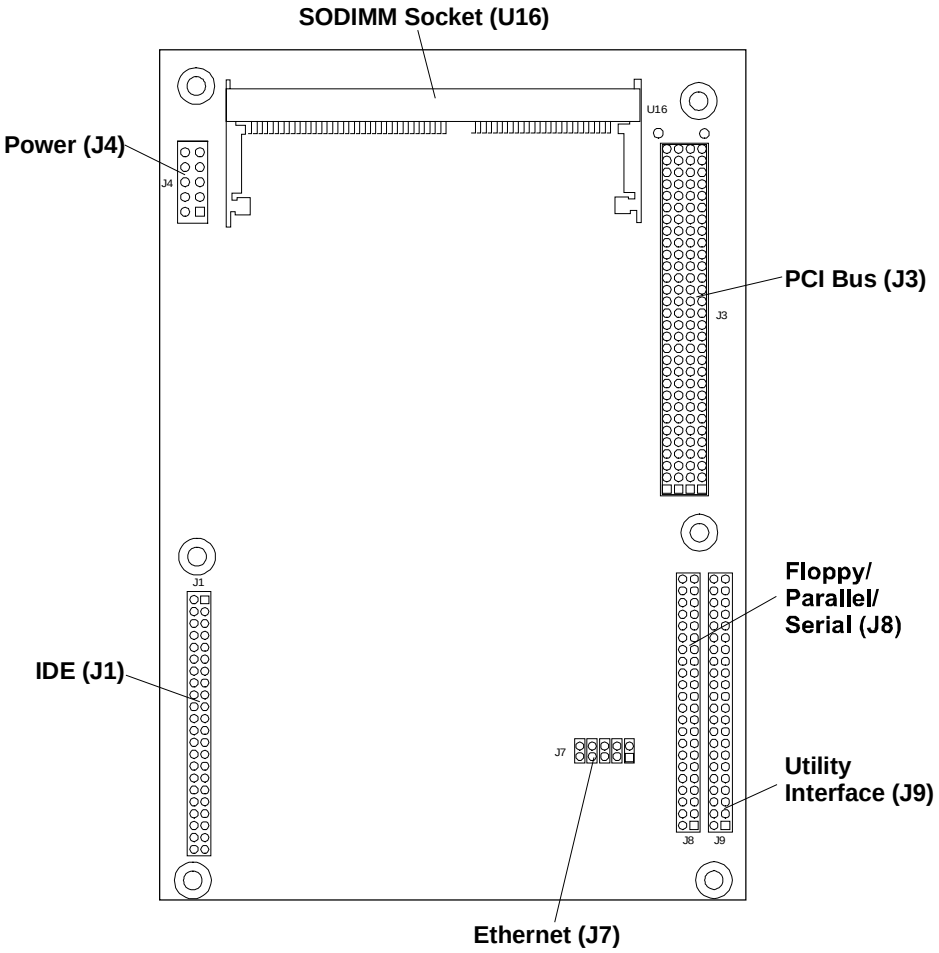

Figure 2-4. EnCore Module (bottom view)

Jumper Settings

Table 2-3 provides the W5 jumper settings for the user-definable GPIO 200 and GPIO 201 lines on the AU1500 CPU. Refer to Figure 2-5.

Table 2-3. User-Definable GPIO Jumper Settings (W5)

Signal	GPIO 201 W5 (1-2)	GPIO 200 W5 (3-4)
High	On	On
Low	Off	Off

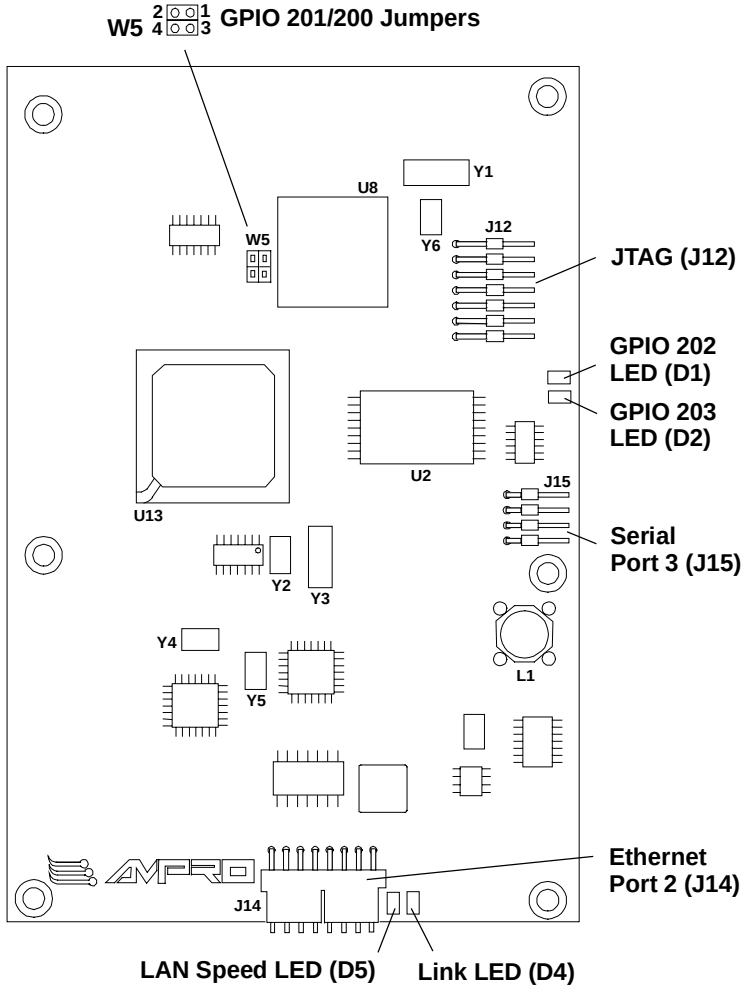

Figure 2-5. Jumpers, Connectors, and LEDs (top view)

LED Indicators

Table 2-4 provides the LED color and definitions for GPIO 202 and 203. Refer to Figure 2-5.

Table 2-4. GPIO LED Indicators

Indicator	Definition
GPIO 202 LED (D1)	This green LED indicates the state of GPIO 202. Refer to Table 3-9 in Chapter 3, *Hardware* for more information.
GPIO 203 LED (D2)	This yellow LED indicates the state of GPIO 203. Refer to Table 3-9 in Chapter 3, *Hardware* for more information.

Table 2-5 provides the LED color and definitions for the Ethernet (LAN) Port #2 located on the EnCore M3 module. The other Ethernet Port #1 is located on the baseboard with its respective LEDs. Refer to Figure 2-5.

Table 2-5. Ethernet Port #2 (J14) LED Indicators

Indicator	Definition
Link LED (D4)	This yellow LED is the activity/link indicator and provides the status of Ethernet Port #2. Refer to Table 3-9 in Chapter 3, *Hardware* for more information.
LAN Speed (D5)	This green LED is the LAN Speed indictor and indicates the transmit or receive speed of Ethernet Port #2. Refer to Table 3-9 in Chapter 3, *Hardware* for more information.

Specifications

Physical Specifications

Table 2-6. Physical Specifications

Item	Dimension
Weight	56.69 g (2 oz) without SODIMM
Height (overall above baseboard)	22.2 mm (0.875 inches) (highest component)
Width	100mm (3.94 inches)
Length	145 mm (5.71 inches)

Mechanical Specifications

The figure below shows the mounting dimensions and mechanical form factors for the EnCore M3 module. The EnCore module conforms to the EnCore physical standard to insure the widest possible design coverage for developers.

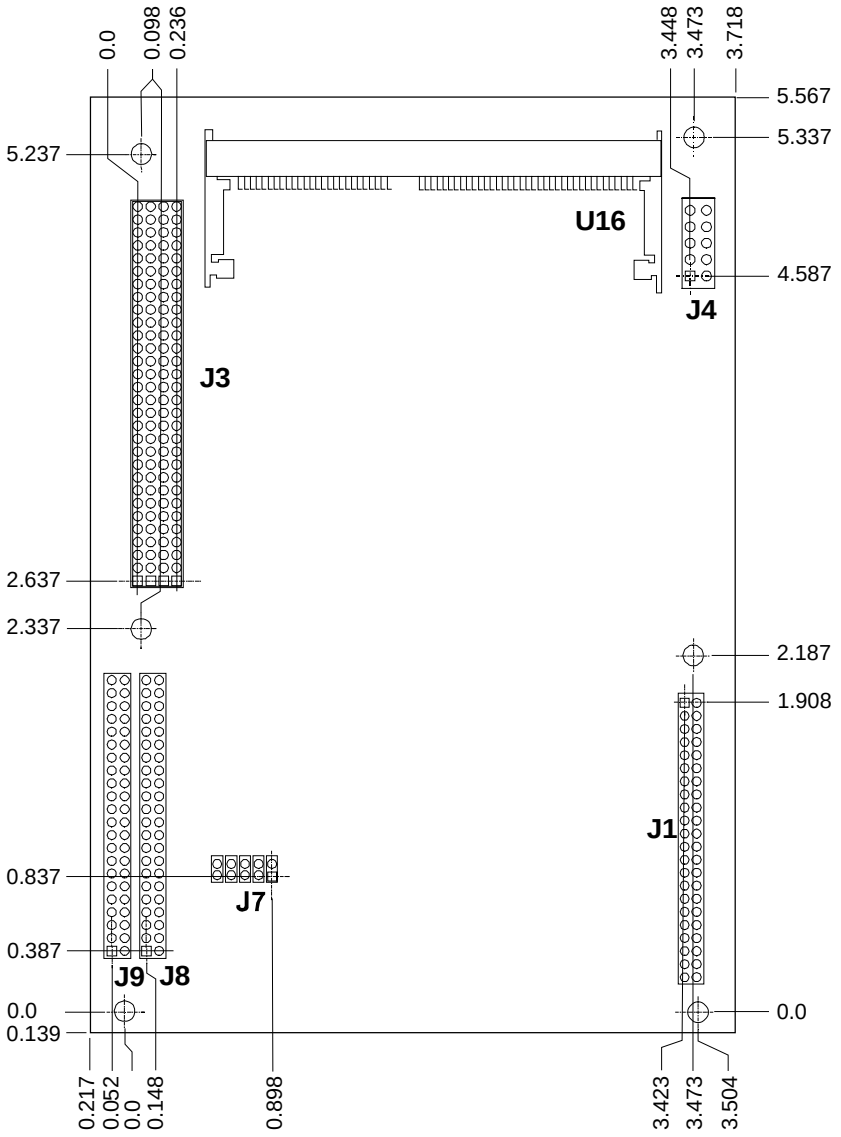

Figure 2-6. Mechanical Dimensions (thorough board view)

NOTE	All measurements are given in inches.

Power Specifications

Table 2-7 provides the power requirements from the baseboard.

Table 2-7. Power Requirements

Parameters	Characteristics
Input Type	Regulated DC voltages
Input Power Requirements	+3.3V +/- 5% @2A
	+5V +/- 5% @2A
Operating Power	2.5W Continuous

Environmental Specifications

Table 2-8 provides the most efficient operating and storage condition ranges required using this module.

Table 2-8. Environmental Specifications

Parameters	Conditions
Temperature	
Operating	0° to 70°C (32° to 158°F)
Extended (optional)	Contact the Factory
Storage	-55° to +85° C (-67° to 185° F)
Humidity	
Operating	20% to 80% relative humidity, non-condensing
Non-operating	5% to 95% relative humidity, non-condensing

EnCore M3 Software

There are several popular operating systems used pervasively in embedded applications and these have already been ported to the EnCore modules. Please check with the Ampro web site for the most up-to-date list of software available for EnCore M3 module. For more information concerning the software provided for the EnCore M3 module, refer to Chapter 4, *Software*.

If a complete support package is available for the desired operating system, the web site will indicate its availability. This would include the required start-up code and drivers for each peripheral device on the EnCore module. These support packages are available either on the EDK CD-ROM included with your EnCore QuickStart Kit, from the Ampro web site, or from the operating system manufacturer. The Ampro web site will direct you to the appropriate source for the operating system you are considering.

Using the support packages provided, you can build, load and boot the supported operating systems on your EnCore QuickStart Kit hardware. There should be no change required to load and boot the operating system on an EnCore module attached to your custom logic board. You should only need to add driver software for the unique hardware added on your custom logic board.

Chapter 3 Hardware

Overview

This chapter is divided into the following chapter headings with descriptions and tables where appropriate.

- CPU
- Memory
  - SDRAM
  - Flash Memory
- PCI Bus Interconnect (J3)
- IDE Interconnect (J1)
- Floppy/Parallel/Serial Interconnect (J8)
  - Floppy disk controller
  - Parallel port
  - Serial ports
- Utility Interconnect (J9)
  - USB (Universal Serial Bus)
  - Audio Controller
  - PC Speaker
  - Infrared (IrDA) Port (U1)
  - Keyboard and PS/2 Mouse Controller
- Ethernet Connections
  - Ethernet interconnect (J7)
  - Ethernet interface (J14)
- Miscellaneous
  - Time of Day Clock
  - EJTAG (J12)
  - LED Signals
- Power Supply Interconnect (J4)

NOTE	The EnCore M3 module uses a variety of chips to provide all of the features listed in this manual. However, not all the features available for each chip in the set have been implemented.

CPU

The Alchemy AU1500™ from AMD is a high performance, low power system-on-a-chip (SOC). The Au1500™ uses the MIPS32™ instruction set. Designed for maximum performance at very low power, the Au1500™ on the EnCore M3 operates at speeds up to 400MHz. CPU power dissipation is less than 0.50 watts @ 400MHz. Highly integrated with on-chip memory controllers and Internet access peripherals, the Au1500™ provides the ultimate high integration, high-performance, low power processor.

The Au1500™ clock speed is software configurable and is variable up to 400MHz. The details of this procedure are available under *Programming Considerations* in Chapter 4, *Software*. Reducing the clock speed lowers power consumption. The processor is fully static, meaning the clock may be halted in a sleep mode. On system reset, Ampro's YAMON monitor sets the clock to 400MHz.

The Au1500™ contains devices that are divided between two buses: the System bus and the Peripheral bus. The devices on the System bus are grouped together into the System Module. Devices on the Peripheral bus are considered as part of the Peripheral Module. The following describes the key capabilities and features of the Au1500 CPU as implemented on the EnCore M3 module.

- **High Speed MIPS32 CPU**
  - Up to 400MHz Core Speed
  - MIPS32™ Instruction Set
  - 32-Bit Architecture
  - High Speed Multiply-Accumulate (MAC) and Divide Unit
- **Memory**
  - SDRAM
  - Supported memory sizes range from 16MB to 256MB
  - 100MHz SDRAM Controller
  - Flash Memory 2MB in a single 16-bit wide flash device
  - Flash Memory Controller
- **Highly-Integrated System Peripherals**
  - Two 10/100BaseT Ethernet ports
  - Two USB ports
  - Two UARTs
  - Audio controller to AC97 interface
- **Caches**
  - 16KB Non-Blocking Data Cache
  - 16KB Instruction Cache
  - Instruction/Data Caches are 4-Way, Set-Associative
  - Write-Back with Read-Allocate
  - Cache Management Features:
    Programmable Allocation Policy;
    Line Locking;
    Prefetch Instructions (Instruction and Data)
  - High Speed Access to on-chip Buses

- **MMU**
  - Instruction and Data Watch Registers for Software Breakpoints
  - Separate Interrupt Exception Vector TLB:
    32 Dual-Entry Fully-Associative;
    Variable Page Sizes 4KB-16MB;
    4-Entry ITB
- **Core Micro-Architecture Pipeline**
  - Scalar 5-Stage Pipeline
  - Load/Store Adder in I-Stage
  - Scalar Branch Techniques Optimized:
    Pipelined Register File Access in Fetch Stage
  - Zero Penalty Branch
- **Multiply-Accumulate (MAC) and Divide Unit**
  - Max Issue Rate of One 32x16 MAC per Clock
  - Max Issue Rate of One 32x32 MAC per Every Other Clock
  - Operates in Parallel to CPU Pipeline
  - Executes all Integer Multiply and Divide Instructions
  - 32 X 16-Bit MAC Hardware
- **Low CPU Power**
  - Less than 500mW @ 400Mhz
  - Power-Saving Modes: Idle and Sleep
  - Static Design To 0Hz

Memory

SDRAM

EnCore M3 memory uses a single 144-pin, 3.3v, Small Outline (SO) DIMM module with PC 100 (100MHz or faster), non-ECC/Parity, unbuffered SDRAM. The SDRAM is controlled by the Au1500 internal memory controller.

