EnCore™ PP1
Embedded Processor
Reference Manual

P/N 5001660A 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, EnCore, Little Board, CoreModule, 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	May/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

List of Figures

List of Tables

Chapter 1 About this Manual

Purpose of this Manual

This manual is for designers of systems based on the EnCore PP1 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 PP1 Specifications
- Environmental requirements
- Major chips and features implemented
- EnCore PP1 connector/pin numbers and definitions
- 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 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

- EnCore Architecture Specifications
- PCI 2.2 Compliant Specifications

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.)
503-797-4207 (International)

- EnCore EBX Baseboard Specification v1.1
- EnCore EBX Baseboard Drawings

White Papers

- Reducing Time to Profit with High-Integration CPU Modules

Chip specifications (web sites) used in the EnCore PP1 module

- Motorola's MPC8245 chip used for the embedded CPU

  Web site: http://e-www.motorola.com/brdata/PDFDB/docs/MPC8245EC.pdf

- VIA Technologies, Inc's VT82C686B chip used for the Super I/O controller

  Web site: http:// www.viatech.com.tw/pdf/producinfo/686abrief.pdf

- AMD's AM29LV160DT chip used for the Flash memory

  Web site: http:// www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/22358b.pdf

- Intel Corporation's 82559ER chip used for the Ethernet controller

  Web site: http:// developer.intel.com/design/network/products/82559er.html

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 PP1 QuickStart Kit simultaneously with the design of your custom logic board.

EnCore PP1 Support Products

- EnCore QuickStart Kit (QSK)

  The QuickStart Kit includes the EnCore PP1 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 PP1 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 peripheral connections necessary to test and debug much of your software. 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, audio connections, and a standard ATX power supply connection are provided on the EnCore EBX baseboard.

- EnCore Development Kit (EDK) CD-ROM

  This is the basic support package provided with the EnCore PP1module. The CD-ROM provides all of the documentation in PDF format, including this reference manual, the EnCore PP1 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 layout files, a bill of materials (BOM), and an Approved Vendor List (AVL). The CD-ROM also contains a software development environment, including the monitor same as for the PPCBOOT as well as the Board Support Packages (BSP) for the supported operating systems (start-up code and drivers for each peripheral device on the EnCore PP1 module).

Other Ampro EnCore Products

- EnCore™ M2 – This embedded processor module is a low-power, high-performance, high-integration MIPS32-compatible module using a 400MHz AMD Alchemy Au1000™ processor rated at 480 Dhrystone MIPS. The typical power consumption of an EnCore M2 module is less than 2.5 watts. In addition to the standard EnCore features, the EnCore M2 includes a second 10/100BaseT Ethernet port.

- EnCore™ 400 – This embedded processor module is a low-cost, high-integration 486-based CPU 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 a high performance 3D graphics controller for CRTs and popular LCD panels.
- EnCore™ 700 – This embedded processor module is the highest performance EnCore product offering a high performance 850MHz Pentium® III processor and incorporates the Intel 815EM chipset. In addition to the standard EnCore features, the EnCore 700 includes a 2D/3D graphics controller, which provides both CRT and TFT flat panel video interfaces.

Other Ampro Products

- Little Board™ Family – These highly integrated single-board computers in the EBX form factor (5.75x8.00 inches) are available with 486, Mobile Pentium and Pentium II processors. The Little Board single-board computer offers 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 (3.6x3.8 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 PP1 Embedded Processor module. After reading this chapter you should understand:

- EnCore Concept
- Developmental strategies using the EnCore PP1 Embedded Processor
- EnCore PP1 architecture
- EnCore PP1 module features
- Major components
- Connectors
- 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.

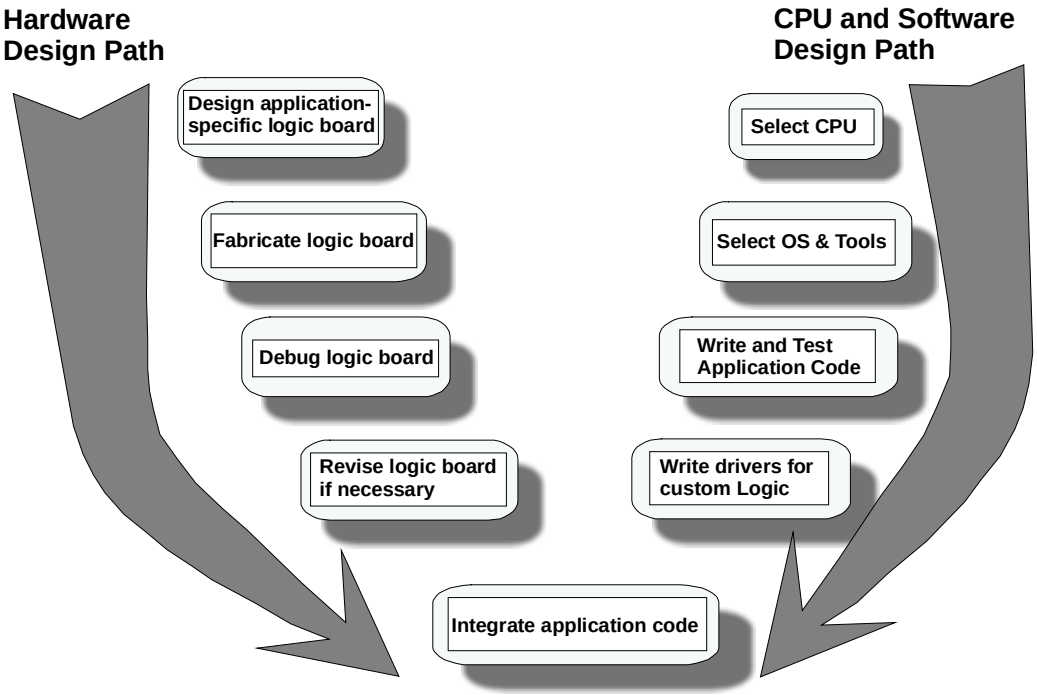

Figure 2-1. Typical Design Flow

Product Description

The EnCore PP1 module is one in a series of modules based on Ampro's EnCore platform. This platform was created to provide the efficient delivery of high performance CPU technology while reducing time-to-market for OEM embedded applications. This module uses the PowerPC™ based instruction set architecture from Motorola for the processor providing the EnCore PP1 module with the functionality of a complete computer in the 100x145mm (3.94x5.7 inch) form factor.

The EnCore PP1 module supports up to 512MB SDRAM in the Small Outline (SO) DIMM package, a floppy drive or ECP/EPP parallel port, USB ports, Infrared port, two serial ports, one serial debug port, primary IDE controller, and PCI expansion bus. The EnCore PP1 module also includes 10/100BaseT Ethernet, and the interface to the AC97 CODEC audio connections.

Like all modules in the EnCore series, the EnCore PP1 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 PP1 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.

Features

- PowerPC™ 603e Processor (MPC8245) Core
  - 300MHz operation
  - Integer Unit (IU), Floating Point Unit (FPU), Load/Store Unit (LSU)
  - System Register Unit (SRU), and a Brand Processing Unit (BPU)
  - 16KB Non-Blocking Data Cache; 16KB Instruction Cache
  - Low power of 1.7Watts @ 300MHz Core Speed
  - Dynamic power management-support 60x nap, doze, and sleep mode
  - Highly-Integrated System Peripherals
    - DMA Controller
    - DUART
    - I^2C Controller
    - Performance Monitor
    - Data Path Diagnostics and Watchpoints
- Memory
  - Programmable timing SDRAM support
  - High-Bandwidth Memory Bus (64-bit Data bus) to DRAM
  - 100MHz Clock Speed
  - Supports 16MB to 512MB SDRAM memory
  - Contiguous memory Mapping
  - 2MB Flash Memory
  - Supports bus-width write to ROM Flash
  - Write buffering for PCI and processor access
  - 144-Pin Standard Small Outline (SO) DIMM Socket
- PCI Bus
  - Up to 66MHz operation
  - PCI 2.2 Compliant Devices (32-bit wide bus)
  - Supports up to 4 PCI devices on the baseboard (connected to bus 0)
  - Supports 3 PCI Master devices on the baseboard (connected to bus 0)
  - 32-Bit PCI Master And Target Operations
- Input/Output Interfaces
  - Ultra DMA IDE Interface
    - Single Bus Master, Two EIDE Devices
    - Supports ATAPI and Tape Peripherals
    - Supports 40-pin IDE interface
  - Utility Interface
    - Four USB V1.1 ports for Host controller interface (two root hubs)
    - Two Serial Ports

- One ECP/EPP Parallel Port/Floppy Drive Interface
- Keyboard and PS/2 Mouse Interface
- One dedicated two-way Infrared (IrDA) Port Interface
- Integrated Peripheral Controllers
  - 2X8237/AT compatible DMA controller
  - 2X8259/AT compatible Interrupt controllers
  - Handles 16 interrupt (IRQs) inputs

- Audio Interface
  - AC97 CODEC Interface
- Ethernet Interface
  - Intel GD82559ER Fast Ethernet PCI Controller
  - IEEE 802.3 10/100BaseTX compatible physical layer
- Miscellaneous
  - Real Time Clock
  - JTAG connector (J6)
  - Serial debug port (J15)
- Power Supply
  - 3.3V @ 0.55Amps
  - 5.0V @ 0.43 Amps

Block Diagram

The functional block diagram of the EnCore PP1 module is shown below, in Figure 2-1.

