User's Guide

Publication number E2476-97005
July 1999

For Safety information, Warranties, and Regulatory information, see the pages behind the index.

Solutions for the Motorola Embedded PowerPC MPC860/821

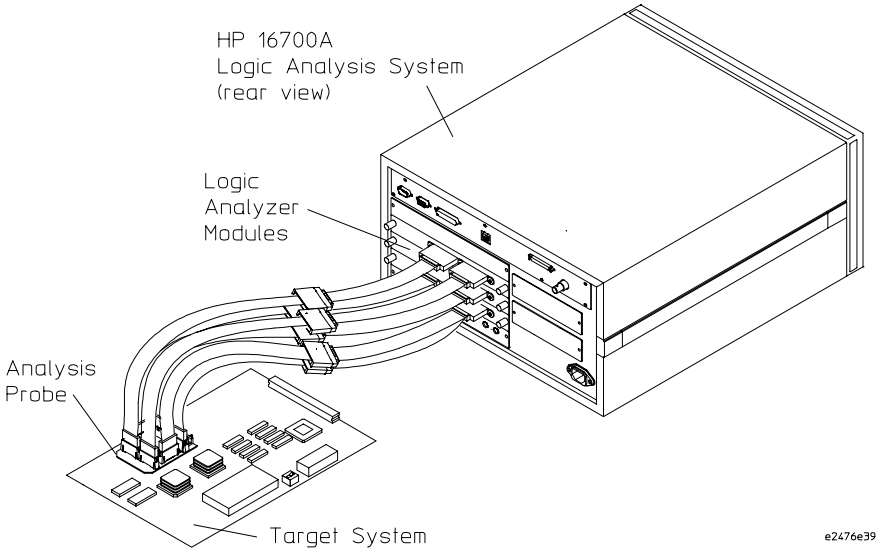
HP Solutions for the Motorola Embedded PowerPC MPC860/821—At a Glance

This manual describes several ways to connect an HP logic analysis system to your target system. These connections use an analysis probe (or custom probing), plus an emulation module (for an emulation solution).

Analysis Probe

The analysis probe connects your logic analyzer to your target system for state and timing analysis. The analysis probe can be used with an HP 16600A/700A-series logic analysis system or with other HP logic analyzers.

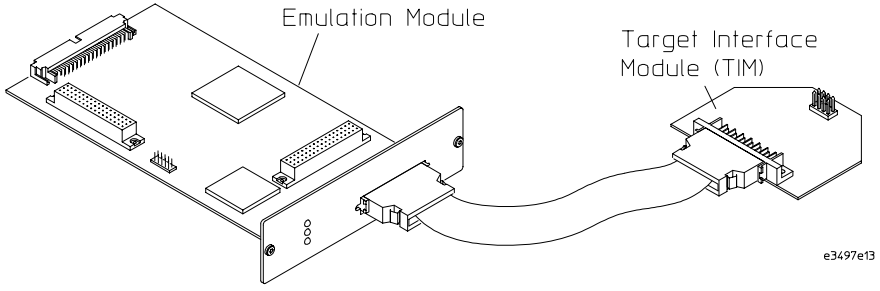
The analysis probe can be purchased alone, or as part of an emulation solution.



If your target system has the appropriate connectors, you can connect the logic analyzer directly to the target system and use the inverse assembler without the analysis probe.

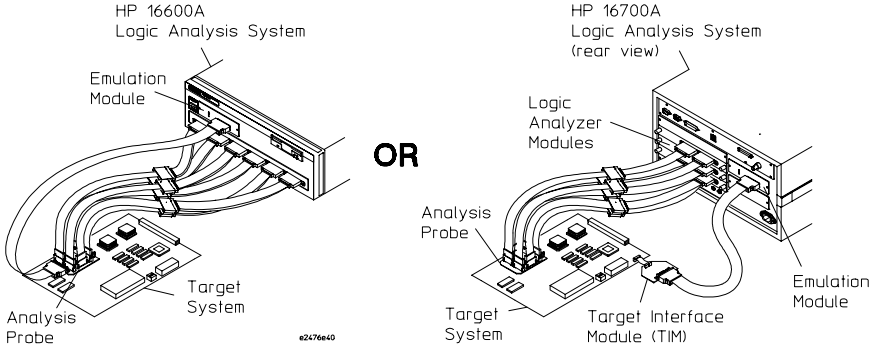
Emulation Module and Target Interface Module

The emulation module plugs into your HP 16600A/700A-series logic analysis system frame. The emulation module lets you use the target processor's built-in background debugging features, including run control and access to registers and memory. A high-level source debugger can use the emulation module to debug code running on the target system. You can connect the emulation module to the analysis probe or you can connect it to a debug port on the target system through the provided target interface module (TIM).



Emulation Solution

The emulation solution includes an analysis probe, an emulation module, cables and adapters, and the HP B4620B Source Correlation Tool Set (for analyzing high-level source code). This solution is designed to be used with an HP 16600A/700A-series logic analysis system.



In This Book

This book documents the following products:

Analysis Probe

Processors supported	Product ordered	Includes
MPC860/821 Up to 50 MHz 357-pin BGA package	HP E9584A Option #002	HP E2476A BGA analysis probe and inverse assembler / execution tracker
MPC860/821 any package, custom probing	HP E9584A Option #001	HP E2477A inverse assembler / execution tracker

Emulation Solution

Processors supported	Product ordered	Includes
MPC860/821 Up to 50 MHz 357-pin BGA package	HP E9484A Option #002	HP E2476A BGA analysis probe, inverse assembler / execution tracker, HP 16610A emulation module, target interface module (TIM), HP B4620B Source Correlation Tool Set
MPC860/821 any package, custom probing	HP E9484A Option #001	HP E2477A inverse assembler / execution tracker, HP 16610A emulation module, target interface module (TIM), HP B4620B Source Correlation Tool Set

Additional Information Sources

Additional or updated information can be found in the following places:

Newer editions of this manual may be available. Contact your local HP representative.

If you have a probing adapter, the instructions for connecting the probe to your target microcontroller are in the **Probing Adapter** documentation.

Application notes may be available from your local HP representative or on the World Wide Web at:

http://www.hp.com/go/logicanalyzer

If you have an HP 16600A or HP 16700A logic analysis system, the **online help** for the Emulation Control Interface has additional information on using the emulation module.

The **measurement examples** include valuable tips for making emulation and analysis measurements. You can find the measurement examples under the system help in your HP 16600A/700A logic analysis system.

If you cannot easily find the information you need, send email to documentation@col.hp.com. Your comments will help HP improve future manuals. (This address is for comments only; contact your local HP representative if you need technical support.)

Contents

Contents

Contents

Contents

Contents

1

Overview

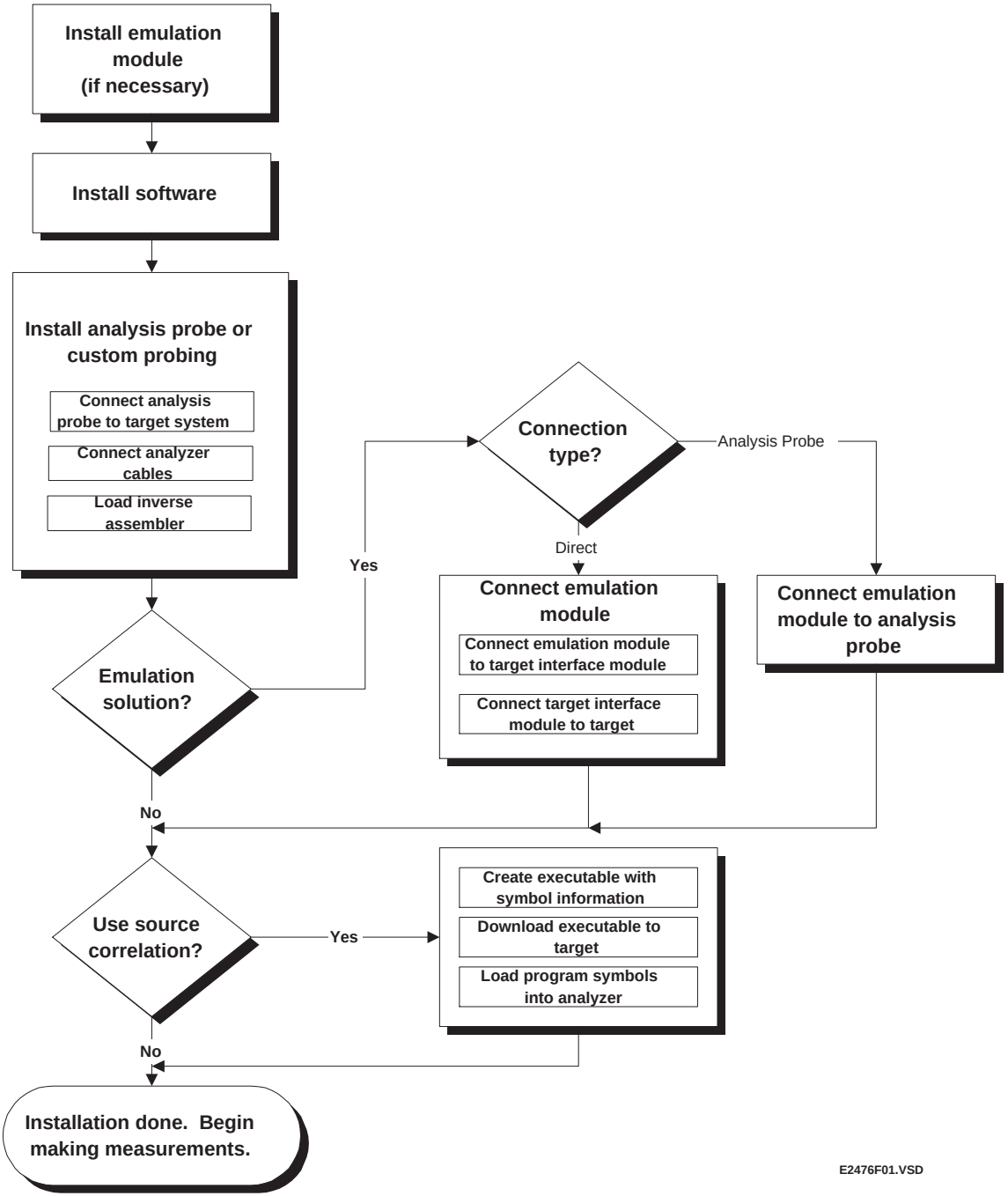
This chapter describes:

- Setup Checklist
- Setup Assistant
- Equipment used with the analysis probe (including a list of logic analyzers supported)
- Equipment used with the emulation module
- System configurations
- Additional information sources

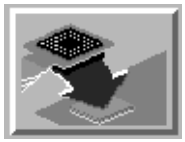
Setup Checklist

Follow these steps to connect your equipment:

- Check that you received all of the necessary equipment. See page 317 and page 30.
- If you need to install an emulation module in an HP 16600A/700A series logic analysis system, see page 164.
- Install the software. See page 35.
- If you have an HP 16600A/700A-series logic analysis system, use the Setup Assistant to help you connect and configure the analysis probe and emulation module. See page 23.
- If you do not have an HP 16600A/700A-series logic analysis system, follow the instructions beginning on page 43.