The EnCore M3 module has been designed to utilize a 32-bit wide SDRAM SODIMM. The Intel Specification Rev 1.0 Feb 1999 allows for two 32-bit banks on the SODIMM and one of the banks may be unpopulated. The standard commercially available SODIMM modules are typically 64-bits wide, divided into two 32-bit banks. The EnCore M3 can only access one 32-bit bank of chips on a common 64-bit wide, 144-pin, SDRAM SODIMM, leaving half of the memory unuseable. There are 32-bit wide SODIMM modules commercially available, but these are less common than the standard 64-bit configuration.

Flash Memory

The EnCore M3 module has 2MB of flash memory on the board in a single 16-bit wide flash device. The flash device used is an AMD 29F160DB. Ampro expects resident flash memory to be used for a bootloader function and the operating system and applications code to be loaded from a network or IDE device. Ampro provides a bootloader for this purpose. See Chapter 4, *Software*, for a description of the Ampro Bootloader and Flash Memory mapping.

PCI Bus Interface (J3)

The PCI Bus uses a 120-pin (30x4) 2mm header interface. This interface header will carry all of the appropriate PCI signals, which can operate at clock speeds up to 66MHz. The interface header is located on the bottom of the module. The module has been designed to accommodate four PCI devices on the baseboard design. The following table describes the PCI Bus (J3) connector information.

NOTE	There are four groups of signals and each group has its own clock signal. The four PCI clock signals have staggered trace lengths, where PCI Clk3 is the shortest, while PCI Clk0 is the longest. Each Clock line has an additional trace length of 650 mils.

Table 3-1. PCI Bus Interconnect Pins and Signals (J3)

Pin #/ (Pin Ref)	Name	In/Out	Description
1 (A1)	NC		Not connected
2 (A2)	VI/O		+5 volts ±5% power supply
3 (A3)	AD05	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
4 (A4)	C/BE0*	T/S	PCI Bus Command/Byte Enables. These lines are multiplexed. During the address cycle, the command is defined. During the data cycle, they define the byte enables
5 (A5)	GND		Digital Ground
6 (A6)	AD11	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
7 (A7)	AD14	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
8 (A8)	+3.3V		+3.3 volts ±5% power supply
9 (A9)	SERR*	O/D	System Error is for reporting address parity errors
10 (A10)	GND		Digital Ground
11 (A11)	STOP*	S/T/S	Stop signal indicates the current selected device is requesting the master to stop the current transaction
12 (A12)	+3.3V		+3.3 volts ±5% power supply
13 (A13)	FRAME*	S/T/S	PCI bus Frame access. This is driven by the current master to indicate the start of a transaction and will remain active until the final data cycle
14 (A14)	GND		Digital Ground
15 (A15)	AD18	T/S	PCI Address and Data Bus Lines, address and data are multiplexed on this pin. A bus transaction consists of an address followed by one or more data cycles
16 (A16)	AD21	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
17 (A17)	+3.3V		+3.3 volts ±5% power supply
18 (A18)	IDSEL0	In	Initialization Device Select signals are used as the chip-select signals during configuration
19 (A19)	AD24	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
20 (A20)	GND		Digital Ground
21 (A21)	AD29	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. Bus transactions consist of an address followed by one or more data cycles.
22 (A22)	+5V		+5 volts ±5% power supply

Pin #/ (Pin Ref)	Name	In/Out	Description
23 (A23)	REQ0*	T/S	Bus Request signals. Request indicates to the arbitrator that this device desires use of the bus.
24 (A24)	GND		Digital Ground
25 (A25)	GNT1*	T/S	Grant signals to PCI Masters. Grant indicates to the requesting device that access has been granted
26 (A26)	+5V		+5 volts ±5% power supply
27 (A27)	CLK2	In	PCI clock outputs for 4 external PCI devices, provide timing for all transactions on the PCI bus
28 (A28)	GND		Digital Ground
29 (A29)	+12V		+12 volts ±5% power supply
30 (A30)	NC		Not connected
31 (B1)	NC		Not connected
32 (B2)	AD02	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
33 (B3)	GND		Digital Ground
34 (B4)	AD07	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
35 (B5)	AD09	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
36 (B6)	VI/O		+5 volts ±5% power supply
37 (B7)	AD13	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
38 (B8)	C/BE1*	T/S	PCI Bus Command/Byte Enables. These lines are multiplexed. During the address cycle, the command is defined. During the data cycle, they define the byte enables
39 (B9)	GND		Digital Ground
40 (B10)	PERR*		Parity Error is for reporting data parity errors
41 (B11)	+3.3V		+3.3 volts ±5% power supply
42 (B12)	TRDY*	S/T/S	Target Ready. This signal indicates the selected devices ability to complete the current cycle of transaction. Both IRDY* and TRDY* must be asserted to terminate a data cycle
43 (B13)	GND		Digital Ground
44 (B14)	AD16	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
45 (B15)	+3.3V		+3.3 volts ±5% power supply
46 (B16)	AD20	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
47 (B17)	AD23	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
48 (B18)	GND		Digital Ground
49 (B19)	C/BE3*	T/S	PCI Bus Command/Byte Enables. These lines are multiplexed. During the address cycle, the command is defined. During the data cycle, they define the byte enables
50 (B20)	AD26	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
51 (B21)	+5V		+5 volts ±5% power supply
52 (B22)	AD30	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles

Pin #/ (Pin Ref)	Name	In/Out	Description
53 (B23)	GND		Digital Ground
54 (B24)	REQ2*	T/S	Bus Request signals. Request indicates to the arbitrator that this device desires use of the bus
55 (B25)	VI/O		+5 volts ±5% power supply
56 (B26)	CLK0	In	PCI clock outputs for 4 external PCI devices, provide timing for all transactions on the PCI bus
57 (B27)	+5V		+5 volts ±5% power supply
58 (B28)	INTD*	O/D	Interrupt D is used to request interrupts only for multi-function devices
59 (B29)	INTA*	O/D	Interrupt A is used to request an interrupt
60 (B30)	NC		Not connected
61 (C1)	+5V		+5 volts ±5% power supply
62 (C2)	AD01	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
63 (C3)	AD04	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
64 (C4)	GND		Digital Ground
65 (C5)	AD08	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
66 (C6)	AD10	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
67 (C7)	GND		Digital Ground
68 (C8)	AD15	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
69 (C9)	SB0*		Snoop Backoff
70 (C10)	+3.3V		+3.3 volts ±5% power supply
71 (C11)	LOCK*	S/T/S	Lock signal indicates an operation that may require multiple transactions to complete
72 (C12)	GND		Digital Ground
73 (C13)	IRDY*	S/T/S	Initiator Ready. This signal indicates the master ability to complete the current data cycle of the transaction
74 (C14)	+3.3V		+3.3 volts ±5% power supply
75 (C15)	AD17	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
76 (C16)	GND		Digital Ground
77 (C17)	AD22	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
78 (C18)	IDSEL1		Initialization Device Select signals are used as the chip-select signals during configuration
79 (C19)	VI/O		+5 volts ±5% power supply
80 (C20)	AD25	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
81 (C21)	AD28	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
82 (C22)	GND		Digital Ground
83 (C23)	REQ1*	T/S	Bus Request signals. Request indicates to the arbitrator that this device desires use of the bus
84 (C24)	+5V		+5 volts ±5% power supply

Pin #/ (Pin Ref)	Name	In/Out	Description
85 (C25)	GNT2*	T/S	Grant signals to PCI Masters. Grant indicates to the requesting device that access has been granted
86 (C26)	GND		Digital Ground
87 (C27)	CLK3	In	PCI clock outputs for 4 external PCI devices, provide timing for all transactions on the PCI bus
88 (C28)	+5V		+5 volts ±5% power supply
89 (C29)	INTB*	O/D	Interrupt B is used to request interrupts only for multi-function devices
90 (C30)	PME*		Using for power management event
91 (D1)	AD00	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
92 (D2)	+5V		+5 volts ±5% power supply
93 (D3)	AD03	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
94 (D4)	AD06	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
95 (D5)	GND		Digital Ground
96 (D6)	M66EN		This signal enables 66MHz operation.
97 (D7)	AD12	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
98 (D8)	+3.3V		+3.3 volts ±5% power supply
99 (D9)	PAR	T/S	PCI bus Parity bit. This is the even parity bit on AD[31:0] and C/BE[3:0]*
100 (D10)	SDONE		Snoop Done
101 (D11)	GND		Digital Ground
102 (D12)	DEVSEL*	S/T/S	Device Select is driven by the target device when its address is decoded
103 (D13)	+3.3V		+3.3 volts ±5% power supply
104 (D14)	C/BE2*		PCI Bus Command/Byte Enables. These lines are multiplexed. During the address cycle, the command is defined. During the data cycle, they define the byte enables
105 (D15)	GND		Digital Ground
106 (D16)	AD19	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
107 (D17)	+3.3V		+3.3 volts ±5% power supply
108 (D18)	IDSEL2		Initialization Device Select signals are used as the chip-select signals during configuration
109 (D19)	IDSEL3		Initialization Device Select signals are used as the chip-select signals during configuration
110 (D20)	GND		Digital Ground
111 (D21)	AD27	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
112 (D22)	AD31	T/S	PCI Address and Data Bus Lines, address and data are multiplexed. A bus transaction consists of an address followed by one or more data cycles
113 (D23)	VI/O		+5 volts ±5% power supply
114 (D24)	GNT0*	T/S	Grant signals to PCI Masters. Grant indicates to the requesting device that access has been granted
115 (D25)	GND		Digital Ground
116 (D26)	CLK1	In	PCI clock outputs for 4 external PCI devices, provide timing for all transactions on the PCI bus

Pin #/ (Pin Ref)	Name	In/Out	Description
117 (D27)	GND		Digital Ground
118 (D28)	RST*	In	PCI bus reset, an output signal to reset the entire PCI Bus. This signal will be asserted during system reset
119 (D29)	INTC*	O/D	Interrupt C is used to request interrupts only for multi-function devices
120 (D30)	GND		Digital Ground

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion.

The Input/Output signals in this table refer to the input/output signals listed in the *PCI Local Bus Manual*, Revision 2.2, Chapter 2, paragraph 2.1, Signal definitions. The following acronyms are used in this table:

- In – Input is standard input only signal
- Out – Totem Pole output is a standard active driver
- T/S – Tri-State is a bi-directional input output pin
- S/TS – Sustained Tri-State is an active low tri-state signal driven by one and only one agent at a time
- O/D – Open Drain allows multiple devices to share as a wire-OR.

Input/Output Interfaces

The EnCore M3 module supports a number of industry standard I/O interfaces. These include an EIDE drive, floppy disk drive, serial and parallel ports, AC97 audio, IrDA, keyboard, and mouse. These interfaces are driven primarily from the Au1500™ CPU and a VIA VT82C686B Southbridge/Super I/O controller. The interface signals are output on one of the three I/O connectors.

The following table describes which I/O interface is driven by each chip to which I/O connector, and the following sections describe the features of each of the I/O interconnections and the connector pin-outs.

Table 3-2. I/O Interface to Chip and I/O Connector

I/O Interface	Chip Source	EnCore M3 I/O Connector
EIDE	VT82C686B	IDE (J1)
Floppy Disk Interface	VT82C686B	Floppy/Serial/Parallel (J8)
Serial Port #1	VT82C686B Port 0	Floppy/Serial/Parallel (J8)
Serial Port #2	VT82C686B Port 1	Floppy/Serial/Parallel (J8)
Serial Port #3	Au1500 Port 0	Serial Port #3 (J15)
Parallel Port	VT82C686B	Floppy/Serial/Parallel (J8)
USB Ports 1 & 2	Au1500	Utility (J9)
AC97 Audio	Au1500	Utility (J9)
IrDA	VT82C686B	Utility (J9)
PS/2 Keyboard	VT82C686B	Utility (J9)
PS/2 Mouse	VT82C686B	Utility (J9)

IDE Interface (J1)

The IDE interface is connected to the baseboard with a 44-pin connector. The IDE is implemented on the VIA VT82C686B Southbridge (Super I/O) chip.

Features:

- Master mode PCI supporting two Enhanced IDE devices including Compact Flash
- Transfer rate up to 33MB/sec
- Increased reliability using Ultra DMA-66 transfer protocols
- 32 levels (double words) of pre-fetch and write buffers
- Supports ATAPI and DVD devices
- Support PCI native and ATA compatibility modes

Table 3-3. IDE Interface Pins and Signals (J1)

Pin #	Name	In/Out	Description
1	RESET*	I	Low active hardware reset (RSTDRV inverted)
2	GND		Digital Ground
3	D7	I/O	Disk Data
4	D8	I/O	Disk Data
5	D6	I/O	Disk Data
6	D9	I/O	Disk Data
7	D5	I/O	Disk Data
8	D10	I/O	Disk Data
9	D4	I/O	Disk Data
10	D11	I/O	Disk Data
11	D3	I/O	Disk Data
12	D12	I/O	Disk Data
13	D2	I/O	Disk Data
14	D13	I/O	Disk Data
15	D1	I/O	Disk Data
16	D14	I/O	Disk Data
17	D0	I/O	Disk Data
18	D15	I/O	Disk Data
19	GND		Digital Ground
20	KEY		Key pin plug
21	IDRQ0	I/O	Primary Device DMA Channel Request
22	GND		Digital Ground
23	IOW*	I	Primary Device I/O Read/Write Strobe
24	GND		Digital Ground
25	IOR*	I	Primary Device I/O Read/Write Strobe
26	GND		Digital Ground
27	IORDY	O	Primary Device I/O-DMA Channel Ready
28	RESERVED		Reserved for future use, not connected
29	IDACK0*	I	Primary Device DMA Channel Acknowledge

Pin #	Name	In/Out	Description
30	GND		Digital Ground
31	IRQ14	O	Interrupt Request 14
32	NC		Not connected
33	A1	I	IDE ATA Primary Disk Address. Used to indicate which byte in either the ATA command block or control block is being accessed
34	NC		Not connected
35	A0	I	IDE ATA Primary Disk Address. Used to indicate which byte in either the ATA command block or control block is being accessed
36	A2	I	IDE ATA Primary Disk Address. Used to indicate which byte in either the ATA command block or control block is being accessed
37	CS0	I	Primary Slave/Master Chip Select
38	CS1	I	Primary Slave/Master Chip Select
39	RESERVED		Reserved for future use, not connected
40	GND		Digital Ground
41	+5V		+5 volts ±5% power supply
42	+5V		+5 volts ±5% power supply
43	GND		Digital Ground
44	RESERVED		Reserved for future use, not connected

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion.