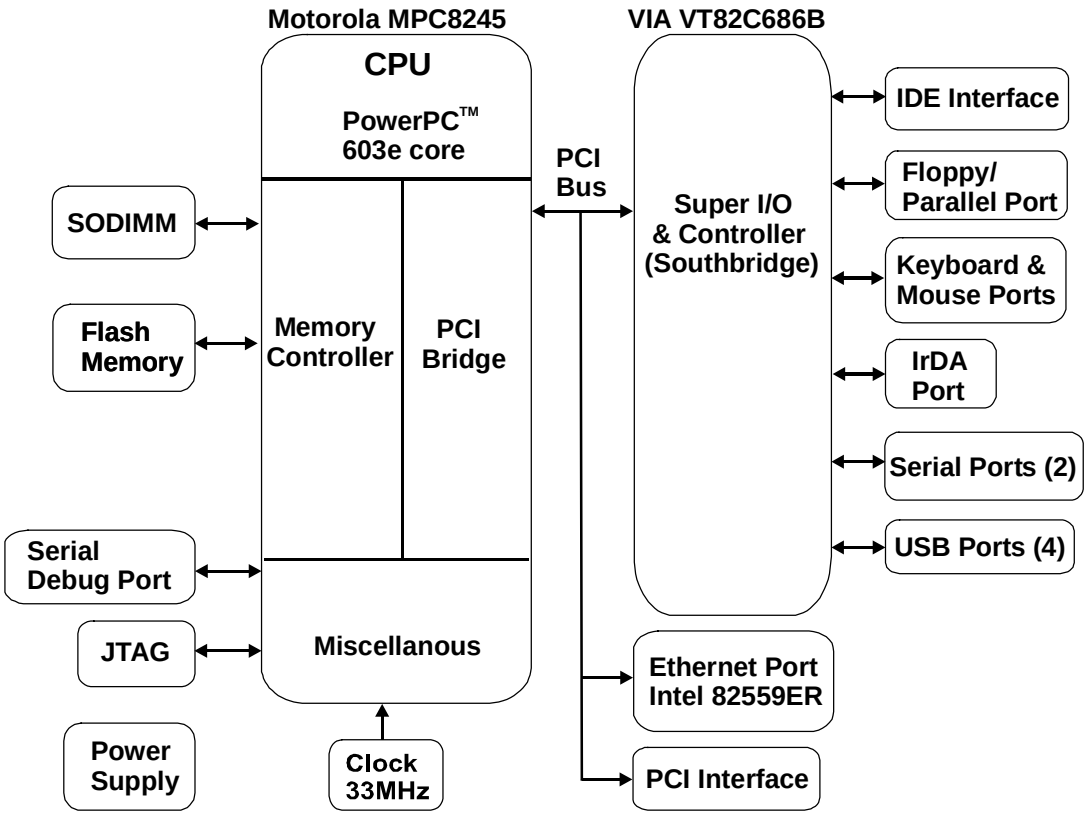

Figure 2-1. EnCore PP1 Block Diagram

Major EnCore PP1 Integrated Circuits

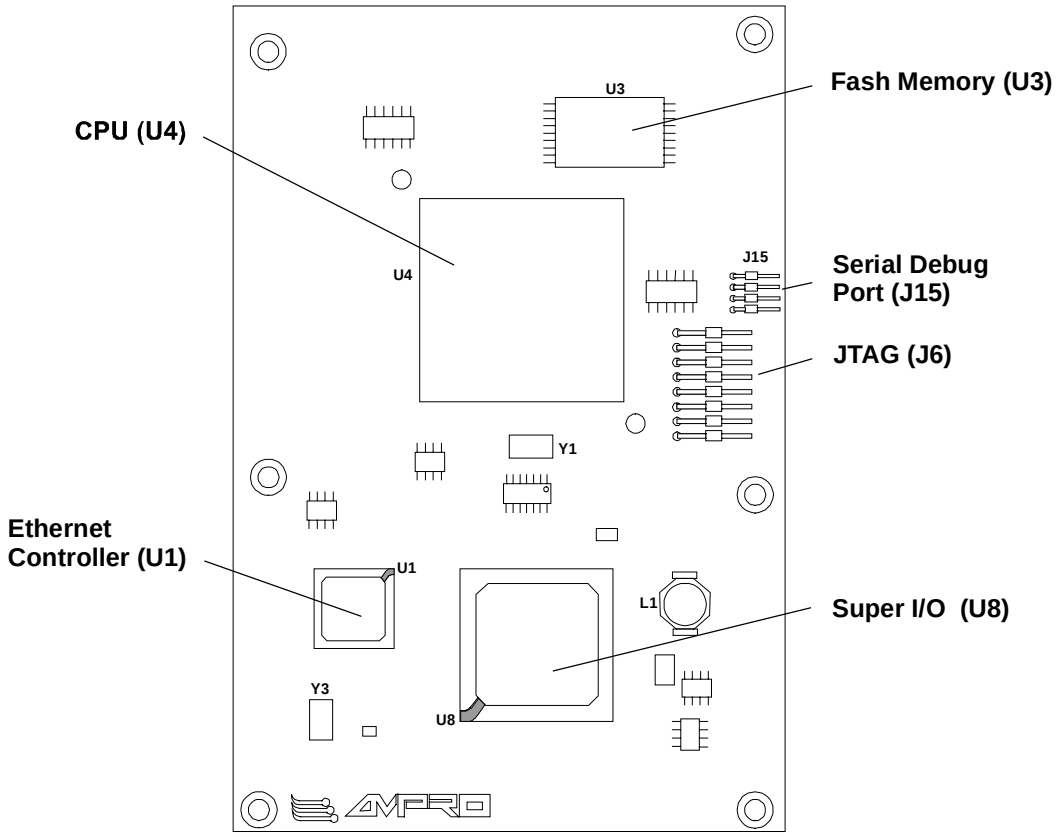

Figure 2-2. Major Component Locations (top view)

Table 2-1. Description of Major Chips

Manufacture	Model	Description	Features
Motorola	MPC8245	Embedded CPU (U4) – This integrated processor combines a PowerPC™ 603e microprocessor with a PCI bridge and high-performance memory controller.	CPU PCI Bridge Memory Controller
AMD	AM29LV160DT	Flash memory (U3) – This chip provides the CMOS flash memory capability for the module.	Flash memory
VIA Technologies, Inc.	VT82C686B	Super I/O (U8) – This chip provides the floppy controller, parallel controller, serial controller, Ultra DMA/PCI-EIDE controller, infrared controller, audio controller, keyboard controller, and USB controller.	Super I/O Controller (Southbridge) Floppy disk controller Parallel controller Serial controller Keyboard controller UDMA controller PCI controller USB controller IrDA controller Audio controller
Intel	82559ER	Ethernet (U1) – This chip provides the 10/100BaseT internet connection.	Ethernet controller

Specifications

Physical Specifications

The following table and figure provides the mounting dimensions and mechanical form factors for the EnCore PP1 Embedded Processor. The EnCore PP1 module conforms to the EnCore physical standard to insure the widest possible design coverage for developers.

Table 2-2. Weight and Footprint Dimensions

Weight	90.7 grams (0.2 lbs)
Height (overall above baseboard)	21.59 mm (0.85 inches)
Width	100 mm (3.94 inches)
Length	145 mm (5.71 inches)

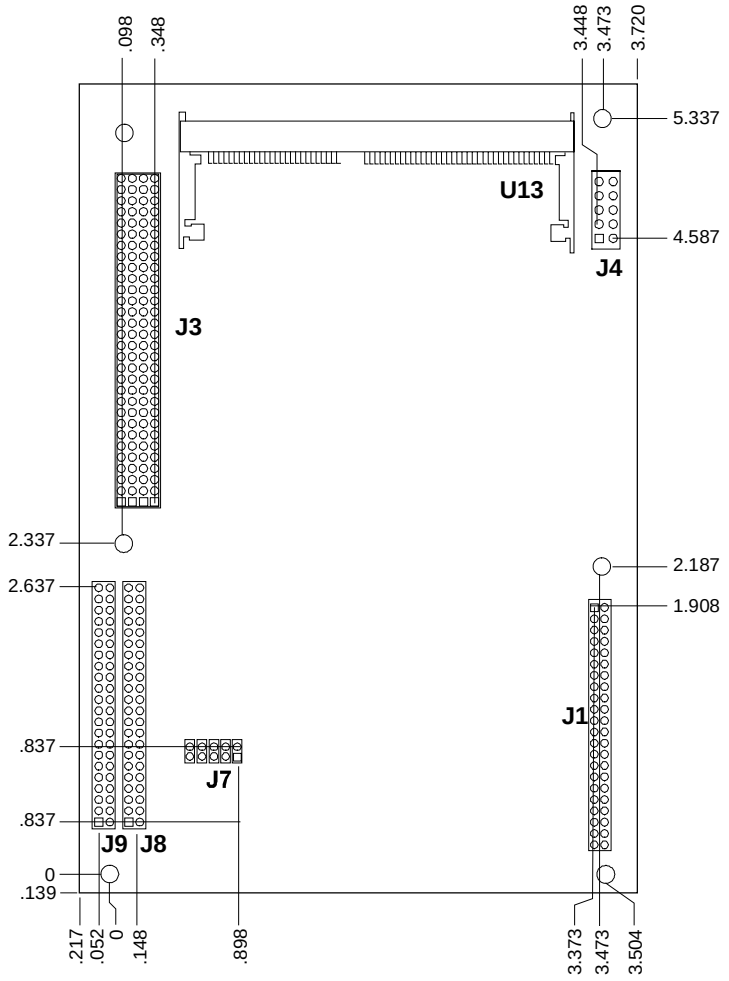

Figure 2-3. Mechanical Layout of System Board (through board view)

NOTE	All dimensions are shown in inches.

Connector Definitions

All of the connectors on the EnCore PP1 module and the respective descriptions are listed in the following table.

Table 2-3. Connector Definitions

Jack # -Identity	Board Access	Description
J1– I/O (IDE)	Bottom	This is a 44-pin connector used for the I/O signals including the IDE interface
J3 – PCI Bus	Bottom	This is a 120-pin connector used for the PCI interface
J4 – Power	Bottom	This is a 10-pin connector used for the Power interface
J6 – JTAG	Top	This is a right angle, 16-pin connector used for debugging purposes
J7 – Ethernet	Bottom	This is a 10-pin connector used for the Ethernet interface
J8 – Floppy/Parallel	Bottom	This is a 44-pin connector used for the Floppy/Parallel interface
J9 – Utility	Bottom	This is a 44 pin connector used for the Utility interface
J15 – Serial Debug	Top	This is a 4-pin connector used as a system console or debug interface.

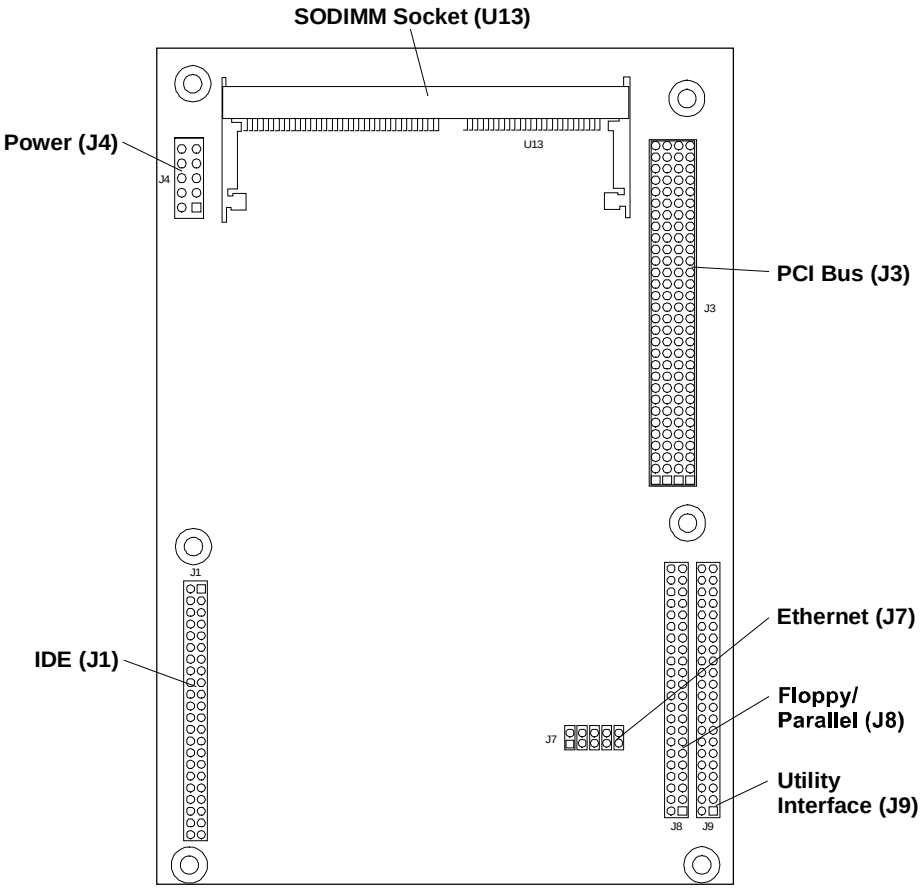

Figure 2-4. Connector Locations (bottom view)

Power Specifications

The EnCore PP1 module power requirements from the baseboard are listed in the following table.

Table 2-4. Power Supply Requirements

Parameter	Characteristics
Input Type	Regulated DC voltages
Inputs Voltages	+5 Volts AC/DC +/- 5 % @ 0.43 Amps
	+3.3 Volts AC/DC +/- 5 % @ 0.55 Amps
Operating Power	3.93 Watts

Operating Environment

Table 2-5. Environmental Limitations

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

Thermal/Cooling Requirements

The CPU and super I/O chips draw most of the power and generate most of the heat. The module is designed to operate at the maximum speed (300MHz) of the CPU. A heat sink may be required if you use the EnCore PP1 near its maximum operating characteristics.

EnCore PP1 Software

There are several popular operating systems used pervasively in embedded applications and these have already been ported to EnCore modules. Please check with the Ampro web site for the most up-to-date list of software available for EnCore PP1 module. For more information concerning the software provided for the EnCore PP1 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
  - Integrated Peripherals
    - DMA Controller
    - DUART
    - I^2C Controller
    - Performance Monitor
    - Data Path Diagnostics and Watchpoints
- Memory
  - SDRAM
  - Flash Memory
- PCI Bus Interface (J3)
- IDE Interface (J1)
- Floppy/Parallel/Serial Interface (J8)
- Utility Interface (J9)
  - USB (Universal Serial Bus)
  - Audio Controller
  - PS/2 Keyboard and PS/2 Mouse Controller
  - Infrared Port (IrDA)
- Ethernet Interface (J7)
- Miscellaneous
  - Real Time Clock (RTC)
  - JTAG/COP (J6)
  - Serial debug port (J15)
- Power Supply Interface (J4)

NOTE	The EnCore PP1 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 EnCore PP1 uses Motorola's MPC8245 chip, which is an integrated processor that combines a PowerPC™ 603e core with an integrated Host-to-PCI bridge capable of operating at 33MHz or 66MHz. The processor core itself can operate at speeds of 266-300MHz. The MPC8245 chip also integrates a high-performance memory controller, which can support various types of DRAM and ROM.