Install emulation module (if necessary)
Install software
Install analysis probe or custom probing
Connect analysis probe to target system
Connect analyzer cables
Load inverse assembler
Emulation solution?
Yes
Connection type?
Analysis Probe
Direct
Connect emulation module
Connect emulation module to target interface module
Connect target interface module to target
Connect emulation module to analysis probe
No
Use source correlation?
Yes
Create executable with symbol information
Download executable to target
Load program symbols into analyzer
No
Installation done. Begin making measurements.
E2476F01.VSD

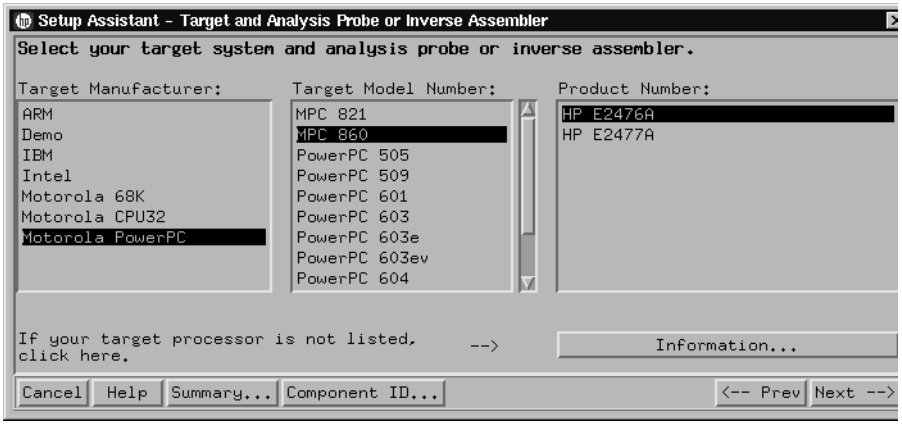
Setup Assistant



The Setup Assistant is an online tool for connecting and configuring your logic analysis system for microprocessor and bus analysis. The Setup Assistant is available on the HP 16600A and HP 16700A-series logic analysis systems. You can use the Setup Assistant in place of the connection and configuration procedures provided in this manual.

This menu-driven tool will guide you through the connection procedures for connecting the logic analyzer to an analysis probe, an emulation module, or other supported equipment. It will also guide you through connecting an analysis probe to the target system.

Start the Setup Assistant by clicking its icon in the system window.



If you ordered this analysis probe or emulation solution with your HP 16600A/700A-series logic analysis system, the logic analysis system has the latest software installed, including support for this product. If you received this product after you received your logic analysis system, see Chapter 2, “Installing Software,” beginning on page 35.

Analysis Probe

This section lists equipment supplied with the analysis probe and equipment requirements for using the analysis probe.

Equipment supplied

The equipment supplied with the analysis probe is shown in the illustration on the next page. It is listed below:

HP E2476A Analysis Probe

- The HP E2476A analysis probe circuit board, which includes a pre-installed male-to-male header and a BGA carrier.
- An extender to protect target board and analysis probe sockets (one is pre-installed on the analysis probe socket; you can install another one on your target system).
- The HP E5355A BGA probing kit, which includes installation instructions.
- Three HP E5346A high-density termination cables.
- Logic analyzer configuration files, the inverse assembler with the cache-on trace reconstruction software on a CD ROM (for HP 16600A/700A series logic analysis systems).
- Logic analyzer configuration files, the inverse assembler, and the cache-on execution tracker software on 3.5-inch disks (for other HP logic analyzers).
- The inverse assembler and cache-on execution tracker on two 3.5-inch disks (for the HP 16505A prototype analyzer).
- This User's Guide.

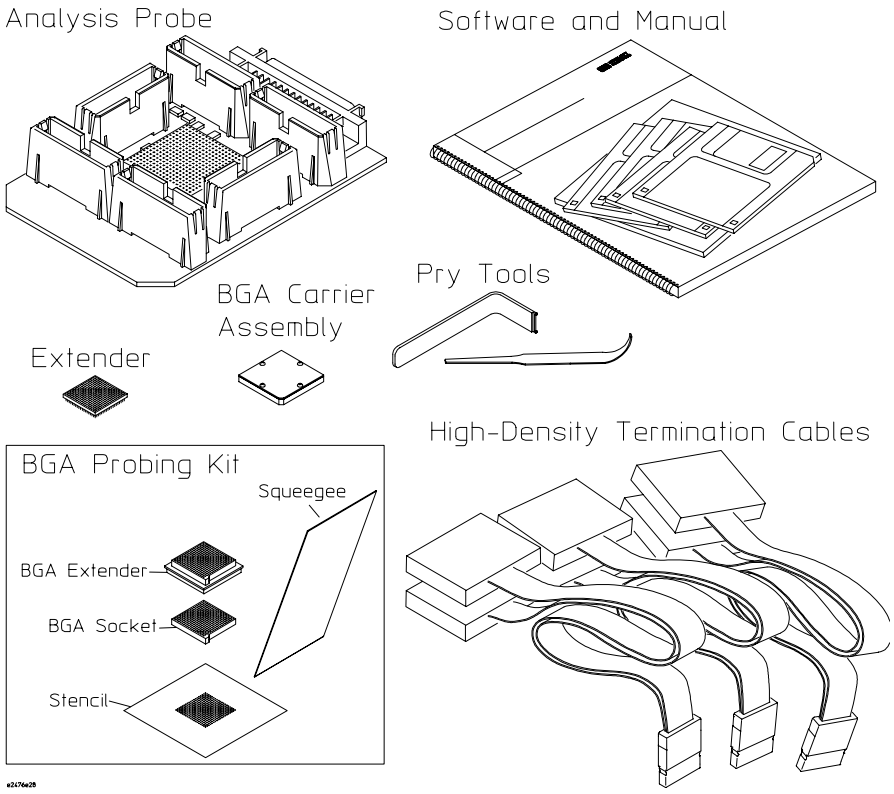
HP E2477A Inverse Assembler Software

If you ordered the HP E2477A inverse assembler, only the software and manuals are supplied.

- Logic analyzer configuration files, the inverse assembler with the cache-on trace reconstruction software on a CD ROM (for HP 16600A/700A series

logic analysis systems).

- Logic analyzer configuration files, the inverse assembler, and the cache-on execution tracker software on 3.5-inch disks (for other HP logic analyzers).
- This User's Guide.



Equipment Supplied with the HP E2476A Analysis Probe

Minimum equipment required

For state and timing analysis of an MPC860/821 target system, you need all of the following items.

- The HP E2476A Analysis Probe. If you are using the HP E2477A Inverse Assembler software, you will also need the appropriate connectors on the target system. See page 85 for information on designing the appropriate connectors into the target system.
- A target system with an empty BGA socket, for the HP E5355A BGA Probing Kit. A BGA microprocessor is also required.
- One of the logic analyzers listed on page 27. The logic analyzer software version requirements are listed on page 29.
- For cache-on execution tracking, a supported logic analyzer with the required minimum software version as listed on page 29. The HP B4620B Source Correlation Tool Set is also highly recommended for correlating cache data with code execution.

Additional equipment supported

Emulation module

The HP E2476A has a built-in connector for an HP 16610A emulation module.

HP B4620B Source Correlation Tool Set

The analysis probe and inverse assembler may be used with the HP B4620B Source Correlation Tool Set.

Logic analyzers supported

The following table lists the logic analyzers supported by the HP E2476A analysis probe and HP E2477A software. Logic analyzer software version requirements are shown on the page following the table.

The HP E2476A and HP E2477A require six logic analyzer pods (102 channels) for inverse assembly. The analysis probe contains six additional pods that you can monitor.

Logic Analyzers Supported

Logic Analyzer	Channel Count	State Speed	Timing Speed	Memory Depth
16710A (1 or 2 cards)	102/card	100 MHz	250 MHz	8 k states
16711A (1 or 2 cards)	102/card	100 MHz	250 MHz	32 k states
16712A (1 or 2 cards)	102/card	100 MHz	250 MHz	128 k states
16715A (2 or 3 cards)	68/card	167 MHz	333 MHz	2 M states
16716A (2 or 3 cards)	68/card	167 MHz	333 MHz	512 k states
16717A (2 or 3 cards)	68/card	167 MHz	333 MHz	2 M states
16600A	204	100 MHz	125 MHz	64 k states
16601A	136	100 MHz	125 MHz	64 k states
16602A	102	100 MHz	125 MHz	64 k states
16550A (1 or 2 cards)	102/card	100 MHz	250 MHz	4 k states
16554A (2 or 3 cards)	68/card	70 MHz	125 MHz	512 k states
16555A (2 or 3 cards)	68/card	110 MHz	250 MHz	1 M states
16555D (2 or 3 cards)	68/card	110 MHz	250 MHz	2 M states
16556A (2 or 3 cards)	68/card	100 MHz	200 MHz	1 M states
16556D (2 or 3 cards)	68/card	100 MHz	200 MHz	2 M states
16557D (2 or 3 cards)	68/card	135 MHz	250 MHz	2 M states
1660A/AS/C/CS/CP/E/ES/EP	136	100 MHz	250 MHz	4 k states
1661A/AS/C/CS/CP/E/ES/EP	102	100 MHz	250 MHz	4 k states
1670A	136	70 MHz	125 MHz	64 k or .5 M states
1670D	136	100 MHz	125 MHz	64 k or 1 M states
1671A	102	70 MHz	125 MHz	64 k or .5 M
1671D	102	100 MHz	125 MHz	64 k or 1 M
1670E	136	100 MHz	125 MHz	1 M
1671E	102	100 MHz	125 MHz	1 M

Logic analyzer software version requirements

The logic analyzers must have software with a version number greater than or equal to those listed below to make a measurement with the HP E2476/7A. You can obtain the latest software at the following web site:

http://www.hp.com/go/logicanalyzer

To use the cache-on execution tracker, use the far right column for minimum software requirement. If your software version is older than those listed, load new system software with the higher version numbers before loading the HP E2476/7A software.