Floppy Disk, Parallel, and Serial Interfaces (J8)

The Floppy Disk Drive interface, Parallel Port interface and two of the Serial Port interfaces are contained within a single 44-pin header (J8). This header is located on the bottom side of the EnCore M3 module to allow it to be plugged directly into a baseboard. Table 3-4 lists the signals on the Floppy/Parallel/Serial interface connector.

The Floppy Disk Drive interface and Parallel Port interface share the same physical set of pins. The actual configuration of the hardware is software controlled. Ampro's bootloader software configures this port for floppy disk drive operation until parallel mode is selected. Refer to *Programming Considerations* in Chapter 4, *Software* for techniques to change the floppy/parallel port configurations.

Table 3-4. Floppy/Parallel/Serial Interface Pins and Signals (J8)

Pin #	Signal	In/ Out	Description
1	STB	I/O	Strobe – Output used to strobe data into the printer. I/O pin in ECP/EPP mode.
2	AFD/ DRVEN0	I/O	Auto Feed – This is Request signal to the printer to automatically feed one line after each line is printed. Drive (Floppy) Density Select 0.
3	PD0/ INDEX	I/O	Parallel Port Data Index – Sense to detect that the head is positioned over the beginning of a track
4	ERR/ HDSEL	O	Error – This is a Status output signal from the printer. A Low State indicates an error condition on the printer Head Select – Selects the side for Read/Write operations (0 = side 1, 1 = side 0)
5	PD1/ TRK0	I/O	Parallel Port Data Track 0 – Sense to detect that the head is positioned over track 0.
6	INIT/ DIR	I/O	Initialize – This signal used to Initialize printer. Output in standard mode, I/O in ECP/EPP mode. Direction – Direction of head movement (0 = inward motion, 1 = outward motion).
7	PD2/ WRTPRT	I/O	Parallel Port Data Write Protect – Senses the diskette is write protected.
8	SLIN/ STEP	I/O	Select In – This signal Output used to select the printer. I/O pin in ECP/EPP mode. Step – Low pulse for each track-to-track movement of the head.
9	PD3/ RDATA	I/O	Parallel Port Data Read Data – Raw serial bit stream from the drive for read operations.
10	GND		Digital Ground
11	PD4/ DSKCHG	I/O	Parallel Port Data Disk Change – Senses the drive door is open or the diskette has been changed since the last drive selection.
12	GND		Digital Ground
13	PD5	I/O	Parallel Port Data
14	GND		Digital Ground
15	PD6	I/O	Parallel Port Data
16	GND		Digital Ground
17	PD7	I/O	Parallel Port Data
18	GND		Digital Ground

Pin #	Signal	In/ Out	Description
19	ACK/ DS1	I	Acknowledge – This is a Status output signal from the printer. A Low State indicates it has received the data and is ready to accept new data. Drive Select 1 – Select drive 1.
20	GND		Digital Ground
21	BSY/ MTR1	I	Busy – This is a Status output signal from the printer. A High State indicates the printer is not ready to accept data. Motor Control 1 – Select motor on drive 1.
22	GND		Digital Ground
23	PE/ WDATA	I	Paper End – This is a Status output signal from the printer. A High State indicates it is out of paper. Write Data – Encoded data to the drive for write operations.
24	GND		Digital Ground
25	SLCT/ WGATE	I	Select – This is a Status output signal from the printer. A High State indicates it is powered on. Write Gate – Signal to the drive to enable current flow in the write head.
26	KEY		Key Plug Pin
27	DCD1	O	Data Carrier Detect 1 – Indicator to serial port 1 that external modem is detecting a carrier signal (i.e., a communication channel is currently open). In direct connect environments, this input will be driven by DTR1 as part of the DTR/DSR handshake. Designed for direct input from external RS232 receiver.
28	DSR1	O	Data Set Ready 1 – Indicator to serial port 1 that external serial communications device is powered, initialized, and ready. Used as hardware handshake with DTR1 for overall readiness to communicate. Designed for direct input from external RS232 receiver.
29	RXD1	O	Receive Data 1– Serial port 1 receive data in.
30	RTS1	I	Request To Send 1 – Indicator to serial output port 1 is ready to transmit data. Used as hardware handshake with CTS1 for low level flow control. Designed for direct input to external RS232 driver.
31	TXD1	O	Transmit Data 1 – Serial port 1 transmit data out.
32	CTS1	O	Clear To Send 1 – Indicator to serial port 1 that external serial communications device is ready to receive data. Used as hardware handshake with RTS1 for low level flow control. Designed for input from external RS232 receiver.
33	DTR1	I	Data Terminal Ready 1 – Serial port 1 indicator that port is powered, initialized, and ready. Used as hardware handshake with DSR1 for overall readiness to communicate. Designed for direct input to external RS232 driver.
34	RI1	O	Ring Indicator 1 – Indicator to serial port 1 that external modem is detecting a ring condition. Used by software to initiate operations to answer and open the communications channel. Designed for direct input from external RS232 receiver (whose input is typically not connected in direct connect environments).
35	GND		Digital Ground
36	SEL1	NC	Interface Mode Select Input 1 – Not connected
37	RXD2	I	Receive Data 2 – Serial port 2 receive data in.
38	RTS2		Request To Send 2 – Indicator to serial output port 2 is ready to transmit data. Used as hardware handshake with CTS2 for low level flow control. Designed for direct input to external RS232 driver.
39	TXD2	O	Transmit Data 2 – Serial port 2 transmit data out.
40	CTS2	O	Clear To Send 2 – Indicator to serial port 2 that external serial communications device is ready to receive data. Used as hardware handshake with RTS2 for low level flow control. Designed for input from external RS232 receiver.
41	VCC		+5 volts ±5% power supply input

Pin #	Signal	In/ Out	Description
42	VCC		+5 volts ±5% power supply input
43	GND		Digital Ground
44	SEL2	NC	Interface Mode Select Input 2 – Not connected

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion

Floppy Disk Controller

The Floppy Disk Controller is provided by the VIA VT82C686B Southbridge (Super I/O) chip. It offers the following capabilities:

- Supports 360k 5.25", 1.2M 5.25", 720k 3.5", and 1.44M 3.5" drives
- 16 Byte FIFO
- Data Rates Up To 1Mbps
- One Floppy Disk Drive

Parallel Port Interface

The Parallel Port is provided by the VIA VT82C686B Southbridge (Super I/O) chip. It offers standard mode, ECP, and EPP protocol support.

Serial Ports

The Serial Ports are provided by both the Au1500 CPU and the VIA VT82C686B Southbridge (Super I/O) chip. The VIA chip is connected to serial ports 1 and 2 found on the baseboard and both ports provide full modem support (RXD, RTS, TXD, CTS, DTR, and RI signals). UART0 on the Au1500 CPU is connected to serial port 3, which is located on the EnCore M3 module.

All three serial ports utilize the TTL signal interface. Serial port 2 is multiplexed to provide the Infrared interface. Serial ports 1 or 2 are typically used for the standard system console interface from YAMON, TimeSys Linux, or Wind River VxWorks. Serial port 3 does not provide full modem support, but does provide a serial debug port with sufficient capability a serial debug interface. Refer to Appendix B for the special serial debug connector used with serial port 3.

Utility Interface (J9)

All remaining I/O signals except Ethernet are provided on a single 44-pin header (J9). This header is located on the bottom side of the EnCore M3 module to allow the module to be plugged directly into a baseboard. Table 3-6 lists the signals on the Utility connector. Key features on the Utility I/O connector include:

- System Management Bus
- Universal Serial Bus
- AC97 Audio interface
- PC Speaker
- IrDA
- PS/2 Keyboard (Port available but not used)
- PS/2 Mouse (Port available but not used)

Table 3-5. Utility Interface Pins and Signals (J9)

Pin #	Name	In/ Out	Description
1	AC_BIT_CLK	O	Audio CODEC 97 Clock
2	AC_DATA_IN	O	Audio CODEC 97 Data In
3	AC_DATA_OUT	I	Audio CODEC 97 Data Out
4	AC_SYNC	I	Audio CODEC 97 Sync
5	AC_RESET#	I	Audio CODEC 97 Reset
6	GND		Digital Ground
7	SPKR	I	Speaker Output
8	GND		Digital Ground
9	RSTSW#	O	Reset Switch
10	KBDATA	I	Keyboard Data
11	KBCLK	I	Keyboard Clock
12	GND		Digital Ground
13	+5V		+5 volts ±5% power supply input
14	MDATA	I	Mouse Data.
15	MCLK	I	Mouse Clock
16	GND		Digital Ground
17	+5V		+5 volts ±5% power supply input
18	NC		Not connected
19	IRTX	I	Infrared Transmit – IR transmit data out from serial port 2
20	IRRX	O	Infrared Receive – IR receive data in to serial port 2
21	GND		Digital Ground
22	NC		Not connected
23	USBPWR0	I/O	USB Port 0 Power Protection – Port 0 is disabled if this input is low. Direct inputs are provided for over current protection.
24	USBP0N	I/O	Universal Serial Bus Port 0 Data Negative
25	USBP0P	I/O	Universal Serial Bus Port 0 Data Positive
26	USBPWR1	I	USB Port 1 Power Protection – Port 1 is disabled if this input is low. Direct inputs are provided for over current protection.
27	USBP1N	I/O	Universal Serial Bus Port 1 Data Negative
28	USBP1P	I/O	Universal Serial Bus Port 1 Data Positive
29	USBP2N	I/O	Universal Serial Bus Port 2 Data Negative
30	USBP2P	I/O	Universal Serial Bus Port 2 Data Positive
31	USBP3N	I/O	Universal Serial Bus Port 3 Data Negative
32	USBP3P	I/O	Universal Serial Bus Port 3 Data Positive
33	NC	NC	Not connected – Held high
34	NC	NC	Not connected
35	RI#	NC	Ring Indicator – Not connected
36	GND		Digital Ground
37	SLPPWR	NC	+3.3V Sleep Power – Not connected
38	AEN	RESVR	Address Enable – Not connected

Pin #	Name	In/ Out	Description
39	SUSC#	NC	Suspend Control – Asserted during power management STD suspends state. Used to control the tertiary power plane. Also connected to ATX power-on circuitry.
40	SMBDATA	Note #	System Management Bus Data
41	SMBCLK		System Management Bus Clock.
42	SMBALRT#		System Management Bus Alert. When the chip is enabled to allow it, assertion generates an IRQ or SMI interrupt or a power management resume event.
43	V1		Voltage 1st – This +3.3 voltage must be applied to the EnCore M3 before any other voltages.
44	GND		Digital Ground

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion.

Address Limits of System Management Bus Data

System Management Bus (SMBus)

The System Management Bus (SMBus), based on the I^2C principles, is a two-wire interface. The SMBus allows communication between various system components on the module and between the module and the baseboard. Some of the SMBus features include allowing a device to provide its manufacturing information, indicating to the system the device's model/part number, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

The SMBus uses a Master/Slave configuration and may have multiple masters and slaves on the bus, but only one device may act as the master device at a time. The SMBus also incorporates an arbitration mechanism to prevent multiple devices from taking control of the bus. The VIA VT82C686B chip is used to control (master) the SMBus and is used to communicate between the SPD (Serial Presence Detect) and the SODIMM EEPROM.

Universal Serial Bus (USB)

The system contains a root hub and two functional ports. Port 1 and Port 2 include over-current detection status on USB inputs. Both USB v.1.1 and Open HCI v.1.1 are supported.

The EnCore M3 does not include fuses for signal conditioning on the module. Both ports require logic and port connectors on a baseboard.

Audio Controller

The Au1500 chip provides the audio control circuitry and when combined with driver software and AC97 CODEC on the baseboard, will provide a complete high quality audio solution.

Speaker

This is the speaker drive signal from the EnCore M3 to the speaker interface on the baseboard used for audio indications of the module status.

NOTE	An external drive circuit is required for the speaker in the baseboard design.

Infrared Port (IrDA)

The IrDA (Infrared Data Association) port provides a two-way wireless communications port using infrared as a transmission medium. The VIA VT82C686B (Southbridge) provides two infrared methods, the SIR method and the Amplitude Shift Keyed Infrared (ASKIR) method. The SIR and ASKIR signals are multiplexed on Serial port 2 from the VIA chip.

The SIR (Serial Infrared) method allows serial communication at baud rates up to and including 115kbps. Each word is sent serially beginning with a zero value start bit. A zero is sent when a single infrared pulse is sent at the beginning of the serial bit time. A one is sent when no infrared pulse is sent during the bit time.

The Amplitude Shift Keyed infrared (ASKIR) method allows serial communication at baud rates up to 19.2kbps. Each word is sent serially beginning with a zero value start bit. A zero is sent when a 500kHz waveform is sent for the duration of the serial bit time. A one is sent when no transmission is sent during the serial bit time.

Both of these methods require an understanding of the timing diagrams provided in the Super I/O controller chip specifications available on the manufacture's web site and referenced earlier in this manual. For more information, refer to the VIA VT82C686B chip data book and the Infrared Data Association web site at http://www.irda.org.

Keyboard and PS/2 Mouse Controller

The Southbridg chip (VT82C686B) provides a controller for the PS/2 keyboard and PS/2 mouse, but these ports are not normally used. If you were to put a graphics controller on your baseboard, then you would find these ports useful. These devices are located at the standard PCI I/O addresses of 0X60 and 0X64.

Ethernet (LAN) Interface

The Ethernet solution on the EnCore M3 is provided by the Au1500 processor, which has two independent Ethernet ports. Ethernet port 1 uses the interconnect (J7), a 10-pin header on the bottom of the module, to provide the Ethernet signal to the baseboard where the magnetics for port 1 must be located.

Ethernet port 2 (J14) is located on top of the EnCore module and includes the magnetics for this port. This port (J14) terminates in an 8-pin connector on the module, but requires a special cable with a standard RJ45 connector on the other end as the standard interface. Ampro provides this special cable in the EnCore QuickStart Kit and a drawing for this cable can be found in Appendix B.

The Au1500 processor uses two independent Ethernet Media Access Controller (MAC) devices for the Ethernet solution. Each MAC device (port) provides the interface between the host application and the physical layer through the Media Independent Interface (MII). Both ports support the standard Ethernet protocol requirements and operate in both half and full duplex modes. Refer to the following features list for more information.