NOTE	The processor clock speed is slowed to 266MHz from 300MHz when the signals to the PCI bus operate at 66MHz. This is required by the clock divider circuitry in the MPC8245.

The EnCore PP1 module also makes use of the integrated peripherals on Motorola's MPC8245 (CPU) chip. These integrated peripheral chip features are described in the following headings. For more information, refer to Motorola MPC8245 data book.

DMA Controller

The integrated DMA controller contains two independent channels (0 and 1). Both channels transfer blocks of data independent of the processor core or PCI hosts, but both are accessible by the processor core and remote PCI masters. The processor core and PCI masters can initiate a DMA transfer. However, the DMA writing capability for local memory is available for SDRAM, but writing is not available for the ROM/PortX interface. Each DMA channel is capable of performing four types of transfers:

- PCI-to-local memory
- Local-to-PCI memory
- PCI-to-PCI memory
- Local to local memory

Other features include:

- Misaligned transfer capability
- Chaining mode (including scatter gathering with a 64-byte queue)
- Direct mode
- Interrupt on completed segment, chain, and error conditions
- PCI Dual Address Cycle (DAC) support

For more information refer to Motorola MPC8245 data book.

DUART

The DUART (Dual Universal Asynchronous Receiver/Transmitter) controls the processor core interface to the serial devices attached to the UART signals. Each UART is capable of converting the parallel data from the processor core into a single serial bit stream for outbound transmission. On the inbound transmission, the UART converts the serial bit stream into the bytes for processor core handling. On the EnCore PP1 module only, the DUART (channel 1) is connected to the serial debug port (J15). For more information, refer to Motorola MPC8245 data book.

I^2C Controller

The I^2C (Inter-Integrated Circuit) controller is a two-wire interface, which allows communication between the CPU and various system components on the module. Some of the I^2C features include allowing devices to indicate manufacturer information, the device's model/part number, save the device's state for a suspend event, report different types of errors, accept control parameters, and return device status.

The I^2C bus 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 I^2C also incorporates an arbitration mechanism to prevent multiple devices from taking control of the bus. The MPC8245 chip is used to control (master) the I^2C bus and is typically used to communicate between the Clock Generator Chip, DRAM EEPROM, and the Serial EEPROM (SEEP).

Currently there are two slave devices on the I^2C bus, the SPD SEEP on the SODIMM module, located at I^2C address 0XA0 and the general purpose SEEP at 0xA8. The general purpose SEEP can be used to store up to 512Bytes of data in non-volatile storage. Software support is provided in the BSPs and the Boot Monitor. The SPD (serial presence detect) SEEP is located on the SODIMM module, and contains configuration information for the SDRAM on the module. The format of this data is described in JEDEC Standard, #21-C, which is available from the JEDEC web site (www.jedec.org). For more information, also refer to Motorola MPC8245 data book.

Performance Monitor

The MPC8245 core logic contains a performance facility that monitors bridge logic events such as SDRAM or PCI bus traffic, DUART, or a number of interrupts emanating from an interrupt controller. The performance monitor can be used for the following functions:

- To optimize overall system performance by monitoring bridge logic events.
- To increase understanding of MPC8245 behavior in any system or software environment. Some systems or software environments are not easily characterized by signal traces or benchmarks.
- To help system developers bring up and debug their systems.

For more information, refer to Motorola MPC8245 data book.

Data Path Diagnostics and Watchpoints

The MPC8245 chip contains six memory data-path diagnostic debug registers, which help you to bring up the system and aid in debugging processor related problems. The MPC8245 chip also provides programmable watchpoints on the internal processor bus allowing you to monitor the state of the internal peripheral logic bus and generate triggers that can be monitored as an output signal. For more information, refer to Motorola MPC8245 data book.

Memory

SDRAM

The EnCore PP1 module supports one standard 64-bit Small Outline (SO) DIMM socket (144-pin) and supports a 64-bit wide data path in a SDRAM package. Ampro supports SDRAM modules with sizes from 16MB to 512MB. Ampro supplies fully qualified and tested memory, which is compatible with the EnCore PP1 module.

Should the user wish to provide his own memory, these SODIMMs are commercially available, but you should use PC100 (100 MHz or faster), 3.3V, 8ns, SDRAM 144-pin SODIMMs.

Flash Memory

The EnCore PP1 module has 2MB of flash memory on the module in a single 8-bit wide flash device. The flash device used is an AMD AM29LV160DT. Ampro expects resident flash memory to be used for a bootloader function. The operating system and applications code can be loaded from a network or IDE device. Ampro provides a bootloader for this purpose. Refer to *Chapter 4, Software* for a Flash Memory Address Map and a description of the Ampro bootloader.

PCI Bus Interface (J3)

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

The PCI Bus has four groups of signals and each group has a separate 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.

NOTE	The processor clock speed is slowed to 266MHz from 300MHz when the signals to the PCI bus operate at 66MHz. Pin-D6 (pin-96) is used to enable 66MHz operation.

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

Pin #	Signal	Input/ Output	Description
1 (A1)	Reserved (Key)		Reserved
2 (A2)	VI/O		+5 volts ±5% power supply
3 (A3)	AD05	T/S	PCI Address and Data Bus Line 5 – There are 32 signal lines (address and data) and the signals on these lines 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 Enable 0 – This signal line is one of four signal lines. These signal lines are multiplexed, so that during the address cycle, the command is defined and during the data cycle, the byte enable is defined.
5 (A5)	GND		Digital Ground
6 (A6)	AD11	T/S	PCI Address and Data Bus Line 11 – Refer to Pin 3 for more information.
7 (A7)	AD14	T/S	PCI Address and Data Bus Line 14 – Refer to Pin 3 for more information.
8 (A8)	+3.3V		+3.3 volts ±5% power supply
9 (A9)	SERR*	O/D	System Error – This signal is for reporting address parity errors.
10 (A10)	GND		Digital Ground
11 (A11)	STOP*	S/T/S	Stop – This 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 signal 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 Line 18 – Refer to Pin 3 for more information.
16 (A16)	AD21	T/S	PCI Address and Data Bus Line 21 – Refer to Pin 3 for more information.
17 (A17)	+3.3V		+3.3 volts ±5% power supply
18 (A18)	IDSEL0	In	Initialization Device Select 0 – This signal line is one of four signal lines. These signals are used as the chip-select signals during configuration
19 (A19)	AD24	T/S	PCI Address and Data Bus Line 24 – Refer to Pin 3 for more information.
20 (A20)	GND		Digital Ground
21 (A21)	AD29	T/S	PCI Address and Data Bus Line 29 – Refer to Pin 3 for more information.
22 (A22)	+5V		+5 volts ±5% power supply
23 (A23)	REQ0*	T/S	Bus Request 0 – This signal line is one of three signal lines. These signals indicate the device desires use of the bus to the arbitrator.
24 (A24)	GND		Digital Ground

Pin #	Signal	Input/ Output	Description
25 (A25)	GNT1*	T/S	Grant 1 – This signal line is one of three signal lines. These signal lines indicate access has been granted to the requesting device (PCI Masters).
26 (A26)	+5V		+5 volts ±5% power supply
27 (A27)	CLK2	In	PCI clock 2 – This signal line is one of four signal lines. These clock signals provide the timing outputs for four external PCI devices and the timing for all transactions on the PCI bus
28 (A28)	GND		Digital Ground
29 (A29)	+12V		+12 volts ±5% power supply
30 (A30)	NC		Reserved
31 (B1)	NC		Reserved
32 (B2)	AD02	T/S	PCI Address and Data Bus Line 2 – Refer to Pin 3 for more information.
33 (B3)	GND		Digital Ground
34 (B4)	AD07	T/S	PCI Address and Data Bus Line 7 – Refer to Pin 3 for more information.
35 (B5)	AD09	T/S	PCI Address and Data Bus Line 9 – Refer to Pin 3 for more information.
36 (B6)	VI/O		+5 volts ±5% power supply
37 (B7)	AD13	T/S	PCI Address and Data Bus Lines 13 – Refer to Pin 3 for more information.
38 (B8)	C/BE1*	T/S	PCI Bus Command/Byte Enable 1 – Refer to Pin 4 for more information.
39 (B9)	GND		Digital Ground
40 (B10)	PERR*		Parity Error – This signal 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 device's 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 Line 16 – Refer to Pin 3 for more information.
45 (B15)	+3.3V		+3.3 volts ±5% power supply
46 (B16)	AD20	T/S	PCI Address and Data Bus Lines 20 – Refer to Pin 3 for more information.
47 (B17)	AD23	T/S	PCI Address and Data Bus Line 23 – Refer to Pin 3 for more information.
48 (B18)	GND		Digital Ground
49 (B19)	C/BE3*	T/S	PCI Bus Command/Byte Enable 3 – Refer to Pin 4 for more information.
50 (B20)	AD26	T/S	PCI Address and Data Bus Line 26 – Refer to Pin 3 for more information.
51 (B21)	+5V		+5 volts ±5% power supply
52 (B22)	AD30	T/S	PCI Address and Data Bus Line 30 – Refer to Pin 3 for more information.
53 (B23)	GND		Digital Ground
54 (B24)	REQ2*	T/S	Bus Request – Not supported
55 (B25)	VI/O		+5 volts ±5% power supply
56 (B26)	CLK0	In	PCI clock 0– Refer to Pin 27 for more information
57 (B27)	+5V		+5 volts ±5% power supply
58 (B28)	INTD*	O/D	Interrupt D – This signal is used to request interrupts only for multi-function devices.
59 (B29)	INTA*	O/D	Interrupt A – This signal is used to request an interrupt.
60 (B30)	NC		Reserved
61 (C1)	+5		+5 volts ±5% power supply
62 (C2)	AD01	T/S	PCI Address and Data Bus Line 1 – Refer to Pin 3 for more information.
63 (C3)	AD04	T/S	PCI Address and Data Bus Lines 4 – Refer to Pin 3 for more information.

Pin #	Signal	Input/ Output	Description
64 (C4)	GND		Digital Ground
65 (C5)	AD08	T/S	PCI Address and Data Bus Line 8 – Refer to Pin 3 for more information.
66 (C6)	AD10	T/S	PCI Address and Data Bus Line 10 – Refer to Pin 3 for more information.
67 (C7)	GND		Digital Ground
68 (C8)	AD15	T/S	PCI Address and Data Bus Line 15 – Refer to Pin 3 for more information.
69 (C9)	SB0*		Snoop Backoff
70 (C10)	+3.3V		+3.3 volts ±5% power supply
71 (C11)	LOCK*	S/T/S	Lock – This 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's 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 Line 17 – Refer to Pin 3 for more information.
76 (C16)	GND		Digital Ground
77 (C17)	AD22	T/S	PCI Address and Data Bus Line 22 – Refer to Pin 3 for more information.
78 (C18)	IDSEL1		Initialization Device Select 1 – Refer to Pin 18 for more information
79 (C19)	VI/O		+5 volts ±5% power supply
80 (C20)	AD25	T/S	PCI Address and Data Bus Line 25 – Refer to Pin 3 for more information.
81 (C21)	AD28	T/S	PCI Address and Data Bus Line 28 – Refer to Pin 3 for more information.
82 (C22)	GND		Digital Ground
83 (C23)	REQ1*	T/S	Bus Request 1 – Not supported
84 (C24)	+5V		+5 volts ±5% power supply
85 (C25)	GNT2*	T/S	Grant 2 – Refer to Pin 25 for more information
86 (C26)	GND		Digital Ground
87 (C27)	CLK3	In	PCI clock 3 – Refer to Pin 27 for more information
88 (C28)	+5V		+5 volts ±5% power supply
89 (C29)	INTB*	O/D	Interrupt B – This signal is used to request interrupts only for multi-function devices.
90 (C30)	PME*		Power Management Event – This signal is used for power management events
91 (D1)	AD00	T/S	PCI Address and Data Bus Line 0 – Refer to Pin 3 for more information.
92 (D2)	+5V		+5 volts ±5% power supply
93 (D3)	AD03	T/S	PCI Address and Data Bus Lines 3 – Refer to Pin 3 for more information.
94 (D4)	AD06	T/S	PCI Address and Data Bus Lines 6 – Refer to Pin 3 for more information.
95 (D5)	GND		Digital Ground
96 (D6)	M66EN		This signal enables 66MHz operation.
97 (D7)	AD12	T/S	PCI Address and Data Bus Line 12 – Refer to Pin 3 for more information.
98 (D8)	+3.3V		+3.3 volts ±5% power supply
99 (D9)	PAR	T/S	PCI bus Parity bit – This signal 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 – This signal is driven by the target device when its address is decoded.
103 (D13)	+3.3V		+3.3 volts ±5% power supply