Logic Analyzer Software Version Requirements

Logic Analyzer	Minimum Logic Analyzer Software Version for use with HP E2476A/77A	Minimum Logic Analyzer Software Version for Cache-on Execution Tracker (I860ET)
HP 16600A-series	The latest HP 16600A logic analyzer software version is on the CD-ROM shipped with this product.	
HP 1660A/AS Series	A.02.01	na
HP 1660C/CS/CP Series	A.02.01	na
HP 1660E/ES/EP Series	A.02.01	na
HP 1670A/D/E Series	A.02.01	na
Mainframes*		
HP 16700A-series	The latest HP 16700A logic analyzer software version is on the CD-ROM shipped with this product.	
HP 16500C Mainframe	A.01.05	A.01.05**
HP 16500B Mainframe	A.03.14	A.03.14**

* The mainframes are used with logic analyzer modules such as the HP 16557D logic analyzer.
** The HP 16500B/C mainframes require the HP 16505A prototype analyzer to use the cache-on execution tracker (I860ET). The HP 16505A requires software version A.01.30 or higher.

The HP 16505A provides a windowed user interface for the HP 16500B/C logic analysis system. Refer to the *HP 16505A Prototype Analyzer Installation Guide* for information on connecting the HP 16505A to the HP 16500B/C logic analysis system.

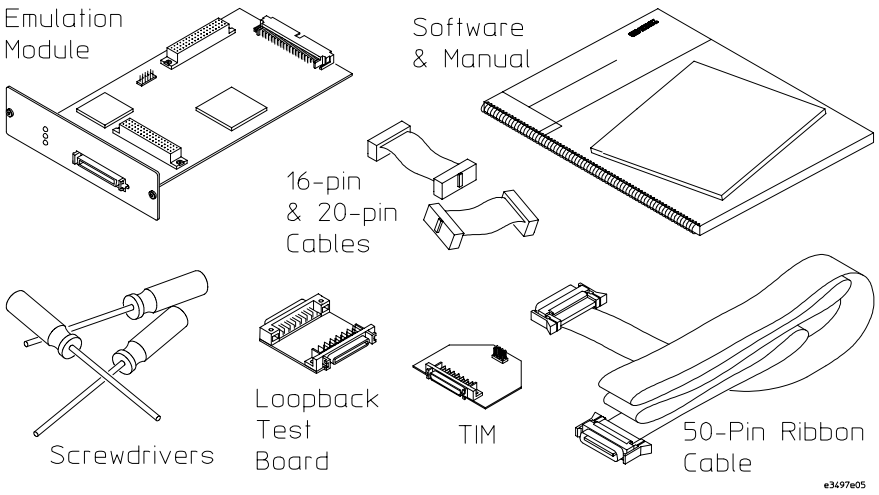
Emulation Module

This section lists equipment supplied with the emulation module and lists the minimum equipment required to use the emulation module.

Equipment supplied

The equipment supplied with your emulation module includes:

- An HP 16610A emulation module. If you ordered an emulation module as part of your HP 16600A or HP 16700A-series logic analysis system, it is already installed in the frame.
- A target interface module (TIM) circuit board.
- An emulation module loopback test board (HP part number E3496-66502).
- Firmware for the emulation module and/or updated software for the Emulation Control Interface on a CD-ROM.
- A 50-pin ribbon cable for connecting the emulation module to the target interface module or the HP E2476A Analysis Probe.
- A 10-pin ribbon cable for connecting the target interface module to the target system.
- One Torx T-10, one Torx T-15, and one T-8 screwdriver.
- This User's Guide.



Equipment Supplied with the Emulation Module

Minimum equipment required

The following equipment is required to use the emulation module:

- A method for connecting to the target system. The HP E2476A analysis probe provides a debug port connector. You can also design a debug port connector on the target system. Chapter 8 provides information on designing a debug port on the target system.
- An HP 16600A or HP 16700A logic analysis system.
- A user interface, such as a high-level source debugger or the logic analysis system's Emulation Control Interface.

Emulation Solution

An emulation solution uses the equipment and software already described in this chapter.

The combination of an analysis probe, an emulation module, and an HP 16600A or HP 16700A logic analysis system lets you both trace and control microprocessor activity on the target system.

The analysis probe supplies signals from the target microprocessor to the logic analyzer. A configuration file sets up the logic analyzer to properly interpret these signals.

You can use a debugger or the logic analysis system's Emulation Control Interface to configure and control the target processor and to download program code.

Additional Information Sources

Additional or updated information can be found in the following places:

Newer editions of this manual may be available. Contact your local HP representative.

If you have a probing adapter, the instructions for connecting the probe to your target microprocessor are in the **Probing Adapter** documentation.

Application notes may be available from your local HP representative or on the World Wide Web at:

http://www.hp.com/go/logicanalyzer

If you have an HP 16600A or HP 16700A logic analysis system, the **online help** for the Emulation Control Interface has additional information on using the emulation module.

The **measurement examples** include valuable tips for making emulation and analysis measurements. You can find the measurement examples in the online help in your HP 16600A/700A logic analysis system.

If you cannot easily find the information you need, send email to **documentation@col.hp.com**. Your comments will help HP improve future manuals. (This address is for comments only; contact your local HP representative if you need technical support.)

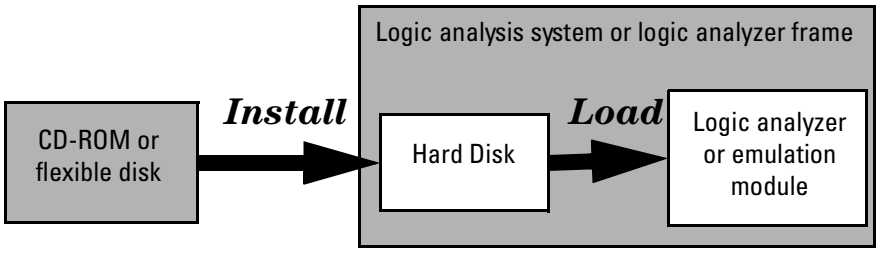
2

Installing Software

This chapter explains how to install the software you will need for your analysis probe or emulation solution.

Installing and loading

Installing the software will copy the files to the hard disk of your logic analysis system. Later, you will need to **load** some of the files into the appropriate measurement module.



What needs to be installed

HP 16600A/700A-series logic analysis systems

If you ordered an analysis probe or emulation solution with your logic analysis system, the software was installed at the factory.

The following files are installed when you install a processor support package from the CD-ROM:

- Logic analysis system configuration files
- Inverse assembler (automatically loaded with the configuration files)
- Personality files for the Setup Assistant
- Emulation module firmware (for emulation solutions)
- Emulation Control Interface (for emulation solutions)

The HP B4620B Source Correlation Tool Set is installed with the logic analysis system's operating system.

Other HP logic analyzers

The following files can be installed from a floppy disk:

- Logic analyzer configuration files, which automatically load the inverse assembler

To install the software from CD-ROM (HP 16600A/700A)

Installing a processor support package from a CD-ROM will take just a few minutes. If the processor support package requires an update to the HP 16600A/700A operating system, installation may take approximately 15 minutes.

If the CD-ROM drive is not connected, see the instructions printed on the CD-ROM package.

1. Turn on the CD-ROM drive first and then turn on the logic analysis system.

   If the CD-ROM and analysis system are already turned on, be sure to save any acquired data. The installation process may reboot the logic analysis system.

2. Insert the CD-ROM in the drive.
3. Click the **System Admin** icon.
4. Click the **Software Install** tab.
5. Click **Install...**.

   Change the media type to "CD-ROM" if necessary.

6. Click **Apply**.
7. From the list of types of packages, double-click "PROC-SUPPORT."

   A list of the processor support packages on the CD-ROM will be displayed.

8. Click on the "MPC860" package.

   If you are unsure whether this is the correct package, click **Details** for information about the contents of the package.

9. Click **Install**.

   The Continue dialog box will appear.

10. Select **Continue**.

    The Software Install dialog will display "Progress: completed successfully" when the installation is complete.

11 If required, the system will automatically reboot. Otherwise, close the software installation windows.

The configuration files are stored in /hplogic/configs/hp/*mpc8xx*.

The inverse assemblers are stored in /hplogic/ia.

See Also

The instructions printed on the CD-ROM package for a summary of the installation instructions.

The online help for more information on installing, licensing, and removing software.

To load a configuration from the floppy disk (HP 16600A/700A)

The preferred method is to install this functionality from the CD-ROM onto the hard disk and load from the hard disk.

To install a configuration and inverse assembler file from the floppy disk that was shipped with your HP analysis probe:

1. Install the floppy disk in the floppy drive on the HP 16600A/16700A-series logic analysis system mainframe.
2. In the Logic Analysis System window, click the **File Manager** icon.
3. In the File Manager window:
   - Set **Current Disk** to Flexible Disk.
   - Set Target to the analyzer you wish to configure.
   - Click the name of the desired configuration file in the Contents frame. The Contents frame lists the configuration files and inverse assembler files available on the floppy disk. These may be either DOS or LIF format files. Either format can be loaded directly into the appropriate logic analyzers.

   Note that the logic analyzers read both DOS and LIF formats. However, only DOS formatted floppy disks can be used to store configurations and data. LIF format floppy disks are read-only.
4. Click **Load**.

   The configuration file you choose will set up the logic analyzer and associated tools. You may see Information, Error, and Warning dialogs that say your configuration has been loaded, and advise you about making proper connections.
5. Click the **Workspace window** icon to see the arrangement of analysis tools in your configuration.
6. Right-click the logic analyzer icon in your configuration and choose its **Setup** button to see the way your configuration file defined the Config, Format, and Trigger options.

   Under the Format tab, buses are labeled, and bits included in each bus are identified by an asterisk "*".

This procedure restores the configuration that was in effect when the configuration file was saved. Because the file was not saved using your system, you may receive error messages about loading the enhanced inverse assembler or about pods that were truncated. Click the Config, Format, and Trigger tabs and modify the configuration to satisfy your measurement desires. Then you can save your customized configuration to DOS format using the **File→**Save Configuration selection in any of your tool windows, or clicking the Save tab in the File Manager. For details about how to save configuration files, open the Help window.