Features:

- IEEE 802.3, 802.3u, 803.3x Specification Compliance
- 10/100Mbs Data Transfer Rates
- IEEE 802.3 Compliant MII Interface
- Full and Half Duplex
- CSMA/CD in Half Duplex
- Flow Control Support for Full Duplex
- Collision Detection and Auto Retransmit on Collisions in Half Duplex
- Preamble Generation and Removal
- Automatic 32-bit CRC Generation and Checking
- Optional Automatic Pad Stripping on the Receive Packets.
- External to Internal Loopback

- Filtering Modes Supported On The Ethernet Side:
  - One 48 Bit Perfect Address
  - 64 Hash-Filtered Multicast Addresses
  - Pass All Multicast Addresses
  - Promiscuous Mode
  - Pass All Incoming Packets with A Status Report
  - Toss Bad Packets
- Separate 32-bit Status Returned for Transmit and Receive Packets
- VCI Compliant Application Interface Bus
- Synchronization of Signals for the MII Receive/Transmit Clocks and the Application Clock
- Supports Big/Little Endian Data Format

Table 3-6. Ethernet Interface Pins and Signals (J7)

Pin #	Name	In/ Out	Description
1	TX+	Out	Analog Twisted Pair Ethernet Transmit Differential Pair. These pins transmit the serial bit stream for transmission on the Unshielded Twisted Pair Cable (UTP). These signals interface directly with an isolation transformer.
2	TX-	Out	Analog Twisted Pair Ethernet Transmit Differential Pair. These pins transmit the serial bit stream for transmission on the Unshielded Twisted Pair Cable (UTP). These signals interface directly with an isolation transformer.
3	TXCT		Center tap
4	RXCT		Center tap
5	RX+	In	Analog Twisted Pair Ethernet Receive Differential Pair. These pins receive the serial bit stream from the isolation transformer.
6	RX-	In	Analog Twisted Pair Ethernet Receive Differential Pair. These pins receive the serial bit stream from the isolation transformer.
7	GND		Digital Ground
8	LINKLED		Link Integrity LED – The LINK LED pin indicates link integrity. If the link is valid in either 10Mbps or 100Mbps, the LED is on; if link is invalid, the LED is off.
9	ACTLED		Activity LED. The Activity LED pin indicates either transmit or receive activity. When activity is present, the activity LED is on; when no activity is present, the activity LED is off
10	SPEEDLED		Speed LED. The speed LED pin indicates the speed. The speed LED will be on at 100Mbps and off at 10Mbps.

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion

Table 3-7. Ethernet Port 2 Pins and Signals (J14)

Pin #	Name	In/ Out	Description
1	TX+	Out	Analog Twisted Pair Ethernet Transmit Differential Pair. These pins transmit the serial bit stream for transmission on the Unshielded Twisted Pair Cable (UTP). These signals interface directly with an isolation transformer.
2	TX-	Out	Analog Twisted Pair Ethernet Transmit Differential Pair. These pins transmit the serial bit stream for transmission on the Unshielded Twisted Pair Cable (UTP). These signals interface directly with an isolation transformer.
3	RX+	In	Analog Twisted Pair Ethernet Receive Differential Pair. These pins receive the serial bit stream from the isolation transformer.
4	NC	NA	Not connected
5	NC	NA	Not connected
6	RX-	In	Analog Twisted Pair Ethernet Receive Differential Pair. These pins receive the serial bit stream from the isolation transformer.
7	NC	NA	Not connected
8	NC	NA	Not connected

Miscellaneous Features

Time of Day Clock

The EnCore M3 provides a time of day clock. It is implemented in the Au1500 processor using the Time of Year (TOY) timers.

EJTAG (J12)

The EJTAG feature is provided through the EJTAG connector (J12). EJTAG is a hardware/software subsystem that provides comprehensive debugging and performance tuning capabilities for MIPS® microprocessors. It exploits the infrastructure provided by the IEEE 1149.1 JTAG Test Access Port (TAP) standard to provide an external interface and extends the MIPS® instruction set and privileged resource architecture to provide a standard software architecture for integrated software debugging.

Further details can be obtained from the MIPS web site at: www.mips.com. Ampro has validated a number of commercially available EJTAG tools with the EnCore M3. Refer to the Ampro web site for a list of available tools.

Table 3-8. EJTAG Interface Pins and Signals (J12)

Pin #	Name	In/ Out	Description
1	TRst	In	Asynchronous TAP reset
2	GND	NA	Digital Ground
3	TDI	I	Test data input to the instruction or selected data registers. This signal will be sampled on the rising edge of TCK.
4	GND	NA	Digital Ground
5	TDO	O	Test data output to the instruction or selected data registers. This signal will transition on the falling edge (valid on rising edge) of TCK.
6	GND	NA	Digital Ground
7	TMS	I	Control signal for TAP controller. This signal is sampled on the rising edge of TCK.
8	GND	NA	Digital Ground
9	TCK	I	Control clock for updating TAP controller and shifting data through instruction or selected data register.
10	GND	NA	Digital Ground
11	Rst	NA	Processor reset – Allows for remote reset
12	Key	NA	None
13	DInt	NA	None
14	VIO	NA	None

Note: The shaded area denotes power or ground. The signals marked with * indicate signal inversion

LED Signals

The EnCore M3 module contains several LED's for convenient visual feedback on key features or user-definable features. These are described in the following tables and the location of each LED may be found in Figure 2-6.

NOTE	The LEDs for Ethernet Port #1 activity and speed are on the baseboard.

Table 3-9. LED Signal Descriptions

Signal	Description
ACTLED / LINKLED	Ethernet Port #2 Activity/Link LED. This yellow LED indicates either transmit or receive activity. When activity is present, the activity LED is flashing.
SPEEDLED	Ethernet Port #2 Speed LED. This green speed LED indicates the speed of the Ethernet link. The speed LED will be on at 100Mbps and off at 10Mbps.
GPIO 202 LED (D1)	User-definable LED attached to GPIO 202 (green) - Pull this corresponding GPIO pin low to activate the user-definable LEDs. Pull this GPIO pin high to turn the user-definable LED off.
GPIO 203 LED (D2)	User-definable LED attached to GPIO 203 (yellow) - Pull this corresponding GPIO pin low to activate the user-definable LEDs. Pull this GPIO pin high to turn the user-definable LED off.

Power Interface (J4)

The power input connector supplies +5V and +3.3V to the EnCore M3 module.

NOTE Since the EnCore M3 module requires extremely low power, some commercial power supplies may become unstable at the low power consumption levels. To alleviate this problem, use a power supply that remains stable at 2W. In sleep mode or at lower clock speeds no reliable data exists but power requirements approach 0 Watts. For this reason, when in sleep mode or at lower clock speeds Ampro recommends using a battery or "brick" as the power source.

Table 3-10. Power Interface Pins and Signals (J4)

Pin #	Name	In/Out	Description
1	+5		+5 volts ±5% power supply
2	GND		Digital Ground
3	+3.3		+3.3 volts ±5% power supply
4	GND		Digital Ground
5	+12		Not used
6	GND		Digital Ground
7	+3.3		+3.3 volts ±5% power supply
8	GND		Digital Ground
9	+5		+5 volts ±5% power supply
10	GND		Digital Ground

Chapter 4 Software

Overview

The EnCore Development Kit (EDK) CD-ROM Set contains support software and development tools for the EnCore M3 platform. The development tools include versions that will run on Windows-based systems and Linux/Unix-based workstations. The EDK CD-ROM Set also includes utilities and Board Support Packages (BSP) for the Ampro supported operating systems. Additional support may found the Ampro web site: The following items are included on the EDK CD-ROM Set:

- YAMON – This is a monitor/bootloader program originally created by MIPS Technologies, which is resident in flash on the EnCore M3 Module.
  - Source code files
  - Binary file – A binary image of YAMON suitable for programming into flash.
- gnu development tools – This is a complete distribution of the gnu tools from the Free Software Foundation.
  - Cygwin (no source files) – This environment is for development on Windows-based systems.
  - Source files for gnu tools
- VxWorks BSP – This includes a complete Board Support Package (BSP) for VxWorks – Tornado. The VxWorks BSP requires licensing a Tornado development environment from Wind River, which can not be purchased through Ampro. Contact Wind River directly for details.
- Linux – a full distribution ofTimeSys Linux. The real time capabilities of this distribution can be activated by obtaining a license key from TimeSys Corp. The standard capabilities (GPL) are provided without out any key or purchase requirements.

YAMON

Development Environment

Software development for the EnCore M3 system is typically done in a Host/Target configuration, where the OS/Application code is developed on a host development workstation, and the resulting executable image is downloaded onto the target system. A typical development environment is shown in the figure below:

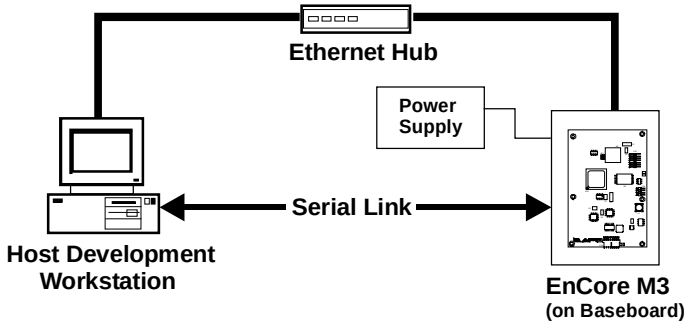

Figure 4-1. Typical Host/Target Development Environment

The EnCore M3 EDK CD Set contains tools for code development for the MIPS32 processor environment. These tools are all based on the gnu compiler technology as distributed by the Free Software Foundation, and Cygwin distributed by Red Hat, Inc.

NOTE	Linux development requires specific tools. These are provided in the TimeSys Linux distribution included with the EnCore QuickStart Kit CD-ROM set.

Ampro has also tested the version of the gnu compiler from Algorithmics, Inc., called the "SDE" version of the compiler. This is available over the web from Algorithmics at www.algor.co.uk.

Installation and Configuration Overview

YAMON is pre-installed in flash memory on the EnCore M3 module. You may use the supplied tools to modify and/or rebuild YAMON. The tools support development under both Win32 systems and Unix workstations. The following instructions detail the installation procedure for the standard cross development tools used with the YAMON environment. These tools can also be used to develop application programs that can be loaded via YAMON. However, most operating systems come with their own tool sets and installation instructions.

Installation on Win 32 Systems

Windows based development workstations will require a POSIX development environment for the gnu tool set. This is provided by the Cygwin environment, which is supplied in the EnCore QuickStart Kit CD-ROM set. You will need a personal computer with Microsoft Windows 98, NT 4.0 SP6, or 2000. Approximately 500M of disk space is necessary. (We also recommend a PC with a 233MHz or faster CPU and at least 128MB RAM).

The tools are precompiled to work under the Cygwin environment. The Cygwin environment provides a Unix-like environment for Windows systems. Most of the standard Unix commands (e.g. make, ls, rm, gcc, perl, etc.) are available with the Cygwin environment, and allow easy migration of the tools and code between Windows and Unix platforms. The latest information related to the Cygwin environment is available at:

```
http:// sources.redhat.com/cygwin
```

The Cygwin environment is freely available and is included here in the EnCore QuickStart Kit CD-ROM set as a convenience. The following steps will install the gnu tools on your system:

Start the setup program by executing Cygwin setup.exe from the CD-ROM

```
M:/cygwin_install/setup.exe  (assuming M: is the CD-ROM drive)
```

The following parameters should be used when installing Cygwin:

```
> Install from Local Directory
> Package Directory: M:/cygwin_install
> Default Text File Type: UNIX
```

Once the Cygwin system is installed, you can install the pre-configured cross development environment. This is provided as a "tar ball", and should be installed in a cygwin shell window.

1. Start a cygwin shell by clicking on the cygwin icon.
2. Go to the installation directory.

   ```
   cd/usr/local
   ```

3. Un-tar the compiler.

   ```
   tar –xvf /cygdrive/m/comp2953.tar(assuming M is the CD-ROM drive)
   ```

The executable gnu tools are now located in the /usr/local/comp/mips-elf/2.95.3/bin directory. It is recommended to add this path to your environment variable $PATH. In the .bashrc file in your login directory, you can add the line:

```
PATH=/usr/local/comp/mips-elf/2.95.3/bin:$PATH; export PATH
```

or, you can edit the current set PATH line to include the compiler path.

Installation on Unix/Linux Systems

For UNIX systems, the source distribution of the tools is included. The compiler will have to be re-built before it can be used. Use the following set of commands to perform the installation. This assumes that you are installing the compiler in a directory called "crossgcc", but you can use any directory that makes sense for your configuration.

NOTE	When building the gcc compiler, you will need to install several other utility functions on the build machine. Ensure you check the gnu site (www.gnu.org) for latest instructions on how to build the compiler before attempting to build the compiler.

```
% mkdir crossgcc
% cd crossgcc
% tar xvzf /cdrom/gcc-2.95.3.tar.gz
% tar xvzf /cdrom/binutils-2.11.tar.gz
% tar xvzf /cdrom/newlib-1.9.0.tar.gz
% tar xvzf /cdrom/gdb-5.0.tar.gz
# allows MIPS32 as –mips2
% cp /cdrom/mips-opc.c binutils-2.11/opcodes
% cp /cdrom/mkcomp .
% vi mkcomp # edit INSTALL location
% ./mkcomp
```

YAMON Overview

YAMON is a monitor program provided by MIPS Technologies, and modified by Ampro for the EnCore M3. This monitor provides basic system initialization and debug capabilities, and provides a mechanism for loading applications and operating systems. YAMON is not an operating system. It does not support switching between several concurrent applications, nor does it manage the memory/TLB. YAMON is a monitor performing "BIOS" like system initialization and allowing the user to examine/modify memory and memory-mapped devices as well as loading and starting applications one at a time. As such, YAMON is well suited for hardware and software bring-up.

Some of the main YAMON features are:

- System initializations, including RAM size/type detection and auto configuration, cache initialization.
- PCI auto detection and auto configuration
- Command Shell with command line history and editing
- Supports traditional shell commands (load, go, dump, edit etc.)
- Ethernet and serial port support
- Configuration of CPU
- Flash programming

YAMON supports the following interfaces:

- Command line interface through a serial port
- Debug interface through a dedicated debug serial port. Interface conforms to gnu-gdb
- "Standard Remote Protocol" with extensions for sde-gdb from Algorithmics
- Vector table based call interface for use by applications
- Ethernet for TFTP-load and "ping" support

Ampro has extended YAMON to offer the following additional features:

- RAM and flash memory test on Startup
- I/O space register access (Southbridge)
- Automatic RAM size detection
- Program loading from IDE devices such as hard disk or Compact Flash using FAT12, FAT16 and FAT32 File Systems
- Support for FAT12 floppy volumes

YAMON Development Environment

The development environment for YAMON is the gnu cross compiler environment. Installation instructions for the gnu environment are in the previous section. This environment allows the user to make modifications to YAMON, or to write programs that can be downloaded and run via YAMON.

Installation and Configuration Sequence

The source for the YAMON monitor is included in a tarball on the EDK. This can be installed on a Win32 based system using the Cygwin environment, or by using a program such as WinZip. You can unzip/untar the baseline anywhere on your system that you find convenient. The makefile supports both the gnu toolset and the Algorithmics toolset. If you are using the Algorithmics SDE4 compiler and development environment, you must also obtain a Perl interpreter for Win32. There are several that can be obtained free on the web.

In order to compile the YAMON code, simply run make on the makefile in the YAMON root directory. This will create the executable image for the reset code, and the Little Endian and Big Endian versions of the monitor. These are concatenated into a single binary and S-Record image that are suitable for downloading into the flash memory.

YAMON comes pre-installed on the EnCore M3 QuickStart Kit. If you want to upgrade the monitor, you will need to program the flash. This can be accomplished either by a tool using the EJTAG port or by the YAMON "load" command. This command will take an S-record image, and load it directly into the flash area. This is the "normal" method for upgrading the YAMON code. See the "load" command in the following section for details, as well as the flash management section under *Programming Considerations*.