Pin #	Signal	Input/ Output	Description
104 (D14)	C/BE2*		PCI Bus Command/Byte Enable 2 – Refer to Pin 4 for more information.
105 (D15)	GND		Digital Ground
106 (D16)	AD19	T/S	PCI Address and Data Bus Line 19 – Refer to Pin 3 for more information.
107 (D17)	+3.3V		+3.3 volts ±5% power supply
108 (D18)	IDSEL2		Initialization Device Select 2 – Refer to Pin 18 for more information.
109 (D19)	IDSEL3		Initialization Device Select 3 – Refer to Pin 18 for more information.
110 (D20)	GND		Digital Ground
111 (D21)	AD27	T/S	PCI Address and Data Bus Line 27 – Refer to Pin 3 for more information.
112 (D22)	AD31	T/S	PCI Address and Data Bus Line 31 – Refer to Pin 3 for more information.
113 (D23)	VI/O		+5 volts ±5% power supply
114 (D24)	GNT0*	T/S	Grant 0 – Refer to Pin 25 for more information.
115 (D25)	GND		Digital Ground
116 (D26)	CLK1	In	PCI clock 1 – Refer to Pin 27 for more information
117 (D27)	GND		Digital Ground
118 (D28)	RST*	In	PCI bus reset – This signal is an output signal to reset the entire PCI Bus. This signal will be asserted during system reset
119 (D29)	INTC*	O/D	Interrupt C – This signal is used to request interrupts only for multi-function devices.
120 (D30)	GND		Digital Ground

Notes: 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.

I/O Interface

The EnCore PP1 Embedded Processor supports a number of industry standard I/O interfaces. These include an EIDE drive interface, floppy disk interface, serial and parallel ports, AC97 Audio, IrDA, keyboard and mouse. These interfaces are driven primarily from the VIA VT82C686 Super I/O controller. The interface signals are output on one of the three EnCore PP1 I/O connectors.

The following table describers which I/O interface is driven by each chip to which EnCore PP1 I/O connector.

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

I/O Interface	Chip Source	EnCore PP1 I/O Connector
EIDE	VIA (VT82C686B)	IDE (J1)
Floppy Disk Interface	VIA (VT82C686B)	Floppy / Serial / Parallel (J8)
Serial Ports	VIA (VT82C686B)	Floppy / Serial / Parallel (J8)
Parallel Port	VIA (VT82C686B)	Floppy / Serial / Parallel (J8)
USB Ports 1-4	VIA (VT82C686B)	Utility (J9)
AC97 Audio	VIA (VT82C686B)	Utility (J9)
IrDA	VIA (VT82C686B)	Utility (J9)
PS/2 Keyboard	VIA (VT82C686B)	Utility (J9)
PS/2 Mouse	VIA (VT82C686B)	Utility (J9)

The following sections describe the features of each of the I/O interfaces and the connector pin-out of each of the I/O connectors. Refer to *Chapter 4, Software* for the I/O Memory Map.

The I/O Interface in the EnCore PP1 is actually divided into three 44-pin I/O interface connectors:

- IDE interface connector (J1)
- Floppy/Parallel/Serial Interface connector (J8)
- Utility Interface connector (J9)

NOTE The I/O device addresses are configured by the PPCBOOT monitor at start up. Refer to *Chapter 4, Software* for more information.

IDE Interface (J1)

The IDE device signals are provided to the baseboard through the IDE 44-pin connector (J1).

IDE Features:

- Master mode PCI supporting Enhanced IDE devices
- Two EIDE devices supported
- Transfer rate up to 33MB/sec
- Increased reliability using UltraDMA-66/100 transfer protocols
- Full scatter-gather capability
- Support ATAPI compliant devices including DVD devices
- Support IDE native and ATA compatibility modes

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

Pin #	Signal	Description
1	RESET*	Low active hardware reset (RSTDRV inverted)
2	GND	Digital Ground
3	D7	Disk Data - These signals (0 to 15) provide the disk data signals
4	D8	Disk Data - These signals (0 to 15) provide the disk data signals
5	D6	Disk Data - These signals (0 to 15) provide the disk data signals
6	D9	Disk Data - These signals (0 to 15) provide the disk data signals
7	D5	Disk Data - These signals (0 to 15) provide the disk data signals
8	D10	Disk Data - These signals (0 to 15) provide the disk data signals
9	D4	Disk Data - These signals (0 to 15) provide the disk data signals
10	D11	Disk Data - These signals (0 to 15) provide the disk data signals
11	D3	Disk Data - These signals (0 to 15) provide the disk data signals
12	D12	Disk Data - These signals (0 to 15) provide the disk data signals
13	D2	Disk Data - These signals (0 to 15) provide the disk data signals
14	D13	Disk Data - These signals (0 to 15) provide the disk data signals
15	D1	Disk Data - These signals (0 to 15) provide the disk data signals
16	D14	Disk Data - These signals (0 to 15) provide the disk data signals
17	D0	Disk Data - These signals (0 to 15) provide the disk data signals
18	D15	Disk Data - These signals (0 to 15) provide the disk data signals
19	GND	Digital Ground
20	Key	Key pin plug
21	IDRQ0	Primary Device DMA Channel Request
22	GND	Digital Ground
23	IOW*	Primary Device I/O Read/Write Strobe
24	GND	Digital Ground
25	IOR*	Primary Device I/O Read/Write Strobe
26	GND	Digital Ground
27	IORDY	Primary Device I/O-DMA Channel Ready
28	Reserved	Reserved – Not used
29	IDACK0*	Primary Device DMA Channel Acknowledge
30	GND	Digital Ground
31	IRQ14	Interrupt Request 14
32	NC	Not connected
33	A1	IDE ATA Primary Disk Address (0 to 2). Used to indicate which byte in the ATA command block or control block is being accessed
34	NC	Not connected
35	A0	IDE ATA Primary Disk Address (0 to 2). Used to indicate which byte in the ATA command block or control block is being accessed
36	A2	IDE ATA Primary Disk Address (0 to 2). Used to indicate which byte in the ATA command block or control block is being accessed
37	CS0	Primary Slave/Master Chip Select

Pin #	Signal	Description
38	CS1	Primary Slave/Master Chip Select
39	Reserved	Reserved – Not used
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 – Not used

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

Floppy/Parallel/Serial Interface (J8)

The Floppy disk controller, Parallel Port, and two Serial Ports are connected to the baseboard through the standard I/O 44-pin header (J8). This header is located on the bottom side of the EnCore PP1 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.

Floppy Disk Controller

The super I/O chip contains the Floppy Disk Controller, which shares the same output connector as the Parallel Port, but you can only use one of these devices at a time. The Floppy Disk Controller supports the following features:

- 16 Bytes of FIFO
- Data Rates up to 1Mbps
- Perpendicular Recording Driver support
- Two FDDs with Drive Swap Support

NOTE	Due to the shared status of the output connector, only one device can be connected at a time. That is, the floppy disk controller (floppy drive) or the parallel interface (printer port), but not both at the same time.

Parallel Port

The super I/O chip also provides the Parallel port interface, which shares the same output connector as the Floppy Disk Controller. The Parallel Port supports the following features:

- Standard Printer Port (SPP) support
- Enhanced Parallel Port (EPP) support
- Enhanced Capabilities Port (ECP) support

Serial Ports

The super I/O chip also provides for two 2 serial ports and both serial ports support TTL signals. The two serial ports support the following features:

- Both serial ports are 16540 compatible
- Each serial port has two 16-bit FIFOs
- Programmable word length, stop bits and parity
- 16-bit programmable baud rate generator
- Interrupt generator
- Loop-back mode

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

Pin #	Signal	Description
1	STB*	Strobe – Output used to strobe data into the printer. I/O pin in ECP/EPP mode.
2	AFD/ DRVEN0*	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*	Parallel Port Data Index – Sense to detect that the head is positioned over the beginning of a track
4	ERR/ HDSEL	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	Parallel Port Data Track 0 – Sense to detect that the head is positioned over track 0.
6	INIT*/ DIR*	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*	Parallel Port Data Write Protect – Senses the diskette is write protected.
8	SLIN/ STEP*	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*	Parallel Port Data Read Data – Raw serial bit stream from the drive for read operations.
10	GND	Digital Ground
11	PD4/ DSKCHG*	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	Parallel Port Data
14	GND	Digital Ground
15	PD6	Parallel Port Data
16	GND	Digital Ground

Pin #	Signal	Description
17	PD7	Parallel Port Data
18	GND	Digital Ground
19	ACK*/ DS1*	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*	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*	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*	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	NC	Not Connected
27	DCD1*	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 RS-232 receiver.
28	DSR1*	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 RS-232 receiver.
29	RXD1	Receive Data 1– Serial port 1 receive data in
30	RTS1*	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 RS-232 driver.
31	TXD1	Transmit Data 1 – Serial port 1 transmit data out
32	CTS1*	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 RS-232 receiver.
33	DTR1*	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 RS-232 driver.
34	RI1*	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 RS-232 receiver (whose input is typically not connected in direct connect environments).
35	GND	Digital Ground
36	SEL1	Interface Mode Select Input 1 – Default is Low State for selecting port 1 as RS-232 mode. This signal is used to control the LTC 1334 Multi protocol Transceiver on board. When this pin is driven High, port 1 changes to the RS-485 mode.

Pin #	Signal	Description
37	RXD2	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 RS-232 driver.
39	TXD2*	Transmit Data 2 – Serial port 2 transmit data out.
40	CTS2*	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 RS-232 receiver.
41	VCC	+5 volts ±5% power supply input
42	VCC	+5 volts ±5% power supply input
43	GND	Digital Ground
44	SEL2	Interface Mode Select Input 2 – This signal is used to control the LTC 1334 Multi protocol Transceiver on board. This signal is in Low State (Default) for selecting port 2 as RS232 mode. When this pin is driven High, port 2 changes to the RS485 mode.

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

Utility Interface (J9)

The Utility interface consist of the several buses and signals from the various components on the EnCore PP1 module. This interface uses a 44-pin 2mm header between the module and the baseboard. The following buses or signal are available through the Utility Interface (J9):

- USB (Universal Serial Bus)
- Audio Controller (Direct Sound Ready AC97 Digital Audio)
- PS/2 Controller (Keyboard and Mouse - Ports available but not used)
- Infrared Port (IrDA)

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

Pin #	Signal	Description
1	AC_BIT_CLK	Audio CODEC ‘97 Clock (for the Direct Sound Ready AC97 Digital Audio)
2	AC_DATA_IN	Audio CODEC ‘97 Data In
3	AC_DATA_OUT	Audio CODEC ‘97 Data Out
4	AC_SYNC	Audio CODEC ‘97 Sync
5	AC_RESET#	Audio CODEC ‘97 Reset
6	GND	Digital Ground
7	SPKR	Speaker Output
8	GND	Digital Ground
9	RSTSW#	Reset Switch
10	KBDATA	Keyboard Data
11	KBCLK	Keyboard Clock
12	GND	Digital Ground
13	+5V	+5 volts ±5% power supply input
14	MDATA	Mouse Data

Pin #	Signal	Description
15	MCLK	Mouse Clock
16	GND	Digital Ground
17	+5V	+5 volts ±5% power supply input
18	NC	Not connected – Reserved for future use
19	IRTX	Infrared Transmit – IR transmit data out from serial port 2
20	IRRX	Infrared Receive – IR receive data in to serial port 2
21	GND	Digital Ground
22	NC	Not connected – Reserved for future use
23	USBPWR0	USB Port 0 Power Protection – Port 0 is disabled if this input is low. Direct inputs are provided for over current protection.
24	USBP0N	Universal Serial Bus Port 0 Data Negative.
25	USBP0P	Universal Serial Bus Port 0 Data Positive
26	USBPWR1	USB Port 1 Power Protection – Port 1 is disabled if this input is low. Direct inputs are provided for over current protection
27	USBP1N	Universal Serial Bus Port 1 Data Negative.
28	USBP1P	Universal Serial Bus Port 1 Data Positive.
29	USBP2N	Universal Serial Bus Port 2 Data Negative
30	USBP2P	Universal Serial Bus Port 2 Data Positive
31	USBP3N	Universal Serial Bus Port 3 Data Negative
32	USBP3P	Universal Serial Bus Port 3 Data Positive
33	PWRBTN#	Power Button
34	DETECT#	Battery Low Indicator – Active Low (10K Pull up to 3.3V if not used)
35	RI#	Ring Indicator
36	GND	Digital Ground
37	SLPPWR	+3.3V Sleep Power
38	AEN	Reserved
39	SUSC#	Suspend Control
40	SMBDATA (NC)	System Management Bus Clock .- This is the clock signal for the SMBus. – Not Connected.
41	SMBCLK (NC)	System Management Bus Data – This is the data signal for the SMBus. – Not Connected.
42	SMBALRT (NC)	System Management Bus Alert - When the chip is enabled to allow it, assertion generates an IRQ or SMI interrupt or a power management resumes event. – Not Connected.
43	NC	Not connected – Reserved for future use
44	GND	Digital Ground

Notes: The shaded area denotes power or ground.