To list software packages that are installed (HP 16600A/700A)

- In the System Administration Tools window, click **List...**.

To install software on other logic analyzers

Consult the documentation for your logic analyzer for details.

3

Connecting and Configuring the Analysis Probe

This chapter shows you how to connect the logic analyzer to the target system through the analysis probe. If you are using custom probing, turn to page 85.

If you are connecting to an HP 16600A-series or HP 16700A series logic analyzer, use the Setup Assistant to connect and configure your system (see page 23). Use this manual for additional information, if desired.

If you are not using the Setup Assistant, follow the instructions given in this chapter. This chapter covers the following tasks; the order shown here is the recommended order for performing these tasks:

- Check that the target system meets the necessary requirements
- Read the power on/power off sequence
- Connect the analysis probe to the target system
- Connect the analysis probe to the logic analyzer
- Configure the logic analyzer

Target System Requirements

You must solder the provided BGA socket onto the target system in place of the processor, as described later in this chapter.

Keep-out area on the target board

The analysis probe requires a 29.30 mm by 29.30 mm keep-out area where it overhangs the BGA socket. The maximum height of components under the analysis probe in this area cannot exceed 19 mm.

If components are too high for the clearance, order the HP E2476-87602 extender to add an additional 6.76 mm of clearance. Do not exceed two extenders.

Clearance above the target board

See the diagram on the next page for the dimensions of the analysis probe.

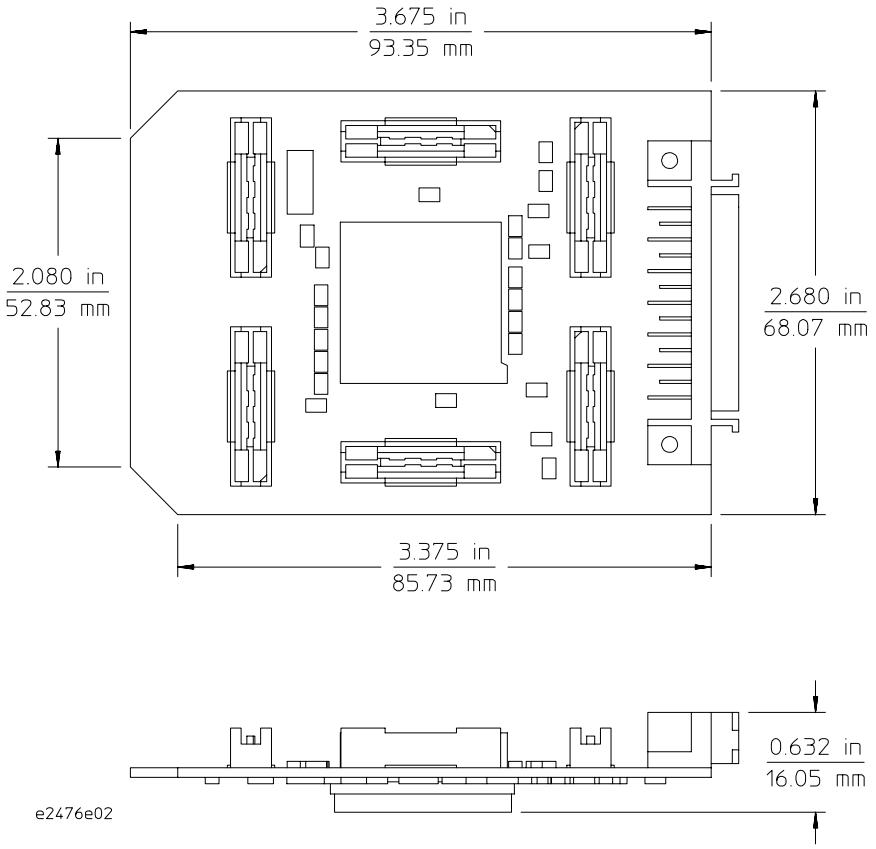
Note that the BGA probing kit will add to the thickness of the analysis probe, for a total of 86 mm. You must also allow space for the cables which plug into the top of the analysis probe.

See Also

The *Solutions for the Motorola MPC 800 Embedded PowerPC Microprocessor Family* data sheet, available from your HP representative, has more detailed information and diagrams regarding the keep-out area and analysis probe dimensions.

Analysis probe — circuit board dimensions

The following figure gives the dimensions for the analysis probe circuit board. The dimensions are listed in inches and millimeters.



HP E2476A Analysis Probe Circuit Board Dimension Diagram

Power-on/Power-off Sequence

Listed below are the sequences for powering on and off a fully connected system. Simply stated, your target system is always the last to be powered on, and the first to be powered off.

To power on HP 16600A and HP 16700A-series logic analysis systems

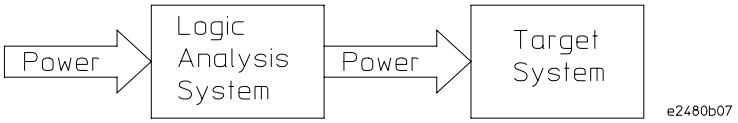
Ensure the target system is powered off.

1. Turn on the logic analyzer. The Setup Assistant will guide you through the process of connecting and configuring the analysis probe.
2. When the analysis probe is connected to the target system and logic analyzer, and everything is configured, turn on your target system.

To power on all other logic analyzers

With all components connected, power on your system in the following order:

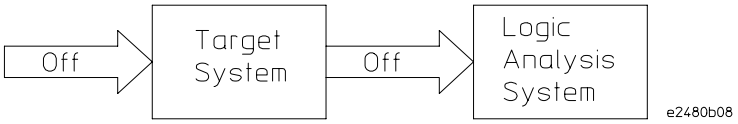
1. Logic analysis system.
2. Your target system.

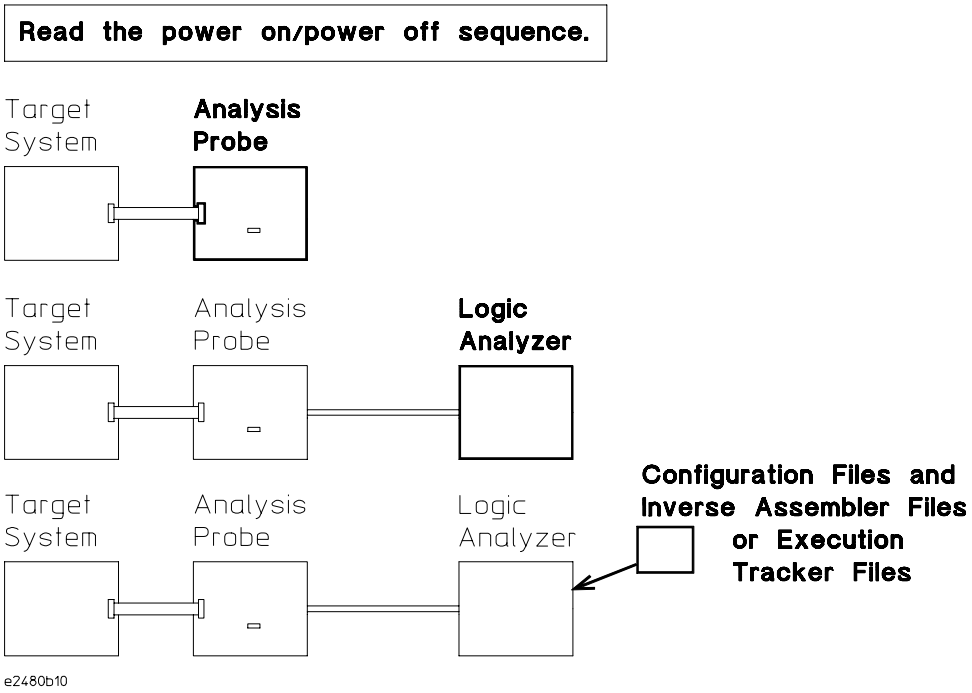


To power off

Turn off power to your system in the following order:

1. Turn off your target system.
2. Turn off your logic analysis system.





Connection Sequence

Connecting the Analysis Probe to the Target System

This section explains how to connect the HP E2476A analysis probe to the target system. Connecting the analysis probe to the target system consists of the following steps, which are described on the following pages:

- Turn off the target system.
- Turn off the logic analyzer (unless you are using an HP 16600/16700A logic analysis system).
- Assemble the microprocessor into the BGA carrier.
- Install the HP E5355A BGA probing kit on the target system.
- Test the target system with the BGA carrier assembly, without the analysis probe, then turn off the power again.
- Disconnect the BGA carrier assembly from the target system.
- Install the analysis probe onto the target system, and then install the BGA carrier assembly onto the analysis probe.

The remainder of this section describes these general steps in more detail.

Protect Your Equipment

The analysis probe socket assembly pins are covered for shipment with a conductive foam wafer or conductive plastic pin protector. This is done to protect the delicate gold-plated pins from damage due to impact. When you're not using the analysis probe, protect the socket assembly pins from damage by covering them with the pin protector.

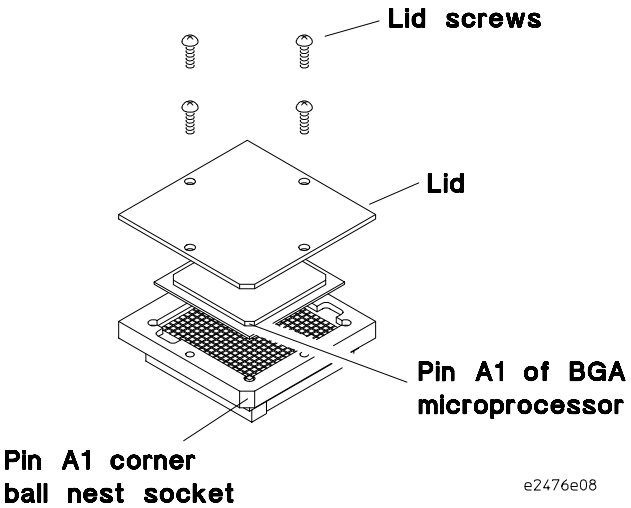
To assemble the microprocessor into the BGA carrier

The HP E2476A analysis probe has a BGA carrier for a 357-pin BGA microprocessor. Use the procedure below to install the BGA microprocessor into the BGA carrier.