YAMON is started as part of the hardware reset process. Turning on the EnCore M3 module will cause execution to begin at the reset vector location (0xbfc00000) and the monitor will start up. The following prompt will appear:

```
YAMON ROM Monitor, Revision 02.20AMP.
Copyright (c) 1999-2000 MIPS Technologies, Inc.
------------ All Rights Reserved --------------
Copyright (c) 2002 Ampro Computers, Inc.

For a list of available commands, type 'help'.

Compilation time =               Aug 22 2002  13:01:29
Board type/revision =            0x05 (EnCore M3) /
MAC address =                    00.00.00.00.02.01
Board S/N =                      xxxxxx (up to 10 digits)
Processor Company ID =           0x03
Processor ID/revision =          0x02 / 0x00
Endianness =                     Big
PU =                             396 MHz
Flash memory size =              1 Mbyte
SDRAM size =                     16 Mbyte
First free SDRAM address =       0x800e53c0

Environment variable 'start' exists but is disabled

YAMON>
```

The last message gives the user the option of having the system auto-load the default image. If you have set up your system to autoload, you may abort the auto-load function, using the **Ctrl-C** keystroke, before the default file starts to load. When the system is started initially, there is no image to load, therefore the environment variable start is disabled.

If the auto-load function is aborted, the YAMON> prompt will appear. The YAMON monitor is now ready to use. Typing the "help" command will display a list of all available commands.

One important command is the one to change the endianness of the processor. Type:

```
YAMON>endian –b
```

to select big endian, or –l for little endian. The endianness will not take effect until the board is re-booted.

Once the monitor is running, the user can configure the hardware setup parameters. These parameters are accessed using the "set" command, and will contain the information necessary for running the EnCore M3 module. The following shows the listing of the set command:

```
YAMON> set

MAC             (R/W)  0
baseboardserial (RO)   M3 00006
bootfile        (R/W)  /tftpboot/vmlinux.srec
bootprot        (R/W)  tftp
bootserport     (R/W)  tty0
bootserver      (R/W)  100.10.10.2
gateway         (R/W)  100.10.10.254
ipaddr          (R/W)  100.10.10.4
memsize         (RO)   0x02000000
modetty0        (RO)   115200,n,8,1,none
modetty1        (RO)   115200,n,8,1,none
prompt          (R/W)  YAMON
start           (R/W)
subnetmask      (R/W)  255.255.255.0

YAMON>
```

In order to modify these settings, the set command is used. For example, to modify the IP address, one can use the following command;

```
YAMON> set ipaddr 192.216.227.123
```

The user can also define his own symbols that can be accessed via YAMON when running user downloaded programs. All settings will be saved in Flash memory, thus only need to be loaded once.

The following table describes all of the critical parameters in the system.

Table 4-1. System Parameter Table

Name	Access	Description	Default Value
MAC	(R/W)	This parameter identifies the specific MAC to use. YAMON only supports a single Ethernet interface, and this parameter is used to select the interface to use. MAC 0 selects the RJ-45 connector on the baseboard, while MAC 1 selects the 8-pin interface on the M3 board.	0
baseboard serial	(RO)	This is the board serial number, as stored in the FLASH parameter sector.	Factory Set

Name	Access	Description	Default Value
Bootfile	(R/W)	This parameter identifies the file to load. Depending on the configuration of the TFTP server, you may need to specify the entire path and file name, when using the TFTP protocol. For example, when booting TimeSys Linux this parameter is typically set to: **vmlinux.srec** This file must be a S-Record format file.	NA
bootprot	(R/W)	This selects the boot protocol. This can either be "tftp", which selects TFTP download via the Ethernet port, "file", which reads from an IDE device, or "asc" which selects download via the serial port. For serial download there is no protocol "handshake." The S-Record file can be downloaded without any control information.	tftp
bootserport	(R/W)	This is the serial port used for serial download when bootprot is "asc"	tty0
bootserver	(R/W)	This is the IP address of the TFTP server that is used for TFTP download. Must be set when bootprot is set to "tftp".	0.0.0.0
gateway	(R/W)	This is the IP address of the gateway host on the local subnet.	0.0.0.0
ipaddr	(R/W)	This is the IP address assigned to the Ethernet interface. YAMON does not support DHCP or BOOTP for IP address assignment.	0.0.0.0
memsize	(RO)	This is a read-only parameter that identifies the size of the RAM area for the system. This is the size of memory available from the SODIMM module.	0x02000000
modetty0	(RO)	This is a read-only parameter that shows the port setting of serial port 0. This can only be changed at compile time.	115200,n,8,1,none
modetty1	(RO)	This is a read-only parameter that shows the port setting of serial port 1. This can only be changed at compile time.	115200,n,8,1,none
prompt	(R/W)	This sets the string to be used as a prompt. There are no special characters – this is just a simple string.	YAMON
start	(R/W)	This is an initial command string that is executed after the start delay period. If startdelay is zero, or undefined, then the command is not executed. If start is not defined, no command is executed.	NA
startdelay	(R/W)	Delay in seconds for execution of the start command or string.	0
subnetmask	(R/W)	This is the subnet mask for the local network. See your network administrator for the proper value.	0.0.0.0

Loading and Execution

User written programs can be loaded and executed via the YAMON monitor. YAMON functions (such as the serial driver or Ethernet driver) can be accessed via a jump table, and provide some basic services. Simple programs can be executed, and will return to the monitor prompt.

An example "hello" program is provided as part of the YAMON distribution. Study this make file as an example for downloading your own code.

YAMON will load programs in S-Record format. YAMON can load from a network, download over a serial port, or load from a local IDE device such as a hard disk or Compact Flash device. YAMON can also be used to load an operating system/application.

Command Summary

The following table shows a list of commands available in YAMON, along with a brief description of their function. To get a complete set of options for YAMON commands, simply use the YAMON help command;

```
YAMON> help <command>
```

Where <command> is the command of interest. This will provide a full description of the command and all options.

Table 4-2. YAMON Command Descriptions

Command	Description/Syntax
cache	cache [on|off] Enable/disable caches by setting K0 field of CPU CONFIG register. The command will flush D-cache before the K0 field is modified. If no parameter is supplied, the current cache setting is displayed.
cd	cd <full path name> This command will set the current default directory, including the default device. The M3 module only supports 1 floppy drive, but can support up to 2 IDE devices on the primary IDE controller. The device names are as follows fd Floppy Drive hd Hard Drive For IDE devices (hd), the device ID is indicated by a letter appended to the name (a for the master and b for the slave). Further, each logical drive on the device is indicated by a number starting from zero. Thus, the first partition on a hard disk that is the primary device will be hda0, the second hda1, and so on. A slave device will start with hdb0. It should be noted that Compact Flash devices look like IDE drives to the system, and thus should be considered hd type devices. Path names are constructed in the form //device/path, where path contains the full path name. Special paths such as .. and . are supported as well. If the device name is omitted, the current device is assumed. An example path name would be //hda0/system/temp Where hda0 is the device name. After this command, the command **cd ..** would make the default directory //hda0/system. The device/path names follow Linux conventions. The file systems supported are FAT12, FAT16, and FAT32.
cksum	cksum <address> <size> Computes 32-bit CRC for the specified memory area. The CRC algorithm and polynomial are the same as the Unix "cksum" command.
compare	compare <address1> <address2> <size> Compares the two specified memory areas. If a difference is encountered during the compare, the address of the first mis-match will be reported.
copy	copy [-f] <src> <dst> <size> The number of bytes specified by <size> are copied from <src> to <dst>. Both the source and destination can be located anywhere in the address space. The copy command knows the address areas for the flash memories in the system and is able to program them. If the destination is flash, the destination area must be cleared using the 'erase' command prior to the copy operation. Note that the copy command prevents the user from overwriting the environment flash area. Unless the -f option is applied, caches are flushed before and after the copy operations (D-cache writeback and invalidate, I-cache invalidate). -f Do not flush caches

Command	Description/Syntax
cp0	cp0 [<name> [<value>]] Read/write CP0 register(s). If no arguments are applied, all CP0 registers are shown. If the name of a CP0 register is applied, this register is written or read depending on whether a value is applied or not. Writing a CP0 register takes effect immediately and should be done with care since it may crash YAMON. Some of the CP0 registers are optional and may not be available for the EnCore M3. Settings of CP0 registers are also applied to user applications (started with 'go' or 'gdb') except for STATUS, for which the IE field is cleared. Also, TLB related registers as well as COUNT and COMPARE are undefined when user application is first started.
date	date [[[cc]yy]mmddHHMM[.ss]] (HH in 24 hour format) Set or read date and time. If no argument is given, the present date and time is output. If an argument is specified, it is used to set the date and time on the real time clock. The argument must have the format [[cc]yy]mmddHHMM[.ss] where the hours HH are represented in 24h format (00-23). The output is shown as Day Mon dd HH:MM:SS ccyy.
dis	dis [-m] <address> [<count>] Disassemble code starting at <address> (only MIPS32 instructions are disassembled). <count> (default 16) specifies the number of instructions to disassemble. -m Prompt user for key press after each screen of data
dump	dump [-m][-8|-16|-32] <address> [<size>] Dumps data from address range starting at <address>. The default data display width is 8 bits. The <size> parameter specifies the number of bytes to dump (default is 256). The continuation command '.' works together with 'dump'. -m Prompt user for keypress after each screen of data -8 Dump data in units of bytes -16 Dump data in units of halfwords -32 Dump data in units of words
echo	echo <data> The echo command echoes all of its arguments to the console. This offers a convenient way to examine the contents of specific environment variables (e.g. 'echo $ipaddr').
edit	edit [-8|-16|-32] <address> Edit memory contents starting at <address>. The default data width is 8 bits. Edit mode is exited by typing '.' or Ctrl-C. During the edit, a data element can be left untouched by just pushing Enter. The edit will then continue with the next data element. Typing '-' will cause the address to be decremented. Data is entered using hexadecimal number format (with no leading "0x"). -8 Edit data in units of bytes -16 Edit data in units of halfwords -32 Edit data in units of words
endian	endian (-b | -l | -q) Select processor endian-ness for next boot cycle. This is required to put the EnCore M3 into a mode where it can boot big or little endian operating systems. This will take effect on the next boot cycle -l Set to Little Endian -b Set to Big Endian -q Display current setting

Command	Description/Syntax
erase	erase [-s|-e|<address> <size>] Erase flash memory. An option may be applied specifying which flash region to erase. If no such option is applied, the address range to be erased is specified by the <address> and <size> parameters. If no such range is specified, the range corresponding to the default option is assumed. • If (and only if) the -e option (erase environment flash) is applied, the system environment variables are reinitialized to factory default values. • If a range is specified, all flash blocks touched by the range are cleared. The block size is 64kB in flash memory. The blocks to be cleared are displayed, and the user is asked for confirmation before the operation is performed. Refer to Table 4-7 for the Flash Memory map. Erasing a large flash area takes time. It can easily take several minutes to erase a 2MB area. Any set flash sector lock bits will be cleared before the sector is erased. If they cannot be cleared (e.g. due to hardware protection of the lock bits), the command will fail. -e Erase and reinitialize entire environment area. -s Erase entire system flash (default option).
fill	fill [-8|-16|-32] <address> <size> <data> Fills the specified memory area starting at <address>. Default data width is 8 bits. <size> specifies the size of the area to fill (expressed in bytes). -8 Fill data in units of bytes -16 Fill data in units of halfwords -32 Fill data in units of words
flush	flush [-i|-d] Flush cache(s). By default, the D-cache is flushed first, followed by an I-cache invalidate. This behavior can be changed by the '-i' and '-d' options. The D-cache flush operation is in reality a write-back of dirty lines (write-back caches only) followed by an invalidate operation. -i Invalidate I-cache -d Flush D-cache
gdb	gdb [-v][-c] [. <args>] Setup connection to gdb debugger on port tty1. The Standard gdb remote protocol is used. If the user application is not currently running, and Ctrl-C is typed at the console port, YAMON will leave gdb mode and return to the command prompt. <args> is broken up in substrings and passed to the application. The list of arguments to be passed must begin with a '.'. The '.' is not passed as an argument. The first argument (argv[0]) will be the string 'gdb'. a0 is set to the number of substrings (argc). a1 is set to the address of an array of pointers to the substrings (argv). a2 is set to the address of the environment table. a3 is set to the memory size. ra holds the return address to YAMON. The application may return to YAMON by jumping to the address specified in ra or by calling the exit(rc) function supplied by YAMON. The verbose (-v) option will cause the commands from the gdb host and the responses from YAMON to be displayed on the console port. The checksum off (-c) option will disable validation of the checksum used in gdb commands. This is useful in case the user wishes to enter commands manually. Two checksum characters should still be used in the commands, but the values are “don't care.” -v Display messages from and to gdb host -c Disable check of gdb checksum

Command	Description/Syntax
go	go [.|<address> [<args>]] Execute application code. If a target address is not specified, the address obtained from the last successful 'load' command (if any) is used as the target address. If arguments for the user program need to be specified, the default execution address can be referenced by a '.'. <args> is broken up in substrings and passed to the application. The first argument (argv[0]) will be the string 'go'. a0 is set to the number of substrings (argc). a1 is set to the address of an array of pointers to the substrings (argv). a2 is set to the address of the environment table. a3 is set to the memory size. ra holds the return address to YAMON. The application may return to YAMON by jumping to the address specified in ra or by calling the exit(rc) function supplied by YAMON.
help	help [<command>] 'help' with no parameter shows a list of all the available commands. To get more detailed help on a particular command, specify the command name as an argument to 'help'. When specifying a command as an argument to help, command completion will be performed if at least two characters of the command name have been specified.
info	info [board | cpu | memory | uart | boot | all | pci | lan | ata | fat] Display information on the requested item (default boot). The following information displays can be requested: board : Board properties cpu : CPU properties memory : Memory properties uart : Serial ports statistics boot : Info displayed after reset all : All info pci : PCI autodiscovery/autoconfiguration lan : Ethernet statistics ata : Configuration information for attached ATA devices fat : File system information for FAT12, FAT16 and FAT32 logical volumes
io	io [-8|-16|-32] ([-r] <addr>) | (-w <addr> <val>) i/o - Read a value or write a value to I/O space via PCI. All arguments are hexadecimal. Range parameter indicates the number of bytes to read. Default action is read. Default width is 32-bit. -r Read I/O space -w Write to I/O space -8 See data in units of bytes -16 See data in units of 16 bit words -32 See data in units of 32-bit words
load	Load [-r] ([tftp:][//<ipaddr>][/<filename>]) | ([asc:] [//(tty0|tty1)]) | ([file:] [//<dev>/<path>/<filename>]) -r Retry on ARP timeout (until load succeeds or Ctrl-c is typed) Load image from serial port, Ethernet, or file to RAM or flash (depending on address). The only image type currently supported is SREC. The default protocol can be any of the three. For the M3 module, the default is [tftp]. This can be changed using the set command. • If loading from serial port, the default port is taken from the environment variable 'bootserport'.