USB (Universal Serial Bus)

The EnCore PP1 contains two root USB hubs with two ports each for a total of four functional USB ports. Two of the ports, Port 1 and Port 2, include over-current detection status on USB inputs. Both USB v.1.1 and Intel Universal HCI v.1.1 are supported, as well as legacy keyboard and PS/2 mouse support.

NOTE	The EnCore PP1 board does not include fuses for signal conditioning on the module. Both USB ports require logic on the baseboard design.

Audio Controller

The super I/O chip provides the audio control circuitry and when combined with driver software and AC97 CODEC, will provide a complete high quality audio solution. The super I/O chip includes an integrated FM synthesizer and Plug-and-Play interface and operates over the PCI version 2.2 bus master.

Speaker Interface

The Super I/O chip (VT82C686B) provides the standard speaker output signal.

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

Keyboard and PS/2 Mouse Controller

The Super I/O 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 PC I/O port addresses of 0X60 and 0X64.

Infrared Port (IrDA)

The IrDA (Infrared Data Association) port provides a two-way wireless communications port using infrared as a transmission medium. The VIA Super I/O chip provides two infrared methods, the SIR method and the Amplitude Shift Keyed Infrared (ASKIR) method. The SIR and ASKIR IR port are multiplexed on COM2.

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) 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 Super I/O chip data book and the Infrared Data Association web site at http://www.irda.org.

Ethernet Interface (J7)

The Ethernet solution is provided by the Intel 82559ER PCI controller chip and consists of both the Media Access Controller (MAC) and the physical layer (PHY) combined into a single component solution. The 82559 family members build on the basic functionality of the 82558 and contain power management enhancements.

The 82559ER is a 32-bit PCI device and provides enhanced scatter-gather bus mastering capabilities, which enables the 82559ER to perform high-speed data transfers over the PCI bus. The 82559ER bus master capabilities enable the component to process high-level commands and perform multiple operations, thereby off-loading communication tasks from the system CPU.

- Optimum Integration
  - Integrated IEEE 802.3 10BaseT and 100BaseT compatible physical layer (PHY)
  - Full duplex or half-duplex operation
- High Performance Networking Functions
  - Chained memory structure
  - Backward compatible software to the Intel 82558 and 82557 chips
  - Full Duplex support at 10bps or 100bps
  - IEEE 802.3u Auto-Negotiation support
  - Two FIFOs -transmit and receive at 3KB – These help prevent data underflow and overflow, allowing the 82559ER to transmit data with of minimum interframe spacing (IFS).
  - IEEE 802.3x 100Base-TX Flow Control support
  - TCP/UDP checksum offload capabilities
- Low Power features
  - Low power 3.3V device
  - Efficient dynamic standby mode

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

Pin #	GND	Descriptions
1	TX+	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-	
3	TXCT	Transmit center tap connected to the magnetic coil of isolation transformer
4	RXCT	Receive center tap connected to the magnetic coil of isolation transformer
5	RX+	Analog Twisted Pair Ethernet Receive Differential Pair – These pins receive the serial bit stream from the isolation transformer.
6	RX-	
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 of the connection. The speed LED will be on for 100Mbps and off for 10Mbps.

Note: The shaded area denotes power or ground.

NOTE	The EnCore PP1 module does not include an Isolation Transformer (magnetic) and RJ45 Jack for signal conditioning of the Ethernet connection. An Ethernet Interface Circuit is required on your custom logic baseboard.

Miscellaneous

Real Time Clock (RTC)

The EnCore PP1 contains a PC-style Real Time Clock (RTC), integrated into the VIA chipset.

JTAG/COP (J6)

The JTAG and COP (common on-chip processor) features are provided through the JTAG connector (J6) on the top surface of the module. JTAG is a hardware/software method used for debugging. For more information refer to the Motorola specifications for MPC8245 chip at the web site listed earlier in this manual. Ampro has validated commercially available JTAG tools with the EnCorePP1. Refer to the Ampro web site for a list of available tools.

Table 3-7. JTAG Interface Pins and Signals (J6)

Pin #	Name	In/ Out	Description
1	TDO		Test data output to the instruction or selected data registers. This signal will transition on the falling edge (valid on rising edge) of TCK.
2	NC		Not connected
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	TRST	In	Asynchronous TAP reset
5	VSense	N/A	Voltage Sense – Senses 3.3V on module.
6	VSense	N/A	Voltage Sense – Senses 3.3V on module.
7	TCK	I	Control clock for updating TAP controller and shifting data through instruction or selected data register.
8	NC		Not connected
9	TMS	I	Control signal for TAP controller. This signal is sampled on the rising edge of TCK.
10	NC	N/A	Not connected
11	SRESET		This functions as output SDMA12 in Extended ROM Mode
12	GND	N/A	Digital Ground
13	HRESET		Hard Reset – Performs a hard rest of EnCore PP1 module.
14	Key	N/A	None
15	VSense	N/A	Voltage Sense – Senses 3.3V on module.
16	GND	N/A	Digital Ground

Notes: The shaded area denotes power or ground.
The SRESET signals are not available in the Extended ROM Mode.

Serial Debug Port

The RS232 signal (DUART channel 1) is routed to J15. This allows you to use the two existing serial ports for normal serial communications task and adds a third serial port for serial debug capability with tools such as gdb.

Power Supply (J4)

The EnCore PP1 requires two regulated DC voltages from the logic baseboard. There is a voltage regulator on the module that supplies 2.0 volts to the CPU core. The power supply connector uses a 10-pin header at 0.100" spacing.

Table 3-8. Power Supply Interface Pins and Signals (J4)

Pin	Signal	Pin	Signal
1	+5	2	GND
3	+3.3	4	GND
5	+12 (NC)	6	GND
7	+3.3	8	GND
9	+5	10	GND

Chapter 4 Software

Overview

The EnCore Development Kit (EDK) CD-ROM contains support software and development tools for the EnCore PP1 platform. The development tools include versions that will run on Windows-based systems and Linux/Unix-based workstations. The EDK also includes utilities and Board Support Packages (BSP) for the Ampro supported operating systems. The following items are included on the CD-ROM:

- PPCBOOT - This is a monitor/bootloader based on PPCBOOT, which is resident in flash on the EnCore PP1 Module.
  - Source code files
  - Binary file – A binary image of PPCBOOT 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 of Windows-based systems.
    - Source files for gnu -
- VxWorks BSP – This includes a complete Board Support Package (BSP) for VxWorks Tornado 2.02. The VxWorks BSP requires licensing a Tornado 2.02 development environment from Wind River, which can not be purchased through Ampro. Contact Wind River directly for details.

Development Environment

Software development for the EnCore PP1 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:

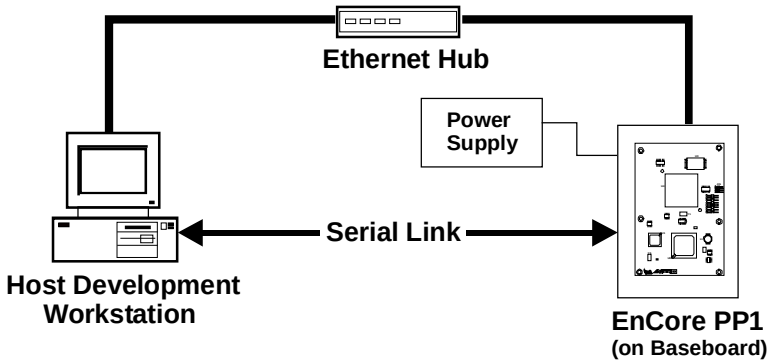

Figure 4-1. Typical Host/Target development environment

Installation and Configuration Overview

The provided tools support development under Windows-based systems and Unix-based workstations. The following text details the installation instructions for the standard cross development tools used with the PPCBOOT environment. These tools can be used to re-compile PPCBOOT, or develop application programs that can be loaded via PPCBOOT.

Installation on Windows Systems

Windows-based development workstations require a POSIX development environment for the GNU tool set. This POSIX development environment is provided by Cygwin tools. You will need a personal computer with Microsoft Windows 98, NT 4.0 SP6, or 2000. Ampro recommends a machine with approximately 500MB of disk space and a processor speed of 233MHz or faster, with at least 128MB RAM.

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

`http://sourceware.cygnus.com/cygwin`

The Cygwin environment is freely available and is included here on the CD-ROM as a convenience. The following steps will install the GNU tools on your system:

1. Insert the EnCore EDK CD-ROM into your system
2. Open the CYGWIN folder on the CD, and click on the *Setup program*.

   The initial screen will display the copyright information with a next button.
3. Click on ***Next*** in the initial screen of copyright information.

   The next screen provides installation locations.
4. Select ***Install from local directory*** from the listed provided.
5. Click on ***Next*** when you have made the selection.
6. Select an appropriate installation path for your system

   The path you enter is where cygwin will be installed.
7. Click on ***Next*** when you have entered the complete path.
8. Click on ***All Users*** for the scope, and then click on ***Unix*** for the default file type.

   This screen asks you to select the "Scope" and file type of the installation.
9. Click ***Next*** when you have made your selections.
10. Select the CD path as the location for the cygwin installation files and then click ***Next***.

    The system will display all of the components available. The default selection should be used, but if you have specific requirements you can select specific packages.

    a. To change the state of a package, click on the *circle arrow* icon

       Source files are provided, but are not required for proper operation.

    b. Click on ***Next*** when your selections are complete.

    The installation process will begin, installing the files in the path and directories you have specified. After it is complete, the CYGWIN environment will be loaded onto your system.
11. Select the icon settings appropriate for your system.

    The final screen prompts you for the selection of a desktop icon, and if CYGWIN is to appear in the start menu.
12. Click ***Finish*** to complete the installation.

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 –xvzf//m/compilers/comp2953CYG.tar (where 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. You can add the line in the *.bashrc* file in your login directory:

```
set PATH=/usr/local/cross/ppc-elf/2.95.3/bin:$PATH; export PATH
```

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

Installation on Linux System

A complete version of the compiler has been created for Linux based systems. This can be directly installed using the following procedure.

1. Go to the installation directory and ensure you have write access for this directory.

   cd/usr/local

2. Mount the CD-ROM, as required:

   mount/mnt/cdrom

3. Untar the compiler

   tar -xvzf/mnt/cdrom/compilers/comp2953LINUX.tar

   When you complete the Untar process the executable files will be installed. If you want to use the compilers, the location of the compilers must be put into the default path.

4. Set the path for the compilers by entering:

   set PATH=/usr/local/cross/ppc-elf/2.95.3/bin:$PATH; export PATH

   This can be added to your .bashrc file, as required.

Installation on Unix Systems

For UNIX systems, the source distribution 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”, you can choose any directory that makes sense for your configuration.

```
% 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
% cp /cdrom/mkcomp .
% vi mkcomp # edit INSTALL location
% ./mkcomp
```

PPCBOOT Overview

PPCBOOT is a monitor program available through the Open Source initiative, and modified by Ampro for the EnCore PP1. This monitor was developed to support booting Linux on non-x86 based platforms, and has become a popular choice for PowerPC-based systems. The development baseline is available at:

www.sourceforge.net.