1 Align pin A1 on the BGA microprocessor with the pin A1 corner of the BGA carrier (see below).

CAUTION: Serious damage to the target system or analysis probe can result from incorrect connection. Note the position of pin A1 on the BGA carrier and BGA microprocessor prior to making any connection.

2 Place the BGA microprocessor into the BGA carrier, and tighten the four lid screws.



Pin A1 Orientation of BGA Microprocessor and BGA Carrier

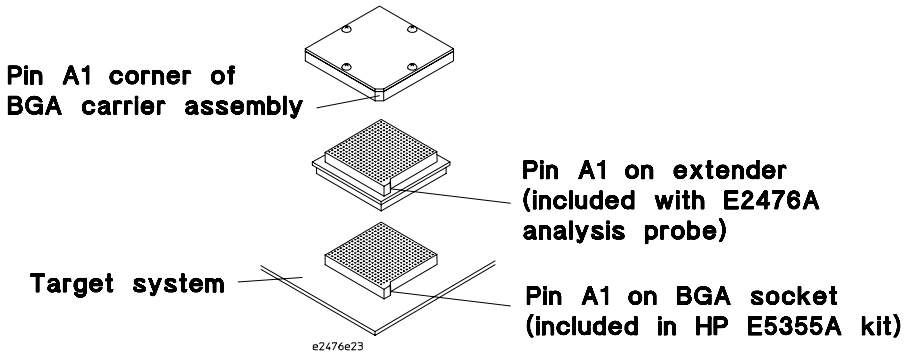
CAUTION: Multiple insertions of the BGA microprocessor into the BGA carrier may degrade the ball nest socket connections. Once the BGA microprocessor is inserted in the ball nest socket, tighten the four lid screws forcefully. Only remove the BGA microprocessor from the BGA carrier when necessary for silicon upgrades.

To install the HP E5355A BGA probing kit on the target system

The HP E5355A BGA probing kit requires a target system with an empty 357-pin BGA pad array. Connect the BGA probing kit using the following instructions.

1. Ensure that your target system has a 357-pin BGA pad array with proper connections for your target microprocessor. This BGA pad array must be clean, unused, and have no solder on its pads.
2. Ensure that pin A1 of the BGA socket is properly aligned with pin A1 on the BGA pad array.
3. Install the socket onto the 357-pin BGA pad array, and solder it in place. Follow the soldering instructions in the process sheet that came with the HP E5355A BGA probing kit.
4. Install the extender into the socket. The extender protects the socket and your target board.

CAUTION: **Target System Damage.** Once the extender is installed, do not remove it from the socket. The socket is held in place on your target system by solder between the socket pins and the BGA pads. If you remove the socket, one or more of the soldered pads may damage connections, traces, and BGA pads of your target system board assembly.



Probing Installation

To test the target system with the BGA carrier assembly

Before installing the analysis probe onto the target system, ensure that the socket and extender have been installed successfully with the following steps.

1. Install the BGA carrier assembly into the extender.
2. Turn on your target system and check operation.

   The BGA socket, extender, and BGA carrier assembly add inductance and capacitance. Ensure that your target system operates properly before installing the analysis probe board assembly.

   Open connections or shorts may exist after soldering the BGA socket to the target board. If a previously functioning target board does not function after installing the socket, check continuity of the socket pins. Touch a dry-tip soldering iron to any open pin.

To disconnect the BGA carrier assembly or the analysis probe from the extender

You must remove the BGA carrier assembly to attach the analysis probe. Use this procedure disconnecting the BGA carrier assembly or the analysis probe from the extender.

The extractor tool comes with an Operating Guide showing how to use the extractor tool to disconnect the BGA carrier assembly or the analysis probe from the extender.

1. Refer to that Operating Guide and use the extractor tool to lift the BGA carrier assembly or the analysis probe from the extender. Keep all connector pins straight during removal.
2. Do not remove the extender from the BGA socket on the target board.
3. Do not plug anything other than the extender into the BGA socket on the target board.

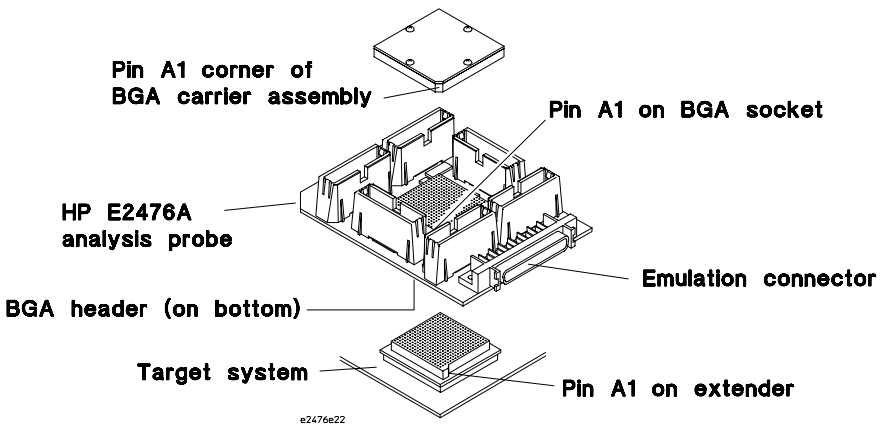
To install the analysis probe between the BGA carrier assembly and the target system

The analysis probe BGA header is on the bottom of analysis probe. It connects to the extender on the target system.

1 Install the analysis probe BGA header into the extender on the target system. Ensure that pin A1 is properly aligned (see figure below).

CAUTION: **Target System Damage.** Serious damage to the target system or analysis probe can result from incorrect connection. Note the position of pin A1 on the target system, analysis probe BGA header, and BGA carrier assembly prior to making any connection.

If the analysis probe interferes with components of the target system, or if a higher profile is required, additional BGA extenders can be used. BGA extenders can be ordered from Hewlett-Packard using the HP part numbers listed in the Replaceable Parts table on page 293.



Connecting the Analysis Probe to the Target System

Connecting the Analysis Probe to the Logic Analyzer

This section shows the connections between the logic analyzer pod cables and the high-density cables on the analysis probe. Use the appropriate page for your logic analyzer. The configuration file names for each logic analyzer are included with the connection diagrams.

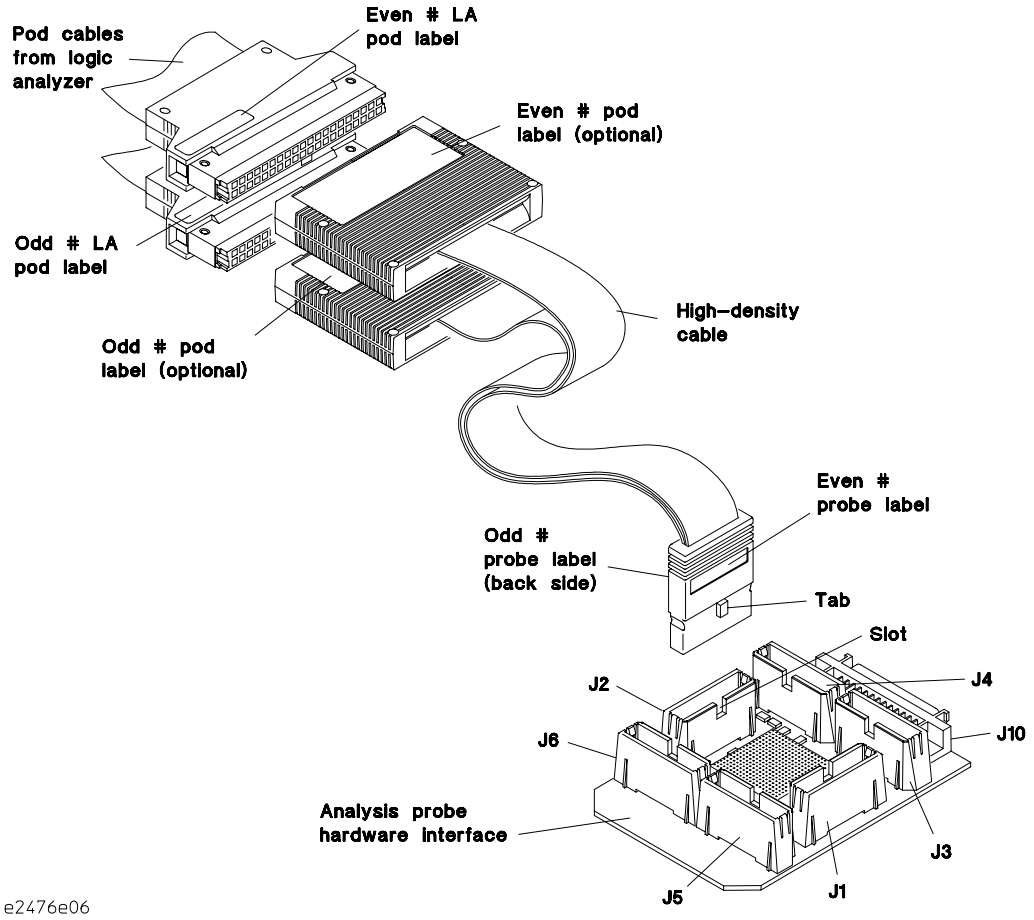
The figure on the following page shows the connectors on the analysis probe. Note that only J1, J2, and J3 are required for inverse assembly. J4, J5, and J6 contain additional signals you might want to monitor; however, they must be connected according to the diagrams on the following pages.