Command	Description/Syntax
	• If loading from file, the file name is taken from the 'bootfile' environmental variable. This variable must contain a fully specified device/path/file string, such as "//hda0/system/rec/startfile.srec". The indicated device must be attached and correctly formatted. • If loading from Ethernet, the IP address of the TFTP server and the filename may be specified. If an IP address is not specified, it is taken from the environment variable 'bootserver'. If a filename is not specified, it is taken from the environment variable 'bootfile'. Note that the TFTP protocol limits the maximum file size (SREC file size, not actual binary data size) to 32 Mbytes (even 16 Mbytes on some TFTP servers). If this limit is exceeded, the load will fail. For the currently supported formats, the execution entry point of the image is embedded in the image. This address is saved so the 'go' command can use it as the default entry point. The load command prevents the user from overwriting the environment flash area. This area contains the environmental variables. The load command also protects the board parameter area, where the MAC addresses, serial number, and other information is stored. **CAUTION** **Ensure the image to be loaded into flash works correctly, before you attempt to load it.** If the existing YAMON image in flash is erased, then loaded with a corrupted image, you have no way of unloading the image. You can only re-program the flash via the EJTAG interface. To program a new version of YAMON using the load command, the user must first erase the area using a series of commands to erase the system flash. Once this is erased, you can then load the YAMON image. Refer to *Programming Considerations* later in this chapter or more information.
ls	ls [Path Name] (-a) This command will list the contents of the indicated directory. It will list all files. This works like the UNIX ls command, but with fewer options -a Provides a longer listing
pcicfg	pcicfg [-8|-16|-32] ([-r] <bus> <dev> <func> <addr> [<range>]) | (-w <bus> <dev> <func> <addr> <val>) Read a value/range or write a value to PCI configuration space. All arguments are hexadecimal. Range parameter indicates the number of bytes to read. Default action is read. Default width is 32-bit. -r Read PCI configuration space -w Write to PCI configuration space -8 See data in units of bytes -16 See data in units of halfwords -32 See data in units of words
ping	ping ipaddr [<datagramsize>] ping - send ICMP ECHO_REQUEST packets to network host. This command is typically used to verify end-to-end network functionality & connectivity in a debug or bring-up situation. An ICMP ECHO_REQUEST packet must be replied to with an ICMP ECHO_REPLY packet from the remote host ('ipaddr'). The ICMP ECHO packet will contain data with the specified size. The default datagramsize is 64 bytes, minimum is 0 bytes and maximum is 1472 bytes. The maximum size is constrained by the Ethernet upper frame size limit (IP segmentation is not supported). If the optional datagramsize parameter is not within the valid range, the default size of 64 bytes will be used. The ping command will stop when the first reply is received from the remote host. If no replies are received, depending on whether the MAC-address of the destination path has been resolute and kept in a cache, ARP or ICMP_ECHO REQUEST packets are retransmitted up to 3 times before an appropriate error message is finally returned. The user may stop the ping command at any time using Ctrl-C.

Command	Description/Syntax
port	port [-a] [-8|-16|-32] <address> [<value>] Perform a read or write operation to the specified <address> with the specified data width (default 32-bits). • If <value> is specified, this value is written, otherwise a read operation is performed and the result is displayed. The command checks the validity of the specified address. This check can be turned off using the '-a' option. The port command will result in exactly one read or write operation with the specified data width. This makes it useful for accessing registers in peripheral devices. -8 Access data byte -16 Access data halfword -32 Access data word -a Allow invalid addresses
search	search [-asc|-hex] <address> <size> <string> Search for string in the memory area specified by <address> and <size>. Default string type is ASCII. If the search string contains spaces, remember to use quotes around the string. If searching for a hex string, the search pattern must be entered as a number of two-digit hexcodes without spaces in between. -asc Search for ascii string -hex Search for hex string
setenv	setenv [<variable> [<value>]] Set the specified environment variable. If no variable is specified, all environment variables are displayed. If no value is specified, the variable is set to the NULL value. When setting a R/W system variable, the value is first validated.
sleep	sleep <ms> sleep - Halt shell for the specified number of milliseconds. **NOTE** The default number format is hexadecimal.
ss	ss PLL_MULTIPLIER [<SYSTEM_BUS_DIVIDER>] ss - Set Speed of processor and optionally change bus multiplier. Normal settings are for a PLL_MULTIPLIER value of 33 and a SYSTEM_BUS_DIVIDER value of 4.
stty	stty [-tty0|-tty1|-tty2] [-b|-u| [-p][<baudrate>][n|o|e][7|8][1|2]] Setup or view serial port setup. Default port is tty2 -b,-u,-p apply to the default port if no port is specified. The possible baud rates are generally 75-460800. Use 'stty –ttyx -b' to get a list of the supported baud rates for a specific port. Available parity settings are n (none), o (odd), e (even). Available data bits are 7 and 8. Available stop bits are 1 and 2. When changing the parameters for a tty which is being used (e.g. the console), some strange characters may appear as a result. Also note that stty does not by default modify the semi-permanent tty setting recorded in the environment variables. Use the '-p' option if you want to set the environment variable for the specific tty as well. -u Force environment settings for port to take effect -p Transfer current settings for port to environment -b List supported baud rates for the specified port -tty0 Setup port 0 -tty1 Setup port 1 -tty2 Setup port 2 - default

Command	Description/Syntax
test	test [-l] | [-m] [<module> [<module arguments>]] test - The test command can perform a number of self-tests on different modules. • If no module is supplied, all available modules are tested and a final pass/fail status is indicated. • If a module is specified, only this module is tested. • If the option '-m' is applied and no module is specified, a list of the available modules is displayed. • If the option '-m' is applied and a module is specified, additional information about the module test and the optional arguments is displayed. • If the option '-l' is applied, all available modules are tested repetitively, until Ctrl-C is pressed or a test fails. The '-l' option cannot be specified together with other options or arguments. -m List available test modules -l Loop default tests until Ctrl-C is pressed
tlb	tlb -i | [<index> <pagesize> <va> <g> <asid> <pa0> <c0> <d0> <v0> <pa1> <c1> <d1> <v1>] tlb - Display or edit TLB. In case there are no parameters, the contents of the TLB is displayed. In case (all) parameters are available, the TLB entry at the requested index is written. There are 32 TLB entries. Available settings of 'pagesize' are : 0x1000 | 4kB 0x4000 | 16kB 0x10000 | 64kB 0x40000 | 256kB 0x100000 | 1MB 0x400000 | 4MB 0x1000000 | 16MB Available settings of c0/c1 (cache algorithm for even/odd page) are processor specific. However, the values 2 and 3 are typically reserved for Uncached (2) and Cacheable (3) modes. Values 0..7 are available. -i Initialise TLB Other parameters are : va : Virtual base address of even/odd pair of pages. g : GLOBAL setting ('n' -> ASID is used, 'y' -> Ignore ASID). asid : ASID setting (only relevant if g = 'n'). pa0 : Physical base address of even page. d0 : DIRTY setting of even page ('y' -> write enabled, 'n' -> write protected). v0 : VALID setting of even page ('y' -> valid, 'n' -> not valid). pa1 : Physical base address of odd page. d1 : DIRTY setting of odd page ('y' -> write enabled, 'n' -> write protected). v1 : VALID setting of odd page ('y' -> valid, 'n' -> not valid). **CAUTION** Do not modify entries 0, 1, or 2 in the TLB table. These are reserved for PCI address spaces. Refer to *TLB Configuration* under *Programming Considerations* later in this chapter. *See **Example** next page -*

Command	Description/Syntax
	Example : TLB index = 2 Pagesize = 4kB Global mapping (i.e. ASID ignored) ASID = 0xff (but ignored) Cache algorithm = 3 (Cacheable) Both pages valid Virtual address Physical address Dirty (i.e. write enabled) --- 0x00000000 0x00200000 Yes 0x00001000 0x00300000 No tlb 2 4kB 0 y ff 200000 3 y y 300000 3 n y
unsetenv	unsetenv <variable> | (-u|-s)+ unsetenv - Unset specified environment variable. If a user created variable is specified, it will be removed from the environment. A system variable will not be removed, but will instead be set to the default value. By using appropriate options, all user and/or system variables can be unset using a single command. -u Delete all user variables. -s Reset all read/write (R/W) system variables to default values

Operating System Support

The EnCore M3 EDK CD-ROM Set comes with support for popular operating systems used on the MIPS32 platform, such as VxWorks by Wind River and TimeSys Linux. Ampro provides a VxWorks Board Support Package (BSP) required to run the operating system on the EnCore M3 and a full TimeSys Linux GPL distribution. Refer to the ReadMe file on the EDK CD-ROM for a list of the other supported OS's, or view the Ampro web site.

NOTE	The TimeSys Linux BSP comes with a “Getting Started Guide” on TimeSys Linux Distribution CD-ROM. Refer to this Guide for more information.

VxWorks

The Ampro distribution contains a Board Support Package (BSP) for the Tornado for MIPS32 platform environment. The following major functions are supported;

- Serial Console and support for virtual console via WDB
- Ethernet interface with full network protocol support
- IDE Hard Disk, CD-ROM, and DVD drivers
- Floppy drivers
- Support for Ethernet debug through WDB
- Boot Line stored in on-board Flash memory (via YAMON)
- Automatic detection of memory size from MCCR registers via SDRAM controller settings

Development Environment

Development for VxWorks requires obtaining the needed development tools from Wind River, which is called “Tornado”. The Ampro BSP is configured for Tornado in the MIPS processor environment, specifically for a MIPS32 based processor system. The Tornado environment runs on either a UNIX workstation or on a Windows-based machine. The typical development environment described earlier also applies to Tornado environment.

Ampro does not distribute the Tornado development tools. If you are interested, please contact your Wind River representative.

BSP Configuration and Installation

The Ampro BSP comes fully configured. To install the Ampro BSP, you must first install the Tornado for the MIPS32 environment. Once this is complete, running the setup program on the Ampro EDK under the BSP directory will automatically install the BSP into your development environment. The BSP will appear as a project platform where a development project can be built.

Programming Considerations

The following items should be considered when writing programs for the EnCore M3 platform.

Interrupt Configuration

The system interrupt configuration yields up to 80 interrupt sources. These are split between the two interrupt controllers in the Au1500™ and the PIC located in the VIA Southbridge. A diagram of the interrupt control mechanism is shown in the figure below.

NOTE	The PIC interrupts are cascaded through the Au1500™ interrupts.

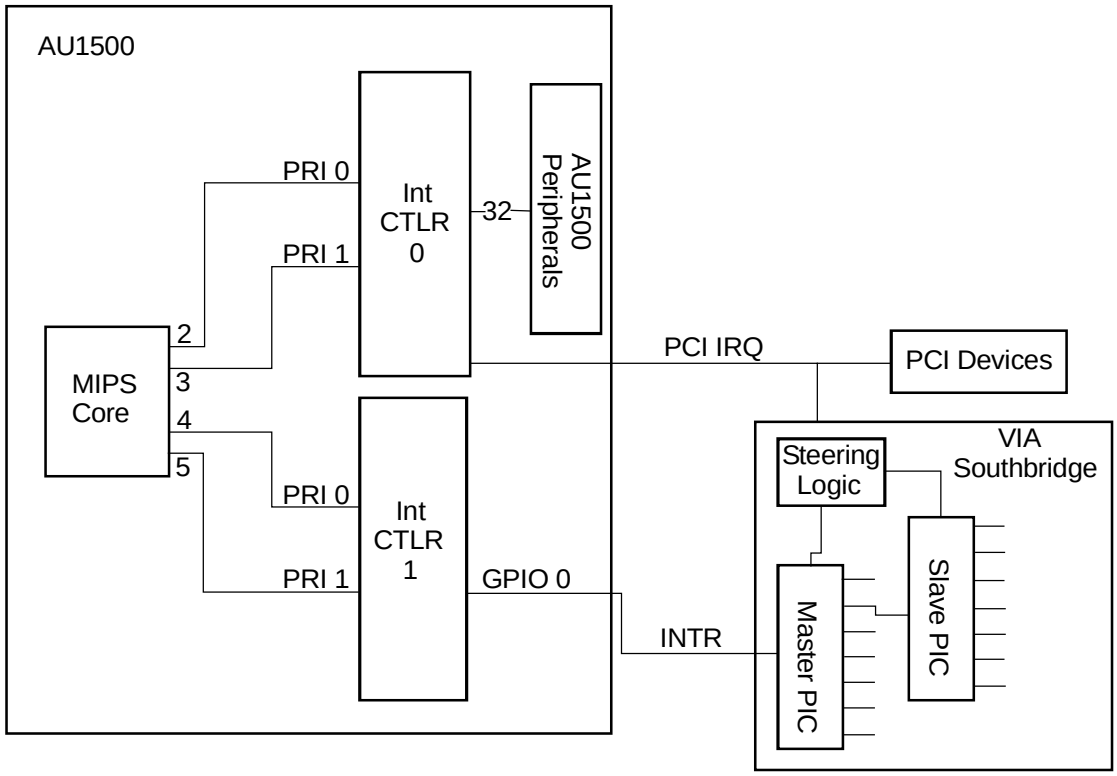

Figure 4-2. Interrupt Structure

PCI interrupts can be routed via the PIC, or can be directly processed. It is recommended that all PCI interrupts be handled directly, thus the VIA PCI interrupt steering registers will be left disabled. It should also be noted that most of the interrupts that come in via GPIO are not used in the M3 module. Many GPIO lines are used for such functions as USB interface and serial ports. Some operating systems will count these as IRQ numbers , since these are sources on the interrupt controller. This means that even though there are 80 IRQ numbers available, only those from 4 to 10 of the 80 interrupts are effectively useable in the M3 module at any one time.

PCI Configuration

The PCI interface is contained in the Au1500 processor, which provides standard PCI signals to the PCI connector and the VIA Southbridge. This device is configured as a host, and provides full PCI capability for the host memory access.

PCI Memory Map

The following table outlines the memory map associated with the PCI bus. The first 3 entries (0, 1, 2) in the TLB table are set aside for this purpose and should not be altered. To set up the first three entries, the CP0 Wired register is set to 3 to avoid erasing these entries while maintaining cache coherency. Refer to *TLB Configuration* later in this chapter.

Table 4-3. PCI Bus Memory Map Table

Physical	Mapped to 32-bit Address	Description
4 0000 0000	6000 0000	PCI Memory Space – Refer to the AMD Alchemy Au1500 data sheets or specifications under the PCI section.
5 0000 0000	5000 0000	I/O Space address register. Used to access PCI I/O space for devices that require I/O access.
6 0000 0000	4000 0000	Configuration Space Registers. All config space registers are mapped into this region based on an addressing scheme described below.

Config Space Access

The Au1500 PCI controller provides access to PCI configuration space via an address block at physical address 0x6 0000 0000. The default configuration space is implemented as 36-bit address space, but must be remapped into 32-bit address space in system memory using a TLB entry. The Au1500 PCI controller can generate both Type 0 and Type 1 configuration cycles. Type 0 cycles are used to access devices on the same PCI bus as the host controller, while Type 1 cycles are used to access devices on PCI busses that are on the other side of PCI-PCI bridges. Both types of cycles are generated by accessing the appropriate 32-bit address in the PCI configuration space as mapped by the TLB entry.

When mapping configuration space using TLBs, the user must be aware of which space to map. Bit 31 of the 36 bit physical configuration address space is used to select type 1 configuration cycles. This is normally out of the range of a single TLBs contiguous address range, which has a maximum of 32MB. There are two ways to implement the TLB mappings.

- First method – Give the configuration space read/write routines a random TLB entry, and configure it for the address range desired every time the configuration read/write function is called. This will slow down configuration operations, but these types of operations are generally only done during initialization.
- Second method – Generate a non-contiguous TLB entry, with the first section set to the type 0 configuration space, and the second set at the type 1 configuration space. The values for this TLB entry would be:

```
Page Mask  0x01ffe000  (16 Mbytes)
Entry Hi:  0x40000000  (Mapped to 0x4000 0000)
Entry Lo0: 0x1800003f  (Maps 0x6 0000 0000 to 0x4000 0000)
Entry Lo1: 0x1A00003f  (Maps 0x6 8000 0000 to 0x4100 0000)
```

However, using this second method does not allow the user to use type 0 configuration cycles to access devices that use address lines 24-31 as IDSEL lines. All type 1 configuration addresses could be accessed. The EnCore M3 uses address lines 16, 17, 18, and 19 as the IDSEL lines 0, 1, 2, 3, respectively, on the EnCore PCI interface. Conceivably, other address lines could be used on your baseboard design, so the preceding caveat should be considered.