This monitor program provides basic commands to boot from various media types, include IDE, Floppy, Ethernet, and Flash Disk. It also provides basic debugging commands and hardware diagnostics that provide a way to test various aspects of the operation of the EnCore PP1 module. Ampro has added functionality to allow the monitor to perform full module initialization automatically.

Some of the main PPCBOOT features are:

- System initializations, including RAM size/type detection and auto configuration, cache initialization.
- PCI auto detection and auto configuration
- User Shell
- Ethernet, serial port, and Hard Disk support
- Configuration of CPU
- Supports DHCP client, TFTP boot

PPCBOOT supports the following interfaces:

- Command line interface through the serial port
- Debug interface through dedicated debug serial port. Interface conforms to gnu-gdb
- Low Level call interface for use by applications.
- Ethernet for TFTP-load and "ping" support
- Support for IDE devices supporting FAT16 and FAT32 file systems.
- Support for I^2C bus devices

The implementation of PPCBOOT used by Ampro provides all of the major functions listed above. In addition, a few test routines (memory and flash) are included. The Ampro version of PPCBOOT will also automatically detect the size of the SODIMM module inserted into the SODIMM slot.

Development Environment

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

Installation and Configuration

The source for the PPCBOOT monitor is included in a tarball on the CD-ROM. This can be installed on a Windows-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. For example, under Cygwin;

```
$ mkdir /usr/home/ppcboot
$ cd ppcboot
$ tar –xvzf //m/ppcboot114.tgz
```

Ampro adds the modifications to PPCBOOT into the source forge baseline, so you can also get the latest version of PPCBOOT from www.sourceforge.net.

In order to compile the PPCBOOT code, simply run "make" on the makefile in the PPCBOOT root directory. This will create the executable image for the PPCBOOT monitor configured for the EnCore PP1 module. An ELF and S-Record file will be created that are suitable for downloading into the flash memory.

PPCBOOT comes pre-installed on the PP1. If you want to upgrade the monitor, you can compile a version with your changes and load it into the flash area. To load the flash area, you have two options.

- You can program the flash device (AMD 29LV160DT) via the JTAG/COP port. Ampro has tested the VisionICE II module and software from WindRiver HIS, and the OC Deamon module and software from Macraigor Systems.
- Otherwise, you could use the PPCBOOT "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 PPCBOOT code. See the command summary in the following section for details, as well as the flash management section under Programming Considerations.

PPCBOOT Setup

PPCBOOT is started as part of the hardware reset process. Turning on the module will cause execution to begin at the reset vector location (0xfff00000) and will start the monitor. After startup, the following prompt will appear on the serial console:

NOTE	The first time you boot the EnCore PP1 module you will get the following error message, due to the empty parameter flash sector on boot up. *** Warning - bad CRC, using default environment***

```
PPCBOOT 1.1.4 (Mar 20 2002 - 14:45:20)

 CPU: MPC8245 Revision 16.20 at 300 MHz:
      16 kB I-Cache 16 kB D-Cache
Board: EnCore PP1 Local Bus at 100 MHz
DRAM:  128 MB
Relocating 7f00000 to 7ebbfa8
FLASH:  2 MB
PCI:   scanning bus for devices ...
Device at Bus: 0 Dev: 24 Func: 0 is preconfigured (80001000) . .OK
In:    serial
Out:   serial
Err:   serial
IDE:   Bus 0: OK
  Device 0: Model: ST34311A Firm: 8.01 Ser#: 5BF2DY6C
             Type: Hard Disk
            Capacity: 4126.9 MB = 4.0 GB (8452080 x 512)
  Device 1: Model: SanDisk SDCFB-48
            Firm: vde 1.10.
            Ser#: 243357J1918
            Type: Removable Hard Disk
            Capacity: 45.8 MB = 0.0 GB (93952 x 512)
Drive 0 MBR 0  Partition  0 Type: 0x6
Drive 0 MBR 0  Partition  1 Type: 0x5
Drive 1 MBR 0  Partition  0 Type: 0x6
Hit any key to stop autoboot:  0
=>
```

Once the monitor is running, the user must configure the hardware setup parameters. These parameters are accessed using the *setenv* command, and will contain the information necessary for running the

EnCore PP1 module. The *printenv* command is used to display the settings of the environmental variables, as shown in the following example:

```
=> printenv
bootcmd=bootm FF820000
bootdelay=5
baudrate=115200
clocks_in_mhz=1
stdin=serial
stdout=serial
stderr=serial

Environment size: 108/16380 bytes
=>
```

The following table describes all of the critical parameters in the system. The user can also define his own symbols that can be accessed via PPCBOOT when running user downloaded programs. All settings will be saved in Flash memory and will only need to be loaded once. Ensure you use the *saveenv* command to save the current settings.

Table 4-1. PPCBOOT Critical Parameter List

Name	Description	Default Value
autoload	If set to "no" (any string beginning with 'n'), "bootp" will just perform a lookup of the configuration from the BOOTP server, but not try to load any image using TFTP	NO (NULL)
autoscript	This indicates that an autoscript function is available. Autoscript allows a remote host to download a command file and, optionally, binary data for automatically updating the target. For example, you create a new kernel image and want to be able to download the image and the machine does the rest. The kernel image is postprocessed with mkimage, which creates an image with a script file prepended. If enabled, autoscript will verify the script and contents of the download and execute the script portion. This would be responsible for erasing flash, copying the new image, and rebooting the machine.	NO (NULL)
autostart	If set to "yes", an image loaded using the "bootp", "rarpboot", "tftpboot" or "diskboot" commands will be automatically started (by internally calling "bootm").	NO (NULL)
baudrate	Baud rate of the serial console	115200
bootaddr	Entry address for boot files. This is where execution starts.	(NULL)
bootargs	Boot arguments when booting an RTOS image. Typically used for VxWorks Bootline.	(NULL)
bootcmd	This contains the default command for booting, typically a bootm or bootvx command.	bootm ffe02000
bootdelay	The number of seconds to delay before executing the default boot command	5
bootdelaykey	If defined, the specific key required to stop the boot delay countdown and break to the PPCBOOT prompt	(NULL)
bootdevice	Identifies the specific boot device number, e.g. for IDE, 0 would be the master and 1 would be the slave.	(NULL)
bootfile	The full path and file name of the file to boot. Used by tftpboot, bootp, and bootfile command	(NULL)

Name	Description	Default Value
bootretry	If positive, the number of seconds of inactivity before the system will retry the default boot sequence.	(NULL)
bootstopkey	The specific key required to stop the boot process after the delay timer is timed out.	(NULL)
clocks_in_mhz	PPCBOOT stores all clock information in Hz internally. For binary compatibility with older Linux kernels (which expect the clocks passed in the bd_info data to be in MHz) the environment variable "clocks_in_mhz" can be defined so that PPCBOOT converts clock data to MHZ before passing it to the Linux kernel.	1
flashchecksum	If set to “y”, will compute and print the checksum of the flash on bootup	(NULL)
gatewayip	IP address of the local gateway. Specified in xx.xx.xx.xx form	(NULL)
hostname	Name of the target. This is used in DHCP protocol as the target’s host name.	(NULL)
initrd_high	If this variable is not set, initrd images will be copied to the highest possible address in RAM; this is usually what you want since it allows for maximum initrd size. If for some reason you want to make sure that the initrd image is loaded below the CFG_BOOTMAPSZ limit, you can set this environment variable to a value of "no" or "off" or "0". Alternatively, you can set it to a maximum upper address to use (PPCBOOT will still check that it does not overwrite the PPCBOOT stack and data). For instance, when you have a system with 16 MB RAM, and want to reseve 4 MB from use by Linux, you can do this by adding "mem=12M" to the value of the "bootargs" variable. However, now you must make sure, that the initrd image is placed in the firs 12 MB as well - this can be done with setenv initrd_high 00c00000	(NULL)
ipaddr	IP address of the target.	(NULL)
loadaddr	Address where files are loaded to for processing.	(NULL)
loads_echo	Indicates if serial download characters need to be echoed back to the host.	(NULL)
netmask	Net Mask for determining the scope of the local subnet. In dotted decimal form.	(NULL)
pci_listing	Indicates if PCI information listings are to be short form or long form.	(NULL)
preboot	If defined, this command will be executed prior to the boot delay period	(NULL)
serverip	IP address of the tftp server, in dotted decimal form	(NULL)
stderr	Indicates the type of port for the stderr output.	serial
stdin	Indicates the type of port for the stdin output.	serial
stdout	Indicates the type of port for the stdout output.	serial
verify	Indicates to autoscript that the contents of the script need to be verified.	(NULL)

Loading and Execution

User written programs can be loaded and executed via the PPCBOOT monitor. PPCBOOT functions (such as interrupt handler or exception handler installation) can be accessed through a function call interface available at run time. Simple programs can be executed, and will return to the monitor prompt. They are typically loaded using a boot or load command, and executed (with parameters) using the go command.

An example program is provided as part of the PPCBOOT distribution. This file is called "Hello", and provides an implementation of the Hello World program. Study this makefile as an example for downloading your own code.

Command Summary

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

```
=> help <command>
```

Where <command> is the command of interest. The following table provides a full description of the command and all options.

Table 4-2. Command Description Table

Command	Description/Syntax
autoscr	autoscr [addr] addr - run script starting at addr A valid autoscr header must be present. This allows the user to put a script into flash memory that can be executed upon bootup
base	base [off] off – offset for load commands This will set or display an offset that is applied to memory and load commands. Normally set to zero.
bdinfo	bdinfo Prints module information to the console, including the addresses of functions available to applications via the PPCBOOT interface.
bootelf	bootelf [address] address – address of ELF file in memory Processes the ELF file located at the indicated location in memory. • If address is not specified, the environmental variable loadaddr is used. This will move the executable file into its proper location, and start execution at the start symbol indicated in the ELF file.
bootp	bootp [loadaddress] [bootfilename] loadaddress – address to load the file into bootfilename – path of the file to load. Usually in the /tftpboot directory of the host. This command performs the BOOTP protocol for attaining an IP address, and booting a remotely located file via TFTP. The environmental variable netmask must be specified. • If loadaddress and bootfilename are not specified, the environmental variables loadaddr and bootfile are used. • If the environmental variable autostart is yes, then the file is booted using the bootm command. This requires the file to be encapsulated into the PPCBOOT image format using mkimage.

Command	Description/Syntax
bootvx	bootvx [loadaddr] bootvx tftp bootvx dhcp bootvx file loadaddr – load address where the vxWorks ELF image is located This will load the vxWorks ELF image into it's execution location, set the bootline specified in the environmental variable bootargs into the proper location, and start execution at the vxWorks start vector. The TFTP form of the command will perform a tftp download of the image, and start execution. For the TFTP form, the environmental variables ipaddr, serverip, gatewayip, netmask, bootfile, loadaddr, and bootargs must all be properly set. The DHCP form of the command will perform a complete DHCP client request. If bootfile is defined, then that is the file that will be loaded. If it is not defined, the file specified in the dhcpd.conf file (or equivalent) shall be used. Finally, if no file is specified, a file name created from the MAC address shall be used. The IP address, Host address, and Netmask shall be appended to any arguments defined in bootargs. The FILE form of the command will load VxWorks from an attached storage device. The file path is stored in the bootfile argument.
bootd	bootd Equivalent to "run bootcmd", where bootcmd is an environmental variable containing the default boot command.
cmp	cmp [.b, .w, .l] addr1 addr2 count addr1 – first address to compare addr2 – second address to compare count – number of locations to compare Compares memory in byte, word, or long word units. Indicates locations of miscompares.
coninfo	coninfo Prints available console device names to the current console.
cp	cp [.b, .w, .l] source target count source – source address for the copy target – target address for the copy count – number of locations to copy Copies the number of locations in byte, word, or long word chunks, from source to target.
crc32	crc32 address count address – start address for calculation count – number of long words to include Calculates the CRC32 checksum of the indicated memory contents, and displays it on the console output.
dcache	dcache on|off Enables or disables data cache
dhcp	dhcp Performs the DHCP client protocol to obtain IP/boot parameters. The Ethernet address of the network controller must be set.