This section shows diagrams for connecting the analysis probe to the logic analyzers listed below:

- HP 16710/11/12A logic analyzers (1 or 2 cards) - see page 57 - 58
- HP 16715/16/17A logic analyzers (2 or 3 cards) - see page 60 - 61
- HP 16600A logic analyzer - see page 63
- HP 16601A logic analyzer - see page 65
- HP 16602A logic analyzer - see page 66
- HP 16550A logic analyzers (1 or 2 cards) - see page 57, 67
- HP 16554/55/56/57 logic analyzer (2 or 3 cards) - see page 69 - 70
- HP 1660A/AS/C/CS/CP/E/ES/EP logic analyzer - see page 72
- HP 1661A/AS/C/CS/CP/E/ES/EP logic analyzer - see page 73
- HP 1670A/D/E logic analyzer - see page 74
- HP 1671A/D/E logic analyzer - see page 75

To connect the high-density termination cables to the analysis probe

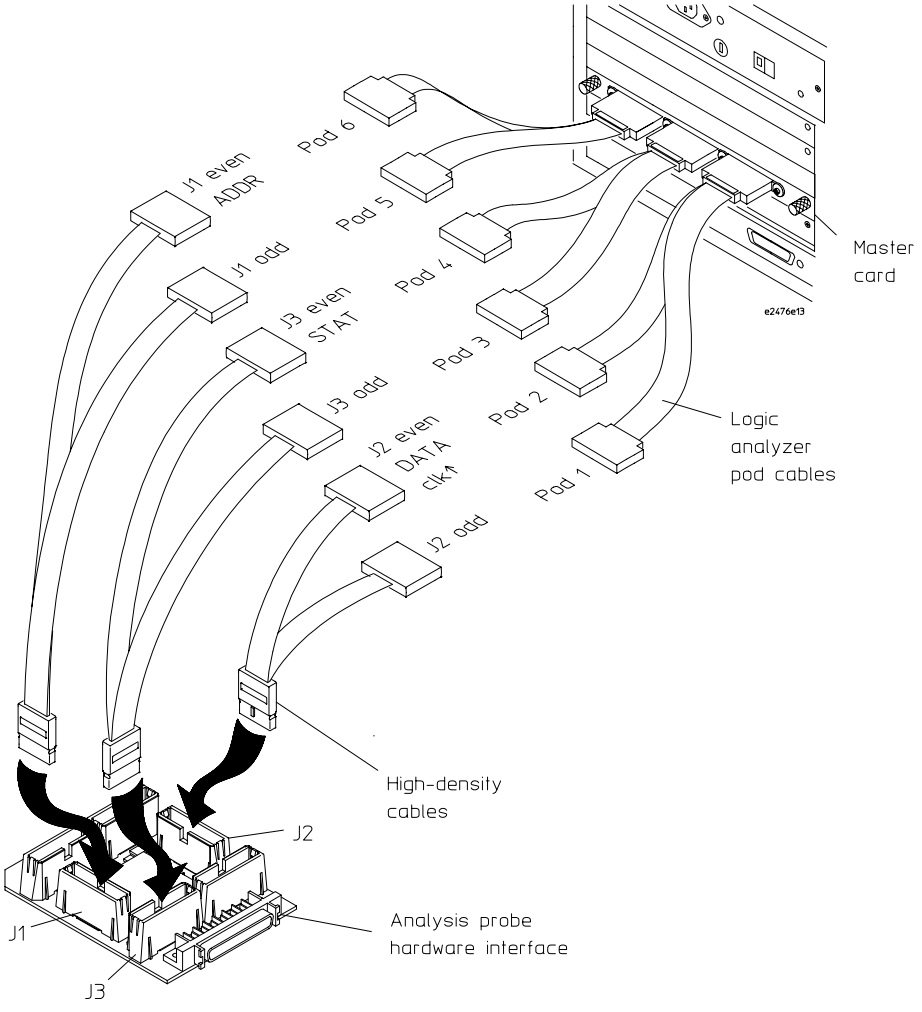
Three HP E5346A high-density termination cables, and labels to identify them, are included with the HP E2476A. Connect the cables to the connectors on the analysis probe as shown in the illustration below. Attach the labels to the cables after connecting the cables to the logic analyzer. Note that only J1, J2, and J3 are required for inverse assembly. J4, J5, and J6 contain additional signals you might want to monitor.



Connecting High-Density Cables to the Analysis Probe

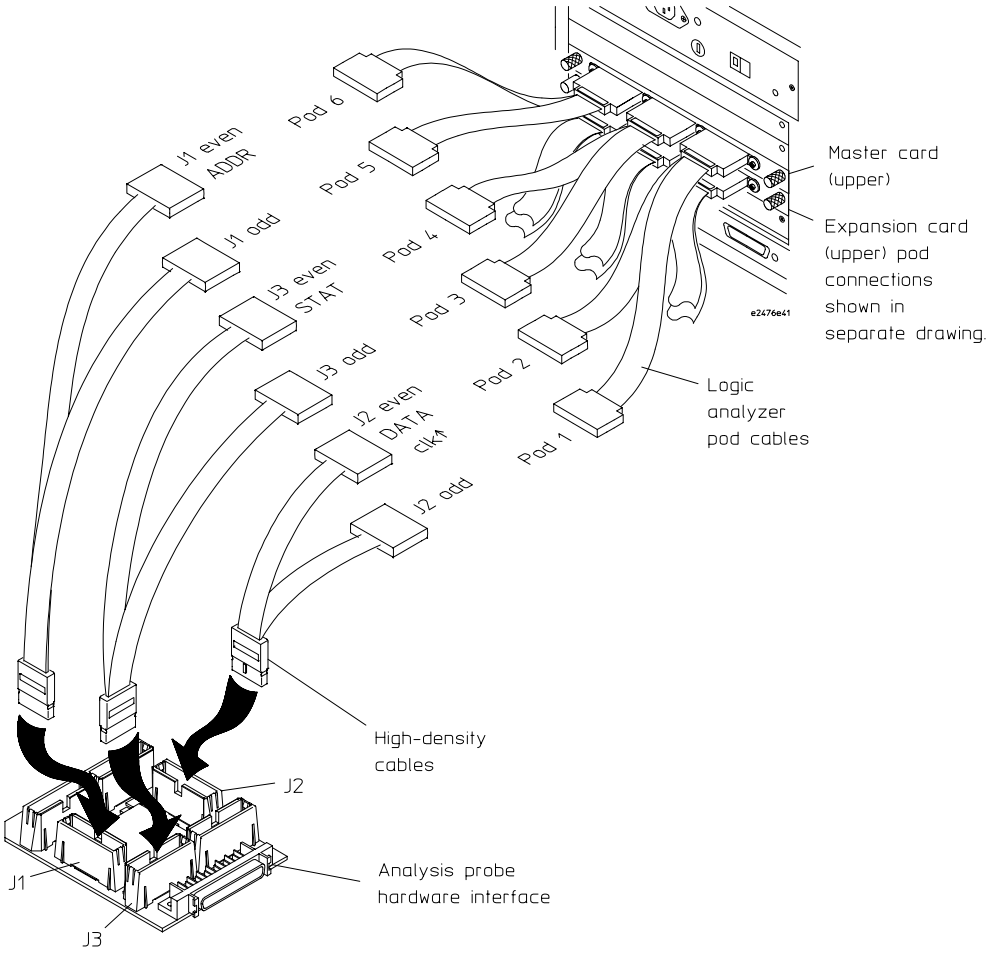
To connect to the HP 16710/11/12A or HP 16550A analyzer (one-card)

Use the figure below to connect the analysis probe to the one-card HP 16710/11/12A or HP 16550A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

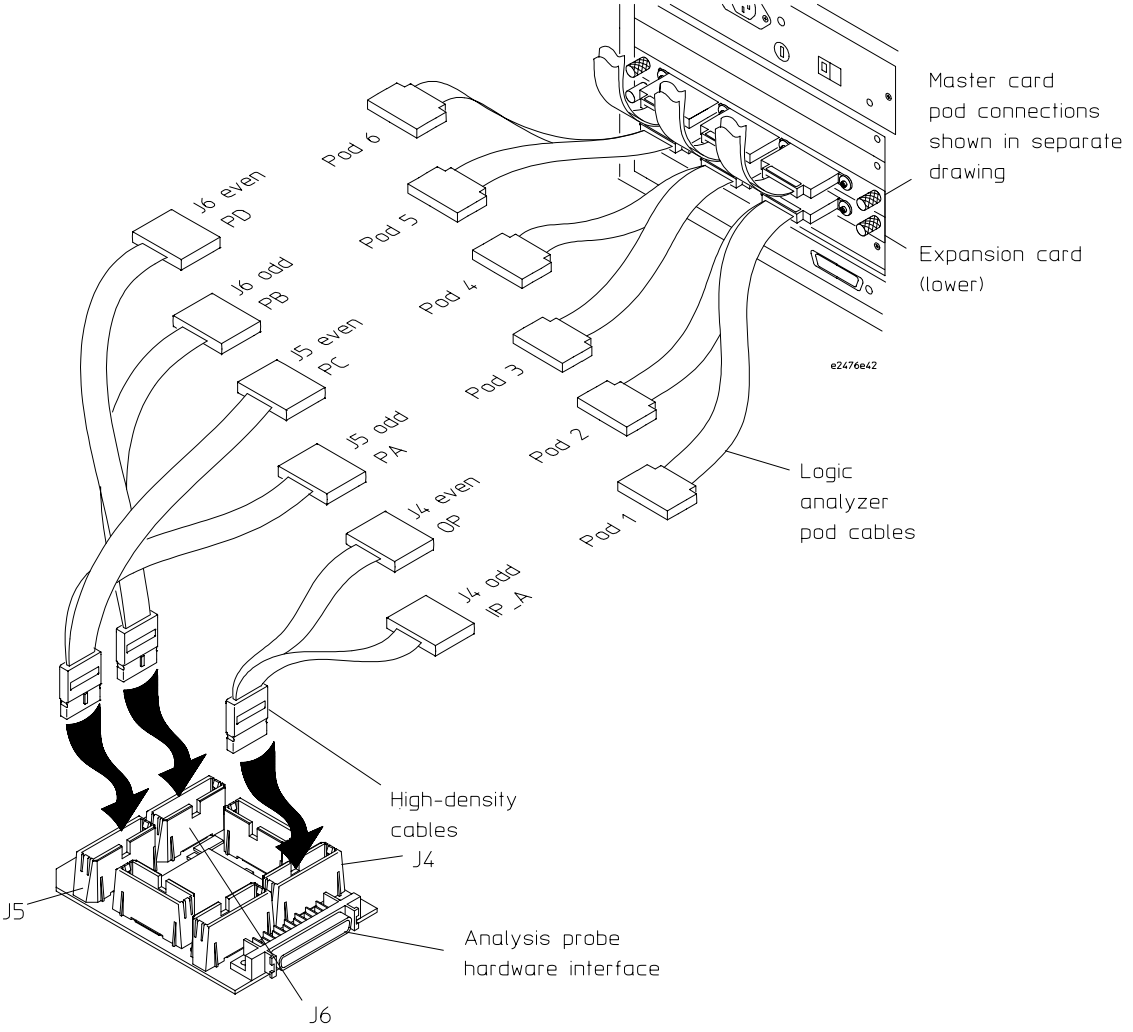


To connect to the HP 16710/11/12A analyzer (two-card)

Use the figure below (continued on next page) to connect the analysis probe to the two-card HP 16710/11/12A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