NOTE	You can map the configuration space to any 32 bit unused address. The address 0x4000 0000 was chosen simply because it was an unused address range on the EnCore M3 system.

Type 0 configuration cycles

Type 0 configuration cycles are created by generating a 32-bit address in the following form

Table 4-4. Type 0 Configuration Cycle Addresses and Descriptions

Address Bits	Description
31:28	0100
27:11	IDSEL bits – This identifies a specific PCI device
10:8	F (Function Select) – This specifies a function
7:2	Reg (Register select) – This identifies the upper 6 bits of the register number
1:0	00 – Type 0 config cycle

Where the IDSEL bit identifies a specific PCI device, the F bits specify a function, and the Reg bits identify the upper 6 bits of the register number. A Type 0 configuration cycle is identified by having the lower two bits set to 00. This implies that registers can only be accessed in 32 bit increments, thus byte access must be implemented in software by performing shift and mask operations on the 32 bit word. Only one IDSEL bit can be active at one time, since a specific IDSEL line will correspond to a specific device on the PCI bus.

Here is an example of creating a Type 0 configuration access. Consider a device on the primary PCI bus that is connected to AD15 as it's IDSEL line. This would be device ID 5 (where device ID 0 corresponds to the host itself). To access register 0x10 in function 1, the effective address in the PCI configuration space would be:

```
0x4008110
```

To generate configuration read cycles, simply read from the example address. Conversely, to create Type 0 configuration write cycles, simply write to the example address.

Type 1 Configuration Cycles

Type 1 configuration cycles are created in the same manner, using the following format;

Table 4-5. Type 1 Configuration Cycle Addresses and Descriptions

Address Bits	Description
31:24	01000001
23:16	Bus – This identifies a specific bus number
15:11	Device – This identifies a specific device
10:8	F (Function Select) – This specifies a function
7:2	Reg (Register select) – This identifies the upper 6 bits of the register number
1:0	00 – Type 1 config cycle

Using the single TLB entry method as previously described, a Type 1 configuration cycle can be generated using the above address formats. The value 0x41 in bits 31:24 map to the type 1 configuration space. All other address bits can be generated using the standard type 1 address format. However, bits 1:0 must be set to 0x00, to keep the 32 bit addressing aligned correctly.

By selecting the address space mapped to the physical address space 0x6 8000 0000, bits 1:0 will be automatically set to 01 on the PCI bus, indicating a type 1 configuration cycle.

NOTE	You must remember bits 1:0 must be set to 0x00, to keep the 32 bit addressing aligned correctly.

Here is an example of generating a Type 1 cycle. Consider a PCI device number 6 on bus number 3. We would like to access register 0x3c on function 0. The address generated would be:

```
0x4103303c
```

Reading or writing to this address generates Type 1 configuration cycles.

TLB Configuration

In the MIPS processing environment, the virtual-to-physical memory mapping is performed by the Memory Management Unit (MMU). The primary controlling mechanism for the MMU is the Translation Lookaside Buffer (TLB) built into the C0 co-processor. In the EnCore M3, this is a 32 entry table that defines the physical block to virtual address allocations. There are many books and data sheets that provide an extensive discussion of TLB mapping, which is beyond the scope of this text. Ampro highly recommends obtaining these materials for more information about TLB mapping.

In the Au1500 processor, the various PCI address spaces of the integrated PCI controller are assigned to the extended 36-bit memory space at locations 0x400000000, 0x500000000, and 0x600000000 respectively. In order for the processor to access these spaces, these must be mapped into the 32-bit virtual memory space by TLB entries. The YAMON monitor on the M3 processor will map the spaces to the virtual (32-bit) memory spaces, 0x40000000, 0x50000000, and 0x60000000, respectively. This is done automatically, without user code or intervention required.

Many operating systems make extensive use of the TLB for process management and protection. Processes can be assigned specific physical memory spaces, but run as if these spaces are all linked at the same location. This requires the operating system to manage TLB entries to keep the current set of memory mappings correct. If the operating system swaps out the TLB entries for the PCI controller, access to the PCI address spaces are disrupted. This could have dire consequences if, for example, a PCI Bus Master DMA from the processor to a device is in progress. Thus, the operating system must be informed not to touch the PCI address mappings.

In the MIPS32 processor architecture, the C0 processor provides a randomized index value for TLB entries to replace. This provides a method for randomly choosing which TLB entries to swap out when a new entry is required. To avoid choosing the PCI address entries, the WIRED register can be used to cause the random index register to choose values above the protected indices. The value the WIRED register (in the C0 processor) is set to provides the minimum value for the random index value. Thus, since PCI address space TLB entries are set into TLB entries 0, 1, and 2, setting the WIRED register to 3 will effectively protect them from random TLB updates.

If the user writes an application that uses the MMU directly, then protecting the PCI address space TLB entries must be taken into account. Do not update entries 0, 1, or 2, since these are reserved for PCI address spaces.

CAUTION	Do not modify entries 0, 1, or 2 in the TLB table. These are reserved for PCI address spaces. Changing these entries could render your EnCore M3 system inoperable.

Floppy/Parallel Configuration

The EnCore M3 contains a standard VIA 82C686B Southbridge, which provides several I/O functions not normally found with MIPS processors. One of these configurations is the Floppy/Parallel controller. In the M3 implementation, these two functions share the same pins on the EnCore interface. Thus, a user cannot use the floppy and parallel interfaces at the same time. In order to select the proper function, one must configure the VIA to the proper function. The following assembly code shows how to perform this function.

Once selected, the function can be accessed at the appropriate I/O location. There is no need to re-program the address locations for each function – they are pre-programmed at system initialization by YAMON.

The following code description shows how to modify the configuration of the Super I/O controller to select Floppy functions. The register definitions are shown in code comments – to select parallel port operations ensure that bit 5 in register 0xf6 is set to 1, and that the parallel port is enabled to the proper mode in register 0xe2.

```
/*************************************************************
 *                            pci_config_access
 *  Description :
 *  -------------
 *  PCI configuration cycle (read or write)
 *
 *  Return values :
 *  ---------------
 *  OK    : If no error.
 *  ERROR_PCI_ABORT : If master abort (no target) or target
 *  abort.
 *

*************************************************************/

UINT32 pci_config_access(
    UINT32 busnum,     /* PCI bus number (0 = local bus)   */
    UINT32 devnum,     /* PCI device number    */
    UINT32 func, /* Function number of device  */
    UINT32 reg,  /* Device register      */
    bool   write,      /* TRUE -> Config write, else read  */
    UINT8  size, /* Sizeof data (1/2/4 bytes)  */
    void   *data )     /* write or read data   */
{
    UINT32 intr;
    UINT32 data32, align, pos;
    UINT32 *addr;
    UINT32 rc;

    /* Always perform 32 bit access */
    if( size != sizeof(UINT32) )
    {
        /* Calc 32 bit aligned word */
        align = reg & ~MSK(2);

        /* Read word */
        /* RECURSIVE CALL */
        rc = pci_config_access(
            busnum, devnum, func, align,
            FALSE, sizeof(UINT32), (void *)&data32 );

        if( rc != OK )
            return rc;

        switch( size )
        {
        case sizeof(UINT8)  :

            /*  Calc position of byte within word */
            /*  (PCI is always little endian)      */
            pos = (reg & MSK(2)) * 8;

            if( write )
            {
```

```
                /* Modify word */
                data32 &= ~(MSK(8) << pos);
                data32 |= *(UINT8 *)data << pos;
            }
            else
            {
                /* Calc byte */
                *(UINT8 *)data = (data32 >> pos) & 0xFF;
            }

            break;
        case sizeof(UINT16) :

            /*  Calc position of halfword within word */
            /*  (PCI is always little endian)         */
            pos = (reg & (MSK(1) << 1)) * 8;

            if( write )
            {
                /* Modify word */
                data32 &= ~(MSK(16) << pos);
                data32 |= *(UINT16 *)data << pos;
            }
            else
            {
                /* Calc halfword */
                *(UINT16 *)data = (data32 >> pos) & 0xFFFF;
            }

            break;
        default : /* Should not happen */
            break;
        }

        if(write)
        {
            /* Write the modified word */
            rc = pci_config_access(
                busnum, devnum, func, align,
                TRUE, sizeof(UINT32), (void *)&data32 );
        }

        return rc;
    }
    else
    {
        /*
        ** Simple mapping of register into PCI register space
        ** 1) Calculate address
        ** 2) Read/write 32 bits
        ** 3) Check for master or target abort
        ** DONE
        */

        addr =  (UINT32 *)   (M3_PCI_CONFIG_BASE     |
            (1 << (devnum + 15))                     |
            (func<<8)                                |
            reg);
```

```
            if( write )
            {
                *(addr) = *(UINT32 *)data;
            }
            else
            {
                *(UINT32 *)data = *(addr);
            }
        }

        return OK;
    }

    /******
    ** setSIO
    ** This routine will set the proper value into the VIA
    configuration register  */

    void setSIO(UINT8 port,UINT8 val)
    {
        IODelay();
        _outb(0x3f0,port);
        IODelay();
        _outb(0x3f1,val);
    }

    /******
    ** getSIO
    ** This routine will get the value from the register/

    UINT8 getSIO(UINT8 port)
    {
        IODelay();
        _outb(0x3f0,port);
        IODelay();
        return (_inb(0x3f1));

    }

    /*****************************************************************
    **  FUNCTION: SIOConfigure
    **
    **  PARAMETERS:
    **
    **  DESCRIPTION: This routine is responsible for setting up the **
    **  south bridge super I/O controller.  Specifically, it selects
    **  between parallel mode or floppy mode for the SIO
    **  configuration.
    **
    **
    **  RETURNS:
    **
    *****************************************************************/

    void SIOConfigure(void)
    {
        UINT8   SIOconfig;
        UINT32  result;
        UINT8   regVal;
```

```
/*
** Steps
**  1) Enable the configuration of the SIO
**  2) Set registers
**  3) Exit configuration
*/

/*
** Step 1
** Enable Super I/O configuration via device 2, function 0,
   register 0x85 */

result = pci_config_access(0,5,0,0x85,0,1,&SIOconfig);
SIOconfig |= 0x0b;
result = pci_config_access(0,5,0,0x85,1,1,&SIOconfig);

/*
** Step 2
** Set the registers
**
** Register 0xe2  Super I/O Function Select
**   BITS 7-5   Reserved
**          4   Floppy Controller Enable (1 = enable)
**          3   Serial Port 2 Enable     (1 = enable)
**          2   Serial Port 1 Enable     (1 = enable)
**        1-0   Parallel Port mode/Enable
**              00  - Unidirectional
**              01  - ECP
**              10  - EPP
**              11  - Disabled
**
** Register 0xe3  Floppy Controller I/O base address
**   BITS 7-2   I/O address bits 9-4 for the controller
**        1-0   MUST BE ZERO
**
** Register 0xe6  Parallel Port I/O Base Address
**   BITS 7-0   I/O address bits 9-2 (must be 4 byte aligned)
**
** Register 0xf0  Parallel Port Control
**   BITS   7   PS2 Type Bidirectional Enable (1 = enable)
**          6   EPP Direction by Register (1 = enable)
**          5   EPP+ECP  (1 = enable)
**          4   EPP Version (0 = 1.9, 1 = 1.7)
**        3-0   Reserved
**
** Register 0xf6 Floppy Controller Configuration
**   BITS 7-6   Reserved
**          5   FDC Mode (1 = FDC on FDC pins, 0 = FDC on
**               Parallel port, required for M3)
**          4   3-D mode (1 = enable)
**          3   Reserved
**          2   Four Floppy Option (1 = Enable)
**          1   FDC DMA Non-Burst (0 = burst, 1 = non-burst)
**          0   FDC Swap (1 = enable)
**
** Register 0xf8 Floppy Drive Configuration
**   Bits 7-6   Drive 3 config
**        5-4   Drive 2 config
```

```
        **        3-2   Drive 1 config
        **        1-0   Drive 0 config
        **
        **  The following code selects Floppy drive on the Parallel
        **  port pins, and sets all drives to the normal
        **  configuration */

        setSIO(0xe2,0x1c);
        setSIO(0xe3,0xFC);
        setSIO(0xf6,0x21);

        /*
        ** For each of the floppy drives, select drive type
        ** 00 = 1.2 MB 5" or 1.44 MB 3"
        ** 01 = 360 KB 5"
        ** 10 = 720 KB 3"
        ** 11 = undefined (1mbit)
        **
        ** Do this for all "4" drives: ddccbbaa
        */

        setSIO(0xf8,0x0);

        /*
        ** Step 3
        ** Set back to normal mode
        */

        result = pci_config_access(0,5,0,0x85,0,1,&SIOconfig);
                 SIOconfig &= 0xfd;
        result = pci_config_access(0,5,0,0x85,1,1,&SIOconfig);

        return
    }
```

Processor Speed Control

In order to control the speed of the processor, the user must configure the PLL clock and the peripheral bus divider count to the appropriate value. This is done on initialization by YAMON, and can be changed using the SS command. The following “C” code function is an example of how to change the speed programmatically.

```
/*************************************************************
**  FUNCTION: freq_init
**
**  PARAMETERS: NA
**
**  DESCRIPTION: This function initializes the PLL and the bus
**  clock divider to the selected frequencies.
**
**  RETURNS: NA
**
*************************************************************/

void freq_init(int pll_mult,int bus_mult)
{
    UINT32  old_ie;
    UINT32  divide;
    UINT32  brgdiv;
```

```
        UINT32  pll_mult;
        UINT32  cpu_speed;
        UINT32  cpu_test;
        UINT32  bus_mult;
        UINT32  bus_mult_old;

        /*
        ** Code Begins
        ** The following code was lifted from ss.c, and correctly
        ** sets the desired parameters
        */

        if(pll_mult > 33 || pll_mult < 16)
        {
            printf("Invalid Multiplier Setting\n");
            return
        }
        cpu_speed = pll_mult * 12000000;
        printf("CPU -> %d MHZ \nSystem Bus Divider -> %d
        \n",cpu_speed / 1000000, bus_mult); brgdiv = (cpu_speed
        / (bus_mult * 2 * 16 * 115200));
         REG32(Au1500_CPU_PLL) = pll_mult; /* Reg at 0xB1900060  */
         REG32(Au1500_POWERUP_CTRL) = bus_mult - 2;
                                        /* Reg at 0xB190003c  */
         REG32(UART0_BASE + SERIAL_Au1500_UCD_OFS) = brgdiv;
    /* Reg at 0xb1100028  */
    }
```

NOTE	The speed calculation is based on the cpu_speed calculation. To compute the PLL multiplier value, you can use the following formula:

```
Pll_multiplier = cpu_speed (Hz)/12000000
```

The practical limits for the CPU speed are 192MHz on the low end (PLL multiplier 16) and 396MHz on the high end (PLL multiplier of 33).

CAUTION	If you choose to operate the EnCore M3 module outside of these ranges, Ampro, Computers, Inc. can not guarantee proper operation of the EnCore M3 module.

It is also important to remember that changing the bus multiplier for the peripheral bus has a direct impact on the baud rate generators for the serial ports. Ensure all serial ports used have the respective baud rate registers re-calculated when the bus multiplier changes.