Command	Description/Syntax
diskboot	diskboot loadaddr dev:part loadaddr – address to load image into dev – device number of IDE device (0=master, 1=slave) part – partition number to boot from This function will load the boot record from the partition indicated. The executable file must be encapsulated into the PPCBOOT image header. • If autostart is enabled, then the bootm command will be executed at the load address. • If loadaddr is not specified, then the environmental variable loadaddr will be used. **NOTE** The executable image must be located in the boot record of the partition, not in the file system itself. See file boot for booting from a file system.
echo	echo \c [args] args – argument string to echo Echos the args string to the console. The \c switch suppresses new lines
eeprom	eeprom read devaddr addr off cnt eeprom write devaddr addr off cnt devaddr – address of EEPROM on I^2C bus addr – address in memory to store/get contents for read/write off – address in EEPROM to read/write cnd – number of bytes to transfer This command will transfer data to/from serial EEPROM devices located on the I^2C bus. For the EnCore PP1, there is a SEEP located at 0xa8, and the SPD SEEP is located at 0xa0

file	file read <path> <addr>
	file write <path> <addr> length
	file append <path> <addr> length
	file create <path>
	file remove <path>
	file boot <path> <type> <addr> <entryaddr>
	path – file path (relative to default)
	addr – address in memory to store/get data for read/write
	length -- length of data to write from memory
	type -- type of file to boot (ELF,VXELF,BIN,SREC)
	entryaddr -- entry address for binary files
	Perform file functions on FAT12/16/32 volumes. The path variable is specified relative the current default path (see cd). Fully specified path variables can also be provided, and devices/partitions can also be specified. A double slash (//) indicates a device string, and a single slash at the beginning of a file path indicates a path relative to the root of the current default device.
	Device strings are of the form
	hd<a|b><part>
	where a or b indicate master or slave IDE device, and part indicates a specific partition. If part is omitted, it is assumed to be zero.
	Thus, the zero partition on the master device is specified as:
	//had and partition 1 for example on the slave device is:
	//hdb1

Command	Description/Syntax
	The following functions are available: Read – This reads the file from disk into memory starting at the given address. This function reads the entire file. Write – This writes data into the file, OVERWRITING EXISTING DATA. Length gives the amount of data to write to the file. Append – This appends data to the end of the given file. Length gives the amount of data to write to the file. Create – This creates a new file in the given directory. The file has 0 length. Remove – This removes the indicated file, freeing all associated allocated blocks. Boot – This loads and executes a file from disk. Used to boot an OS or application directly from the disk. Supported object formats are ELF, VXELF (vxworks format), SREC (S-Record), and BIN (binary). ELF and VXELF files must have a load address specified, and BIN files must also have an entry address specified.
ls	ls <path> path – file path of directory to display Displays a listing of files in the indicated directory file on the filesystem. If path is omitted, the default path is displayed.
cd	cd <path> path – file path to make the default Change the current default file path. The new path can be fully specified, or can be relative to the current default path.
more	more [-h] path path – file path of a file to display on the console output. This command will display a file to the console output, and optionally perform a hex dump of the file (-h switch). The path is relative to the current default path.
erase	erase start end erase N:SF[-SL] erase bank N erase all start – flash start address end – flash end address N – number of sectors SF – start sector SL – end sector This command will erase the indicated areas of flash memory. Flash must be erased prior to re-programming. PPCBOOT has a software protection mechanism for flash memory – see the protect command for details. The area of flash to be erased must have its protection removed prior to erasure. To load a new PPCBOOT image into flash, you must unprotect the area at 0xfff00000, erase it, and load the new flash image using a file, serial, or network load command. **CAUTION** Erasing flash memory without loading a new PPCBOOT image will cause the module to be inoperative the next time is booted. The only way to recover this situation is via a JTAG download of a programming algorithm. Do not erase the PPCBOOT image unless you immediately load a new image.

Command	Description/Syntax
fdcboot	fdcboot loadaddr drive loadaddr – load address to put the image into drive – floppy drive number (must be 0 for EnCore PP1) This command will load an image from the master boot record on the floppy (assuming it is formatted as FAT12) into memory, and if autostart is enabled, process it by using the bootm command. This image must be located in the boot record of the floppy, and it must be encapsulated in the PPCBOOT image header by using the mkimage utility on the development host.
flinfo	flinfo [bank] bank – bank number This command will display the protection information of the flash bank indicated. For the EnCore PP1, there is only 1 bank of flash, so bank number can be omitted.
go	go addr [arg ...] addr – address for start of execution (start symbol) arg – optional argument string that is passed to the application Once an executable image is loaded into memory, it can be executed using the go command. This command will pass pointers in the r3, r4, and r5 registers to the application that contains the module info structure, argc, and argv pointers respectively. The application can use PPCBOOT functions via the module info structure. When the application completes, it has a return address back to PPCBOOT.
help	help [command] command – specific command to display help for This command will print out the general help information to the console. If command is specified, help info specific to the command will be displayed.
i2c	i2c reset i2c scan i2c recv i2c_addr data_addr size i2c send i2c_addr data_addr size i2c rcvs i2c_addr sec_addr data_addr size i2c snds i2c_addr sec_addr data_addr size i2c_addr – address on i2c bus of the device data_addr – address in memory for data storage/retrieval sec_addr – for devices that have several locations, the address within the device size – size of the transfer in bytes This command performs i2c reads/writes to the indicated device, and stores or retrieves data from the data_addr location. The reset will reset the bus and scan will try to determine what devices are present on the bus. For devices that have several internal addresses (such as EEPROM devices), the sec_addr value indicates the internal address. The eeprom command will use these commands internally to perform actual read/write operations.
icache	icache on|off Enable or disable instruction cache

Command	Description/Syntax
ide	ide reset ide info ide device [dev] ide part [dev] ide read addr blk# cnt ide write addr blk# cnt dev – device number (0=master, 1=slave) addr – memory address to hold data being transferred blk# -- logical block number on the ide device cnt – number of blocks to transfer This set of commands provides low level access to the IDE devices. The reset command will reset the IDE controller, where info will display the pertinent information about the attached devices. The part command will display all discovered partitions. The read and write commands will read blocks into memory, or write blocks out, respectively. All block addresses are logical blocks. This assumes if a CHS address is required by the device, it will be translated by the driver. Therefore, no CHS calculation is ever required.
iminfo	iminfo addr [addr ...] addr – address in memory where the image is located. This will print header information for application image starting at address 'addr' in memory; this includes verification of the image contents (magic number, header and payload checksums). Optionally, a list of addresses can be provided.
loadb	loadb [off] [baud] off – offset to apply to the load from “base” baud – baud rate for the download This will download a binary file via the console serial port. Off provides an offset from the “base” defined by the base command. This command uses kermit protocol to perform the serial download. This command will only load the file – it does not perform auto-execution.
loads	loads [off] off – offset to apply to the S-Record file This command will start an S-Record file download via the console serial port, and the current baud rate. There is no protocol with this command – only a direct ASCII dump of the S-Record file. Offset shall be applied if specified.
md	md [.b, .w, .l] address [# of objects] address – address to display # of objects – number of objects to display (size of dump) This command will dump the contents of the indicated memory locations to the system console in a hexadecimal dump format. The dump can be specified in byte, word, or long word format. The default is long word.
mm	mm [.b, .w, .l] address address – address to modify This command will display the current value of the memory on the console, and prompt the user for a new value. After receiving the new value, it will increment the address, and repeat. Use CTRL-C to terminate the command.

Command	Description/Syntax
mtest	mtest [start [end [pattern]]] start – start address for test end – end address for test pattern – optional pattern to put into memory This function will perform a simple RAM test by putting a fixed pattern into memory, and verifying that the memory contains that pattern. Do not test code areas, since this is a "destructive" test.
mw	mw [.b, .w, .l] address value [count] address – starting address for fill value – value to insert into memory count – number of locations to set This function will insert the provided value, in byte, word or long word chunks, into the memory starting at address, for count number of values
nm	nm [.b, .w, .l] address address – address to modify This is identical to mm, except the memory address is not incremented – the same address is changed repeatedly.
pciinfo	pciinfo [bus] bus – PCI bus number to scan This command will scan the indicated PCI bus (default 0), and print the configuration header information for all detected devices to the console output.
printenv	printenv [var] var – specific variable to print This command will print the value of the indicated variable to the console output, or all variables if var is omitted.
protect	protect on|off start end protect on|off N:SF[-SL] protect on|off bank N protect on|off all start – flash start address end – flash end address N – number of sectors SF – start sector SL – end sector This set of commands will manage the protection of the flash sectors. Since the EnCore PP1 has only 1 flash bank, the bank N and all commands are identical. Flash areas must be unprotected prior to being erased and re-programmed.
rarpboot	rarpboot [loadaddress] [bootfilename] loadaddress – load address in memory for the image bootfilename – file path and name for the file being booted This command provides for getting an IP address and loading an image via the RARP protocol. This function will set the serverip, ipaddr, gatewayip, and netmask environmental variables. If autostart is enabled, the image will be booted via the bootm command. This requires the image to be encapsulated in the PPCBOOT header format using the mkimage utility during building the image.

Command	Description/Syntax
reset	reset Performs a CPU reset, and jumps to boot vector
run	run var [var . . .] var – environmental variable containing command string This command will execute the sequence of commands stored in the provided environmental variables. User defined variables can be used.
saveenv	saveenv This command saves the environmental variables to the flash area. Protection need not be disabled to perform the save.
setenv	setenv var value var – environmental variable value – string containing the value for the variable. Spaces can be used in the value. This sets the indicated variable to the provided value. This does not save the variable in persistent storage – see saveenv.
sleep	sleep N N – number of seconds Delays PPCBOOT execution by the indicated number of seconds
tftpboot	tftpboot [loadaddress] [bootfilename] loadaddress – address to load the file into bootfilename – full name and path of the file to load (usually /tftpboot/"filename") This function performs a TFTP download of the provided file into the memory location indicated. If loadaddress or bootfilename are not specified, the variables loadaddr and bootfile are used. The environmental variables serverip, ipaddr, gatewayip, and netmask must all be defined. This command simply loads the file into memory. Subsequent boot or go commands can be performed on this image.
version	Version Print monitor version to console output
?	Alias for 'help'

Operating System Support

The EnCore PP1 EDK comes with support for a popular operating system used on the PowerPC platform, VxWorks by Wind River. Ampro provides a VxWorks Board Support Package (BSP) required to run the operating system on the EnCore PP1 module.

Functionality

The Ampro distribution contains a Board Support Package (BSP) for the Tornado 2.02 for PowerPC 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
- Support for I^2C Bus peripherals

- Boot Line stored in on-board Flash memory (via PPCBOOT)
- Automatic detection of memory size from MCCR registers

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 2.02 in the PowerPC environment, specifically for a PowerPC 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 2.02.

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 2.02 for the PowerPC 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 PP1 platform.

Memory Configuration

PPCBOOT is configured to detect the type of SODIMM module that has been inserted as the main memory system, and configure the memory controller properly to use the memory. This is done by reading the Serial Presence Detect (SPD) EEPROM located on the SODIMM module. The MCCR registers in the MPC8245 processor will be correctly configured for the installed module. If the SPD cannot be read for any reason, the system will default to a memory size of 16MB. This will allow diagnostic software, once operational, to try to determine the reason why the SPD is not readable.

The memory controller supports either 1 or 2 Chip Select type SODIMM memory modules.

Interrupt Configuration

The system interrupt configuration supports interrupts from the MPC8245 EPIC, and the 8259A compatible Programmable Interrupt Controllers (PICs) in the VIA Southbridge. Depending on the operating system, the PIC interrupts may be chained to one interrupt request, or may be split into separate IRQ numbers. The Embedded PIC (or EPIC) has been designed for this configuration, and is normally run in "mixed mode" as opposed to "Pass-through mode". The 4 PCI interrupt lines (A-D) are connected to the remaining external IRQ lines on the MPC8245. A diagram of the interrupt control mechanism is shown in Figure 4-2.