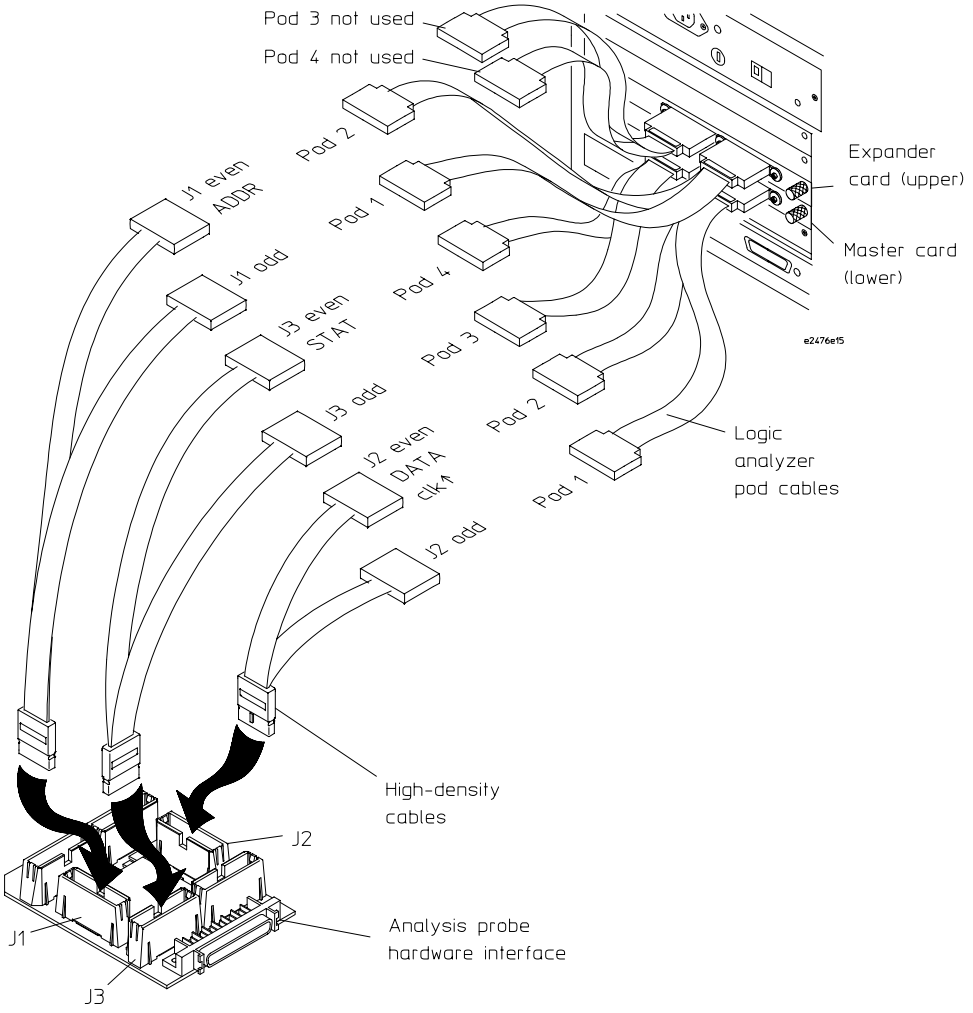
HP 16710/11/12A analyzer two-card connections, part 1



HP 16710/11/12A analyzer two-card connections, part 2

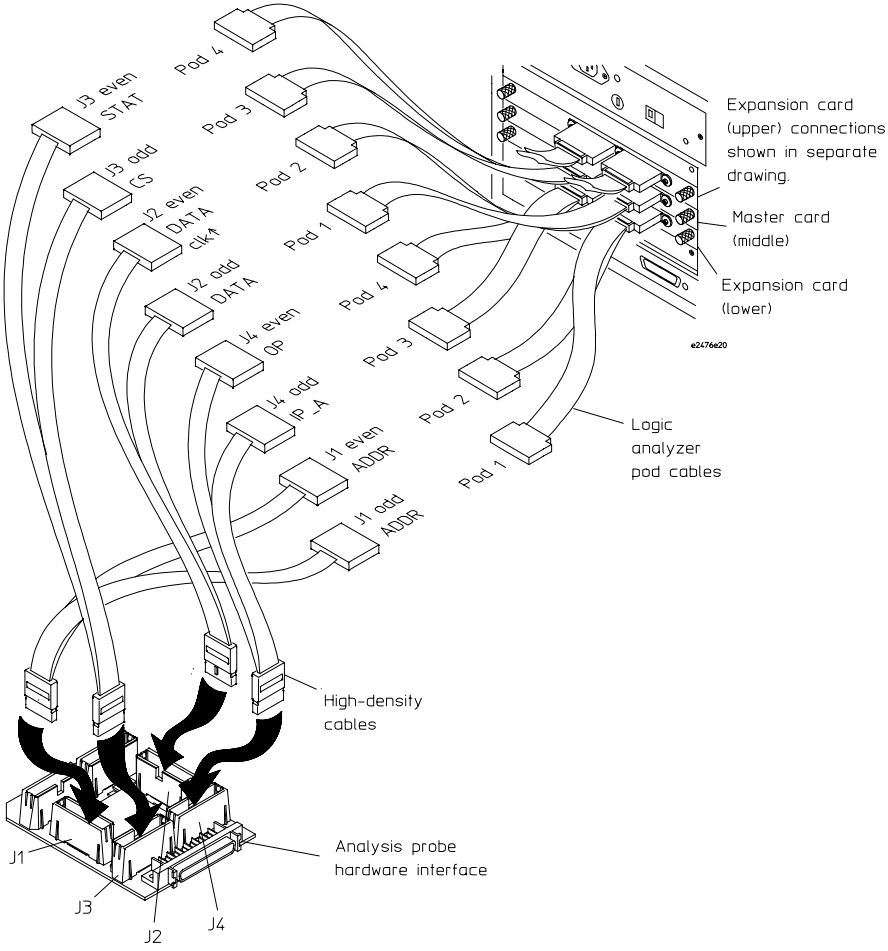
To connect to the HP 16715/16/17A analyzer (two-card)

Use the figure below (continued on next page) to connect the analysis probe to the two-card HP 16715/16/17A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

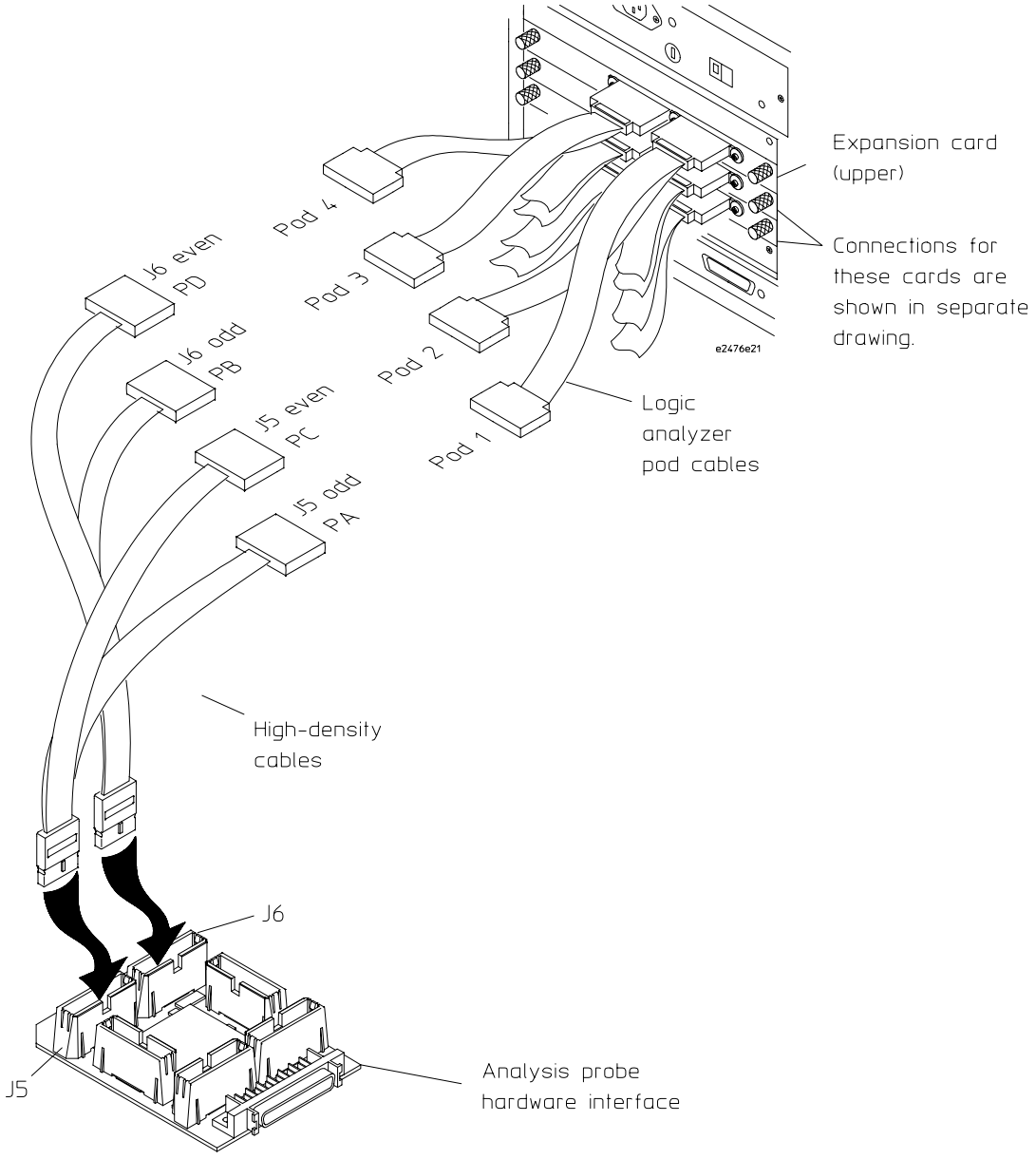


To connect to the HP 16715/16/17A analyzer (three-card)

Use the figure below (continued on next page) to connect the analysis probe to the three-card HP 16715/16/17A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