Flash Management

Managing the contents of the FLASH device can be a tricky undertaking. The flash is divided into 6 main sections and each of these areas has specific contents and use.

1. Monitor Flash area
2. System flash area
3. Environmental flash area
4. Boot Line area
5. NVRAM area
6. Parameter flash area

Table 4-6. Flash Memory Sections and Descriptions

Section	Description
Monitor Flash	This area contains the executable code for the YAMON monitor. Most of the time you will only change the Big or Little Endian version of the YAMON code, leaving the reset vector code as is.
System Flash	This area contains user data, and can be used for any purpose.
Environmental Flash	This area contains the environmental variables used by YAMON.
Boot Line Flash	This is a single 8K sector in the flash that is normally used to store an Operating System boot line, or other bootup information.
NVRAM Flash	This is a single 8K sector that the user can use to store any parameter information for application use. The OS support packages have access routines that use this area for Non-Volatile storage
Parameter Flash	This is a protected area of Flash, and should not be modified. It contains the board parameters such as serial number, MAC address, and other board specific information.

In order to program any of these areas the user must first erase the data that exists there. The YAMON erase command can be used for this function.

CAUTION	DO NOT ERASE CRITICAL FLASH AREAS UNLESS YOU ARE SURE YOU HAVE THE PROPER DATA TO INSERT INTO THESE AREAS. THE MODULE MAY NOT BOOT UP IF CRITICAL AREAS REMAIN ERASED.

Simple procedure for erasing the monitor Flash:

```
YAMON> erase bfc00000 4000

YAMON> erase bfc10000 F0000

YAMON> load
```

Once the area is erased, the YAMON load command can be used to load the appropriate S-Record file into the flash. The YAMON monitor consists of three major components; the Reset code, the Big Endian image, and the Little Endian image. The Reset code is that which is executed upon startup or reset exception of the processor. This code will perform minimal configuration, determine the endianness selected, and jump to the proper endian image. The YAMON image (Big or Little Endian) will then perform further initialization, copy itself to RAM, and start system execution.

When making modifications or extensions to YAMON, you must ensure the changes that you make do not cause the system to come up in an unusable state. This would cause the system to become inoperable, and would not allow you to re-program the flash with another image, except with a EJTAG debugging tool. If you are making changes to YAMON, it is highly recommended that you have just such a tool available.

Changes to the other flash areas are more innocuous, and can be undone relatively easily. Using the YAMON “erase –e” command, for example, will erase all of the Environmental Flash and set up the default parameters. You could then re-load any other variables using the “set” command. Since YAMON does not use the System, BootLine, or NVRAM parameters, these can be changed per user requirements.

The following table outlines the memory map for the flash device:

Table 4-7. Flash Memory Map

Sys Address	Flash Address	Sectors	Description
bfC00000 – bfC03FFF	00000000 – 00003FFF	0	Reset Image (16kB)
bfC04000 – bfC05FFF	00004000 – 00005FFF	1	Boot Line (8kB)
bfC06000 – bfC07FFF	00006000 – 00007FFF	2	Parameter Flash (8kB)
bfC08000 – bfC0FFFF	00008000 – 0000FFFF	3	User NVRAM (32kB)
bfC10000 – bfC8FFFF	00010000 – 0008FFFF	4-11	YAMON Little Endian (512kB)
bfC90000 – bfD0FFFF	00090000 – 0010FFFF	12-19	YAMON Big Endian (512kB)
bfD10000 – bfDEFFFF	00110000 – 001EFFFF	20-33	System Flash (896kB)
bfDF0000 – bfDFFFFF	001F0000 – 001FFFFF	34	Environmental Flash (64kB)

The AMD29F160DB device has a feature called the “boot sector.” In this device, the first 4 sectors are smaller than the baseline 64kB per sector. Sector 0 is 16kB, sectors 1 and 2 are 8kB, and sector 3 is 32kB. This allows for smaller sections of data to be stored without having to manage a larger 64kB sector. This is where the boot line, NVRAM, and parameter areas are selected. When performing a sector erase, ensure you take these last 4 sectors into account.

CAUTION Do not programmatically perform a “entire device” erase. This will erase the parameter storage area, causing the MAC addresses and serial numbers to be lost.

GPIO Jumper (W5)

This jumper allows the user to set GPIO 200 and GPIO 201 to a specific state. The user can read the state of these pins by looking at the register. The state of these pins can be read at register location: 0xb170000C, bits 0 and 1. A value of 1 indicates high, and a value of 0 indicates low. For more information refer to the AMD Alchemy Au1500 data sheet, section 8.3.

Appendix A Technical Support

Ampro Computers, Inc. provides a number of methods for contacting Technical Support listed in the Table A-1 below. Ampro provides a comprehensive listing of Frequently Asked Questions on our web site at the Virtual Technician. If you can not find the answers to your questions, please continue in the Virtual Technician and ask for Personal Assistance. Requests for support through the web site are given the highest priority, and usually will be addressed within one working day.

- **Internet** – Provides the most information concerning Ampro products, including reference material and white papers.
  - Ampro Virtual Technician – This service is free and available 24 hours a day through the Ampro Computers World Wide Web site at http://www.ampro.com. However, you must sign in to access this service.

    The Ampro Virtual Technician is a searchable database of Frequently Asked Questions, which will help you with the common questions asked by most customers. This is good source of information to look at first for your technical solutions.
  - Embedded Design Resource Center – This service is also free and available 24 hours a day at the Ampro web site at http://www.ampro.com. However, you must sign in to access this service.

    The Embedded Design Resource Center was created as a resource for embedded system developers to share Ampro's knowledge, insight, and expertise gained from years of experience. This page contains links to White Papers, Specifications, and additional technical information.
- **Personal Assistance** – This is the quickest way to obtain a response to your support questions. Please go to the following location on Ampro's web site to submit your request 24 hours a day, 7 days a week.

Table A-1. USA Technical Support Contact Information

Method	Contact Information
Web Site	http://www.ampro.com
E-mail	support@ampro.com
Standard Mail	Ampro Computers, Incorporated, 5215 Hellyer Avenue, San Jose, CA 95138-1007, USA

Appendix B Special Interface Cables

The cables listed in this Appendix are used with the EnCore M3 module and are not typically commercially available in these configurations.

The cable in Figure B-1 is used as the interface between the Ethernet port (J14) on the EnCore M3 module and an Ethernet Port 2 connector (RJ45).

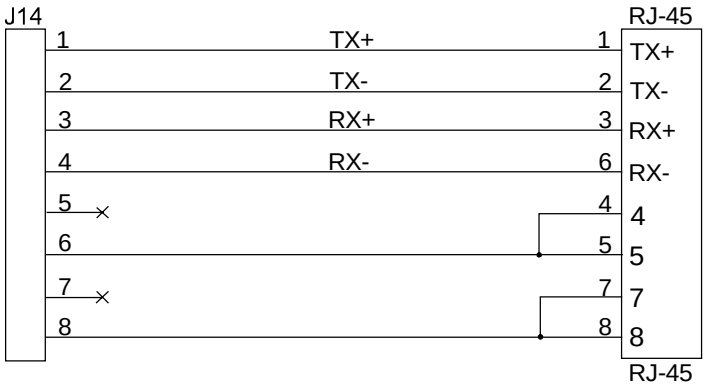

Figure B-1. Ethernet Port 2 Interface Cable

The cable in Figure B-2 is used as the interface between the Serial Debug port (J15) on the EnCore M3 module and a common DB-9 connector used for debugging.

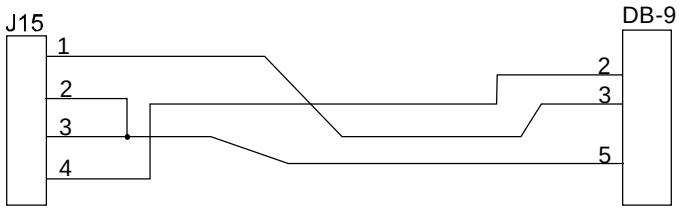

Figure B-2. Serial Debug Port Cable

Index

Ampro
  bootloader provided 17
Ampro Products
  CoreModule Family 3
  EnCore 400 3
  EnCore 500 3
  EnCore 700 3
  EnCore PP1 2
  Little Board Family 3
  MiniModule Family 3
Ampro web site
  Embedded Design Resource Center 1
  up-to-date OS support list 14
Audio controller
  requires external drive circuit 29
baud rate generator
  serial ports affected 57
bootloader
  configure floppy disk drive 25
  software configure 25
bus signal timing
  not provided 1
cache
  CPU 16
CD-ROM support
  EnCore Development Kit (EDK) 2
chip specifications, detailed
  web sites 2
clock speed
  power consumption 16
CPU
  Alchemy Semiconductor's Au1500™ 16
  cache 16
  implemented features 16
  peripheral bus 16
  power-saving modes 17
  system bus 16
CPU choices, EnCore
  Intel processors 5
  low power RISC processors 5
  Motorola processors 5
CPU clock
  halted in sleep mode 16
CPU clock speed
  software configurable 16
  variable to 400MHz 16
  programmable 57
CPU peripheral bus
  peripheral module 16
CPU power dissipation
  < 500mW @400MHz 16
CPU system bus
  system module 16
current requirements
  EnCore M3 module 13
custom logic board
  write driver software 14
Cygwin
  Windows-based environment 35
detailed chip specification
  not provided 1
development tools
  gnu 35
EJTAG
  features 33
Embedded Design Resource Center
  Ampro web site 1
embedded processor
  provides more choices 5
embedded system
  EnCore M3 module 1
EnCore Architecture Specification 1
EnCore baseboard
  ATX power supply connection 2
  audio connections 2
  floppy drive connection 2
  IDE CD-ROM 2
  IDE hard drive 2
  keyboard connection 2
  mouse connection 2
  parallel printer port 2
  serial ports 2
EnCore Concept
  custom baseboard 5
  easy upgrade path 5
  industry-standard I/O signals 5
  pre-tested processor module 5
EnCore Development Kit (EDK) 2
  Allegro layout files 2
  Board Support Packages (BSP) 2
  CD-ROM support 2
  development environment 2
  EnCore M3 operating system 2
  Monitor 2
  Monitor and Support package 2
  operating system 2
  OrCad schematic files 2
  peripheral drivers 2
  start-up code 2
EnCore Development Kit CD-ROM 2
EnCore EBX baseboard
  drawings 1
  gold board comparison 2
  parts list 1
  reference design manual 1
  schematics 1
EnCore M3 module
  100 x 145 mm format 6
  2MB Flash memory 6
  400MHz 6

480-Drystone MIPS 6
AC97 audio interface 6
audio controller 29
block diagram 8
CPU 400MHz 16
CPU Core current 13
CPU core voltage requirements 13
CPU features 7
current 13
ECP/EPP bi-directional parallel port 6
EJTAG feature 7
embedded processor 5, 6
EnCore platform series 6
enhanced Ultra 33/66 Synchronous DMA 6
environmental limitations 14
floppy disk controller 6
high performance 6
high-integration MIPS32-compatible 6
I/O features 7
IDE features 7
infrared port (IrDA) 29
IrDA port 6
keyboard and mouse 30
LED indicators 8
less than 2.5 watts 6
low power 6
major integrated circuits (chips) 9
mate to custom logic board 6
MIPS32™ instruction set (CPU) 16
memory 17
memory features 7
module connectors 1
network interface features 7
operating systems 35
PCI Bus features 7
physical dimensions 12
pin numbers 1
power and voltage 8
power dissipation 16
power use 13
programming considerations 1
PS/2 keyboard and mouse 6
Real Time Clock 7
signal definitions 1
special Ethernet cable 30
specifications 12
software 1
supported CPU frequencies 13
two 10/100BaseT Ethernet ports 6
two IDE drives 6
two serial ports 6
two USB ports 6
up to 256MB SODIMM 6
Utility interface features 7
voltage requirements 13
EnCore M3 QSK
QuickStart Kit (QSK) 2
EnCore modules
Intel architecture (x86) 5
MIPS processor 5
PowerPC processors 5
EnCore software support
operating system support packages 35
YAMON monitor source 35
utilities 35
EnCore software support
development environment 35
environmental limitations
EnCore M3 module 14
environmental requirements 1
Ethernet (LAN) Interface
features 30
Ethernet port 1
baseboard connector 30
Ethernet port 2
LEDs 12, 34
M3 connector with cable 30
special cable 30, 63
flash erased
re-program 58
YAMON load command 58
flash memory
2MB, 16-bit wide 17
boot sector 59
Floppy Disk Controller
features 27
Floppy Drive configuration
programmable 51
gnu development tools 35
gold board comparison
EnCore EBX baseboard 2
GPIO 200
jumper settings 11, 59
GPIO 201
jumper settings 11, 59
GPIO 202
LEDs 12, 34
GPIO 203
LEDs 12, 34
GPIO interrupts
not all used 49
GPIO Jumper (W5) 59
GPIO 200 59
GPIO 201 59
I/O interface signals
three connectors 22
IDE interface
features 23
input and output codes
table legend 22
internal registers
not provided 1
Jumper (W5)
GPIO 201 59

jumper settings
GPIO 200 ..... 11, 59
GPIO 201 ..... 11, 59
LEDs
Ethernet Port #2 ..... 12, 34
GPIO 202 ..... 12, 34
GPIO 203 ..... 12, 34
Linux
serial ports 27
TimeSys operating system 35
major integrated circuits (chips)
implemented in EnCore M3 1
module location 9
mechanical dimensions
measurements in inches 13
Memory Management Unit (MMU) 51
next-generation system designs 5
better heat dissipation 5
increased performance 5
Internet connectivity 5
lower power consumption 5
not provided
bus signal timing 1
detailed integrated circuit specifications 1
internal registers 1
theory of operation 1
operating systems
VxWorks 35
Parallel port configuration
programmable 51
PCI interface
bus speed up to 33MHz 18
programmable logic device 50
interrupts directly processed 49
PCI Local Bus Specification 1
physical dimensions
height (overall above baseboard) 12
weight 12
width 12
measurements in inches 13
power requirements
baseboard 13
current 13
power in watts 13
voltages 13
processor speed
programming change 57
QuickStart Kit
EBX baseboard 2
EnCore Development Kit CD-ROM 2
software 2
various utilities 2
re-program flash
EJTAG debugging tool 58
SDRAM
Au1500 controller 17
memory control 17
Serial Debug port (J15)
special cable 63
serial ports
baud rate generator 57
Linux, TimeSys 27
VxWorks, Wind River 27
YAMON 27
SODIMM
requirements 17
software support
peripheral drivers 14
software tools 35
start-up code 14
support package 35
speaker
requires external drive circuit 29
special interface cables
Ethernet port (J14) 63
Serial Debug port (J15) 63
table legend
input and output codes 22
Technical Support
e-mail request 61
Embedded Design Resource Center 61
Virtual Technician 61
theory of operation
not provided 1
time of day clock
features 33
TLB (Translation Lookaside Buffer) 51
TLB configuration 51
Translation Lookaside Buffer (TLB) 51
Unix systems
rebuild compiler 36
Utility interface
features 27
voltage requirements
baseboard 13
CPU core voltage 13
EnCore M3 module 13
VxWorks
Board Support Package (BSP) 48
BSP functions 48
Wind River operating system 35
web sites
CPU specifications 2
detailed chip specifications 2
flash memory 2
Southbridge specifications 2
Super I/O specifications 2
Win32 systems
Cygwin environment 36
Windows-based environment
Cygwin 35
YAMON
monitor program features 37
serial ports 27
software support 35