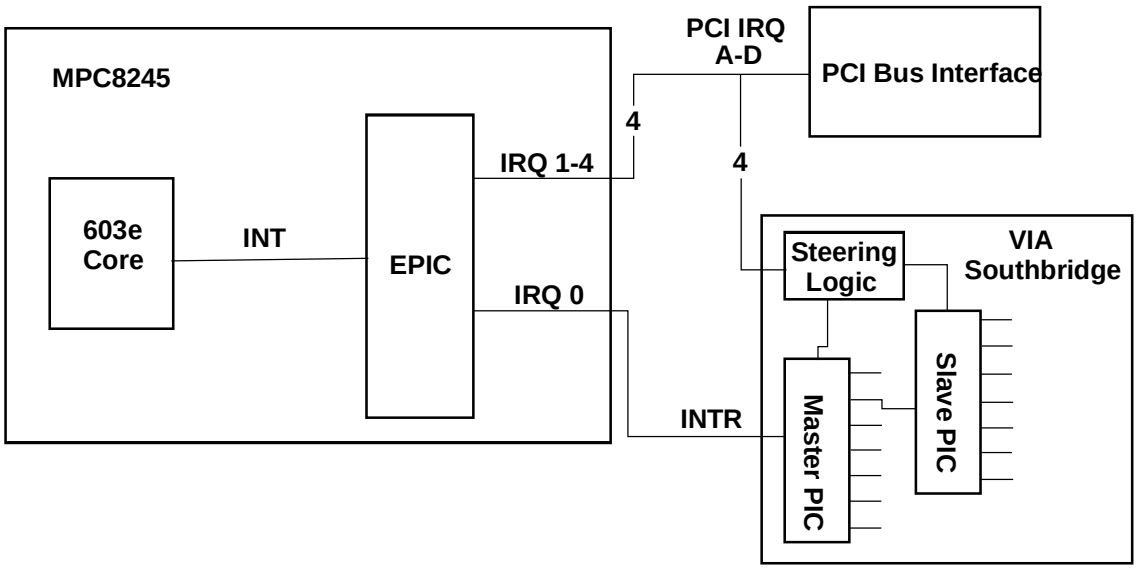

Figure 4-2. EnCore PP1 Interrupt Structure

PCI interrupts can be routed via the PIC, or can be directly processed. Ampro recommends handling all PCI interrupts directly, causing the VIA PCI interrupt steering registers to be left disabled.

PCI Configuration

The PCI interface is implemented internally to the MPC8245 processor by an integrated MPC107 PCI interface that provides standard PCI bus signals to the PCI connector and the VIA Southbridge.

PCI Memory Map

The following table outlines the memory map associated with the PCI bus and the MPC8245. This conforms to Memory Map "B" as described in the MPC8245 user's manual.

Table 4-3. PCI Memory Map

Physical	PCI address	Description
0x00000000-0x7fffffff	0x00000000-0x7fffffff	Local Memory Space. These addresses are reserved for the SDRAM controlled by the MPC8245. Memory masters on the PCI bus can access this local memory space at the same address range.
0x80000000-0xfbffffff	0x80000000-0xfbffffff	PCI Memory Space. PCI devices using memory space will have the MBAR registers allocated from this PCI memory space during the bus scan.
0xfc000000-0xfc0fffff	0xfc000000-0xfc0fffff	Embedded Utilities Memory Block. The EnCore PP1 has configured the EUMB to be located at address 0xfc000000. This block contains the registers for the peripherals integrated into the MPC8245.
0xfc100000-0xfdffffff	0xfc100000-0xfdffffff	Additional PCI memory space
0xfe000000-0xfe00ffff	0xfe000000-0xfe00ffff	PCI Low IO Space. This corresponds to the low 64K of PCI I/O space accesses, mapped directly.
0xfe010000-0xfe7fffff	0xfe010000-0xfe7fffff	RESERVED
0xfe800000-0xfebfffff	0xfe800000-0xfebfffff	Extended I/O Space. This corresponds to a 24 bit I/O space address. This overlaps the low I/O space.

Physical	PCI address	Description
0xfec00000-0xfedfffff	0xfec00000-0xfedfffff	Config Address Space. Writing a configuration space address anywhere within this space sets the address for configuration cycles.
0xfee00000-0xfeefffff	0xfee00000-0xfeefffff	Config I/O Space. Reading or writing to this space will cause configuration cycles to the address specified in the configuration address space.
0xfef00000-0xfeffffff	0xfef00000-0xfeffffff	Interrupt Acknowledge Space.
0xff000000-0xffdfffff	0xff000000-0xffdfffff	Unused by EnCore PP1. Could be used by ROM on PCI bus.
0xffe00000-0xffffffff	0xffe00000-0xffffffff	FLASH Area (2MB)

PCI Configuration Space Access

Configuration space is accessed through a data and address register set. These registers are mapped to a memory region versus a single location, to allow for simplification of some bus scanning software. The configuration address and data registers are 32-bit registers that appear in little endian format to the processor. If there is a requirement for writing a 32-bit address to the configuration space address register, and the processor is running in big endian mode, then the 32-bit word must be endian swapped prior to being written to the address. This can be accomplished by a single assembly instruction, stwbrx. The following assembly code illustrates the usage;

```
/*************************************************************
**  FUNCTION: __pci_config_write_32
**
**  PARAMETERS: r3 = PCI address for configuration registers
**              r4 = data to be written to the configuration
**                   registers
**
**  DESCRIPTION:
**
**  RETURNS:
**
*************************************************************/

    .globl __pci_config_write_32
__pci_config_write_32:
    lis     r5, 0xfec0
    stwbrx   r3, r0, r5
    sync
    lis      r5, 0xfee0
    stwbrx   r4, r0, r5
    sync
    blr
```

There are forms of the instruction for byte and halfword operations, also. Corresponding forms also exist for input with endian swap.

The address of the configuration register is determined by the following set of fields. If the bus number is not 0, type 1 configuration cycles are generated for being forwarded by a bridge, else type 0 cycles are generated.

Table 4-4. Configuration Space Address Formats

Address Bits	Description
31	Enable. 1 indicates configuration cycle, so all configuration register addresses will start with 0x80000000.
30:24	Reserved, 0
23:16	Bus Number
15:11	Device Select, a number between 0 and 31
10:8	Function Select. This selects function number on a multi-function device.
7:2	Register select. This contains the upper 6 bits of the register number. Thus, addresses can only be specified on 32 bit boundaries.
1:0	Reserved. Must be 0

All accesses must be made on 32 bit boundaries. To select a specific byte register, read the 32-bit segment that contains the register, and mask out the appropriate byte. Similarly, to write to a byte register you must read the entire 32-bit block, insert the appropriate bits, and re-write the 32-bit block.

Example: Address for device 2, function 1, register 1C,on bus 1;

```
0x8001111c
```

Memory Space Access

Memory space is directly mapped between the PCI bus and the host processor. Local memory, that is memory controlled by the host processor, is always located in the address range 0x00000000-0x7fffffff. Memory controlled by PCI devices must be assigned an address through the device's Memory Base Address Registers (MBARs, registers 0x10,0x14,0x18,0x1c) in the devices configuration space. These addresses are assigned in the range 0x80000000- 0xfdffffff.

NOTE	The EUMBBAR registers are located at a 1MB region starting at 0xfc000000, and cannot be used by PCI peripherals.

Once the MBAR registers of all of the devices are set, the memory can be accessed directly at the set location, with no address translation.

I/O Space Access

I/O space is mapped to the region 0xfe000000 for the 64K (16-bit) memory addressing, and to 0xfe800000 for 24 bit addressing. Most of the time the 64K address space is used for compatibility with older software.

NOTE	Some address space does overlap, specifically 0xfe000000 and 0xfe800000, so only devices using I/O addresses greater than 16 bits should use the extended I/O address.

The I/O space addresses are translated by the relationship 0xfe000000 (mem) = 0x00000000 (IO), so all I/O addresses are offset by 0xfe000000. Performing direct memory reads/writes to this region will cause I/O cycles on the PCI bus.

Interrupt Acknowledge Access

This space is used to acknowledge interrupt controllers on the slave devices. This is normally not used, but is mapped as in the above memory map.

EnCore PP1 Peripherals

The EnCore PP1 contains a fixed set of peripherals on PCI bus 0. The following table outlines the configuration addresses.

Table 4-5. Peripheral Bus 0 Addresses

Device	Function	Description
0	0	PCI Host interface controller (MPC107)
22	0	PCI to ISA Bridge Controller. Used to access the super I/O functions of the VIA chipset.
22	1	IDE/ATA controller. Provides access to the IDE interface.
22	2	USB UHCI compatible controller, ports 0 and 1
22	3	USB UHCI compatible controller, ports 2 and 3
22	4	Power Control functions, SM Bus access
22	5	AC97 Digital Audio interface
22	6	MC97 Analog Audio interface
24	0	82559er Network Controller

Floppy/Parallel Configuration

The EnCore PP1 contains a standard Southbridge (VIA VT82C686B), which provides several I/O functions not normally found with PowerPC processors. One of these configurations is the Floppy/Parallel controller. In the EnCore PP1 implementation, these two functions share the same pins on the EnCore interface. Therefore, you cannot use the floppy and parallel interfaces at the same time. To select the proper function, ensure you configure the VIA chip properly.

An example program is included on the EnCore EDK CD-ROM under the PPCBOOT/Examples directory. The VxWorks BSP has a completely operational driver for the floppy disk in this configuration.

Flash Management

Managing the contents of the FLASH device can be a tricky undertaking. The flash is divided into 3 main sections –

1. PPCBOOT parameter storage
2. PPCBOOT Code
3. Parameter flash area.

Each of these areas has specific contents and use;

- PPCBOOT Parameter Storage — This area contains the environmental variables used by PPCBOOT. Only about the low 4K is used.
- PPCBOOT Code — This area contains the executable PPCBOOT image. This image is generally about 128-256K in size, and must be located at system address 0xFFF00100. Execution starts at 0xFFF00100.
- Parameter Flash — This is a protected area of Flash, and should not be modified. This contains module specific information, such as serial number and module revision.

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

Table 4-6. Flash Memory Map

Sys Address	Flash Address	Sectors	Description
0xffe00000 – 0xffefffff	0x00000000 – 0x000fffff	0	PPCBOOT Environment Variables
0xffe10000 – 0xffefffff	0x00010000 – 0x0001ffff	1-15	Free space
0xfff00000 – 0xfff2ffff	0x00100000 – 0x001fffff	16-18	PPCBOOT Executable Code
0xfff30000 – 0xfffbffff	0x00130000 – 0x001fbfff	19-33	Free space
0xffffc000 – 0xffffffff	0x001fc000 – 0x001fffff	34	Parameter Storage (16 K)

The AMD29F160BT device has a feature called the "boot sector". In this device, the last 4 sectors are smaller than the baseline 64KB per sector. Sector 31 is 32KB, sector 32 and 33 are 8KB, and sector 34 is 16KB. This allows for smaller sections of data to be stored without having to manage a larger 64K 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.

Programming the Flash

Use the following procedure to program the flash.

CAUTION	The following procedure, if not followed correctly, could leave the module inoperable. Ensure your Boot image is valid before attempting to program the flash. Perform all steps in order – Do Not Omit Any!!

1. Ensure you have a viable flash image you can download into the processor.

   This image must be in a raw binary format.

2. Perform a tftpboot operation, loading the binary image into a convenient area of RAM.

```
setenv bootfile/tftpboot/ppcboot.bin
tftp
```

NOTE	Ensure the environmental variable, *bootfile*, is set correctly.

3. Verify the correct information is located at the RAM address by doing memory dumps.

NOTE	The binary image will have the actual start code offset into the image by 0x100 bytes. The string at the beginning of the image will give the revision and date that PPCBOOT was compiled.

4. Disable protection of the flash by issuing the disable command.

```
protect off fff00000 ffffbfff
```

5. Erase the flash sectors from 0xfff00000 to 0xffffbfff.

```
erase fff00000 ffffbfff
```

6. Copy the valid image to the flash.

   Assuming the tftp image was loaded at location 0x800000, and the size of the file is 0x1fc00, the command is

   ```
   cp.b 800000 fff00000 1fc00
   ```

7. Reboot the module.

   The new flash image should now be operating correctly.

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't 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.

  http://www.ampro.com/scripts/virtual_technician.exe/people

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, Inc., 5215 Hellyer Avenue, San Jose, CA 95138-1007, USA

Index