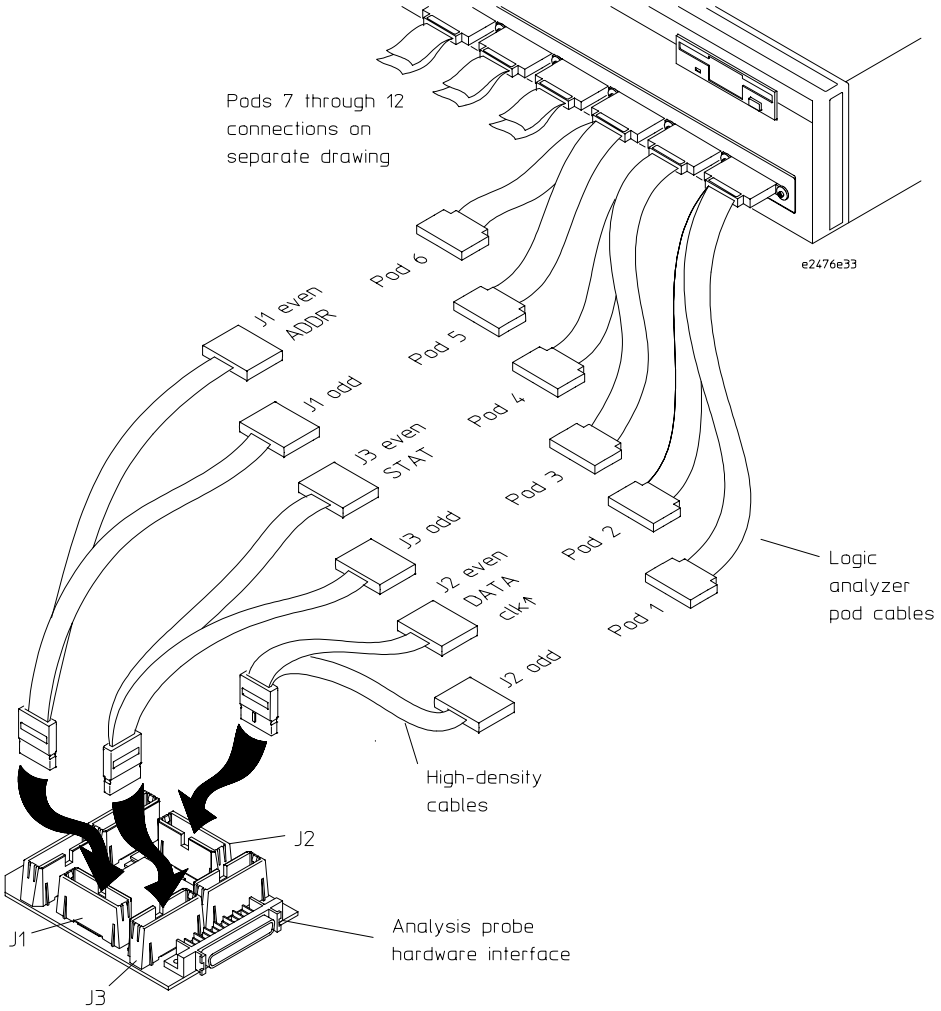
HP 16715/16/17A analyzer three-card connections, part 1



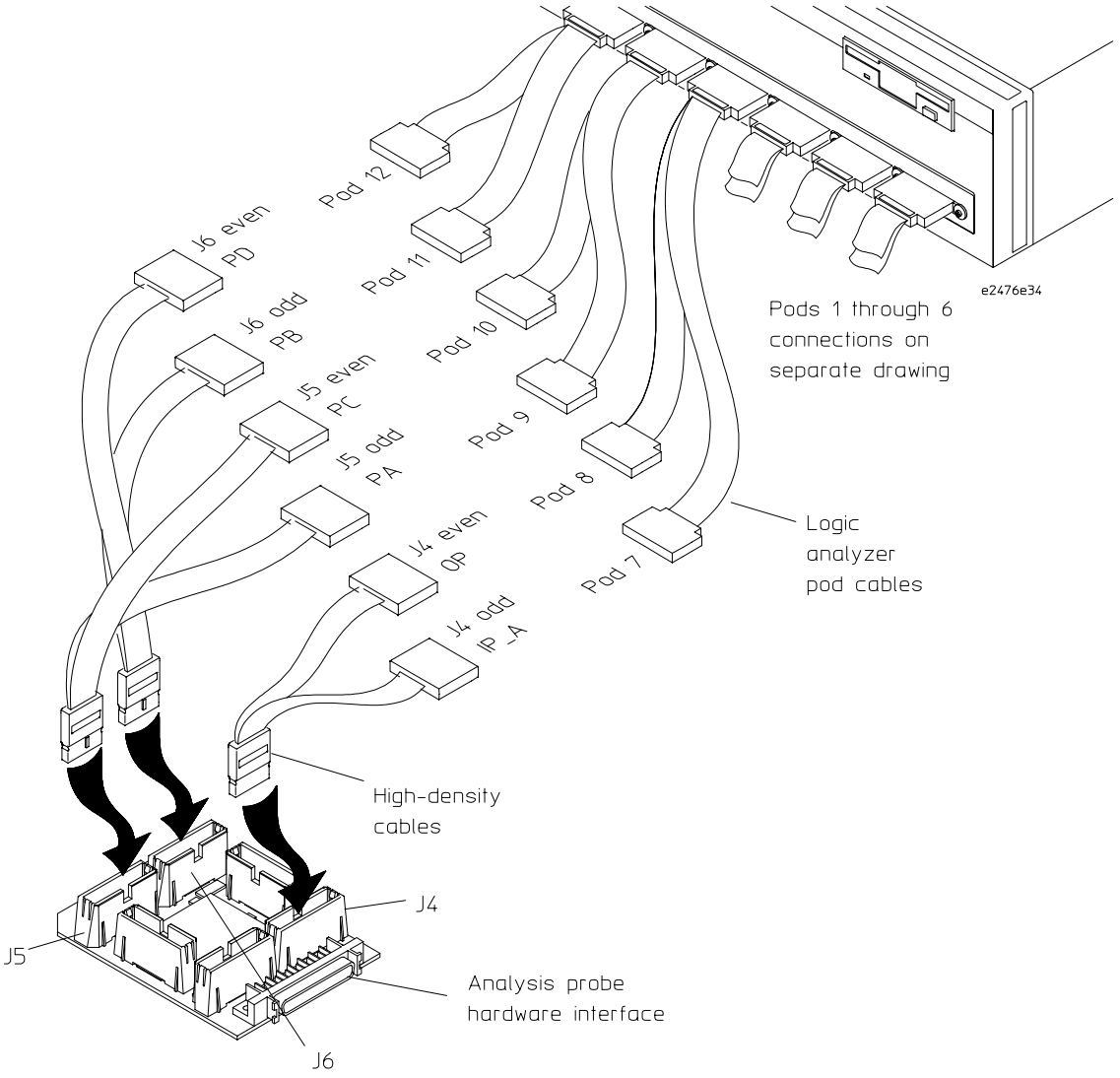
HP 16715/16/17A analyzer three-card connections, part 2

To connect to the HP 16600A logic analyzer

Use the figure below to connect the analysis probe to the HP 16600A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



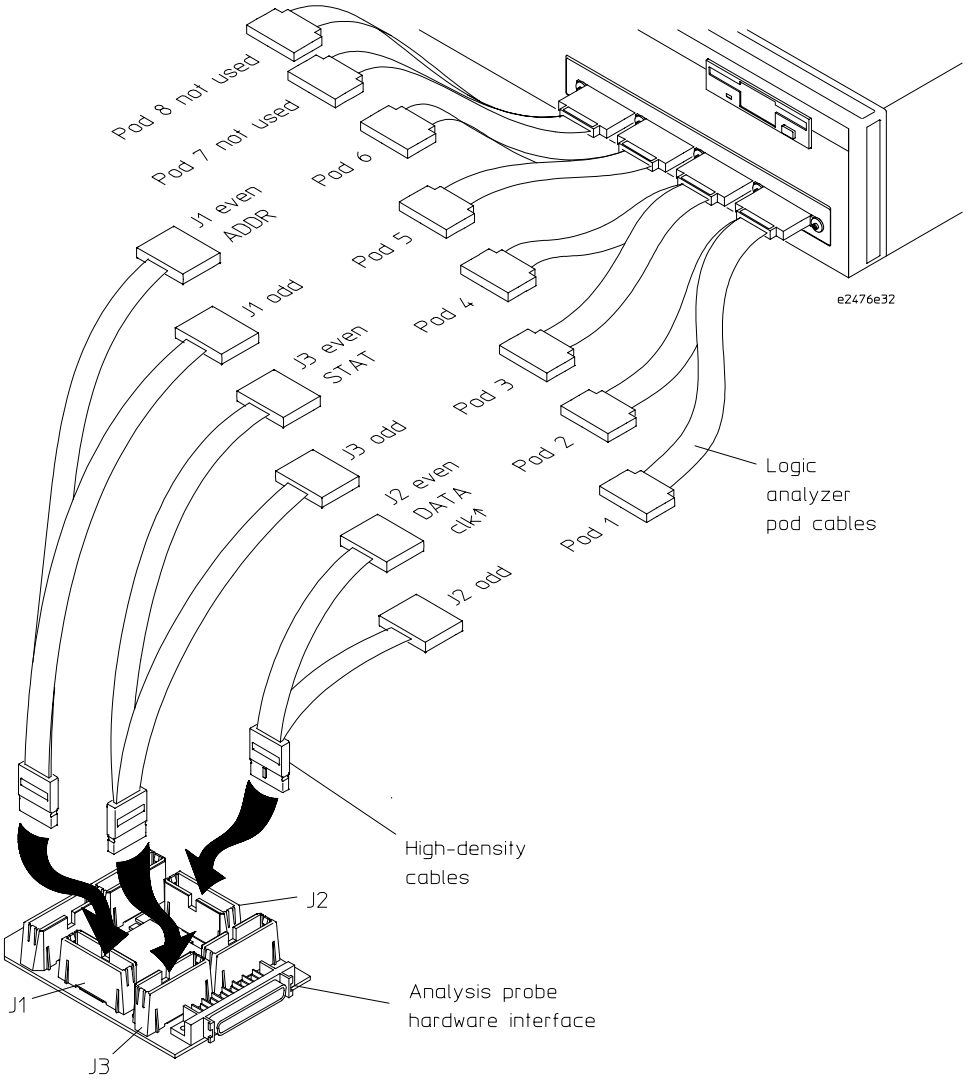
HP 16600A analyzer connections, part 1



HP 16600A analyzer connections, part 2

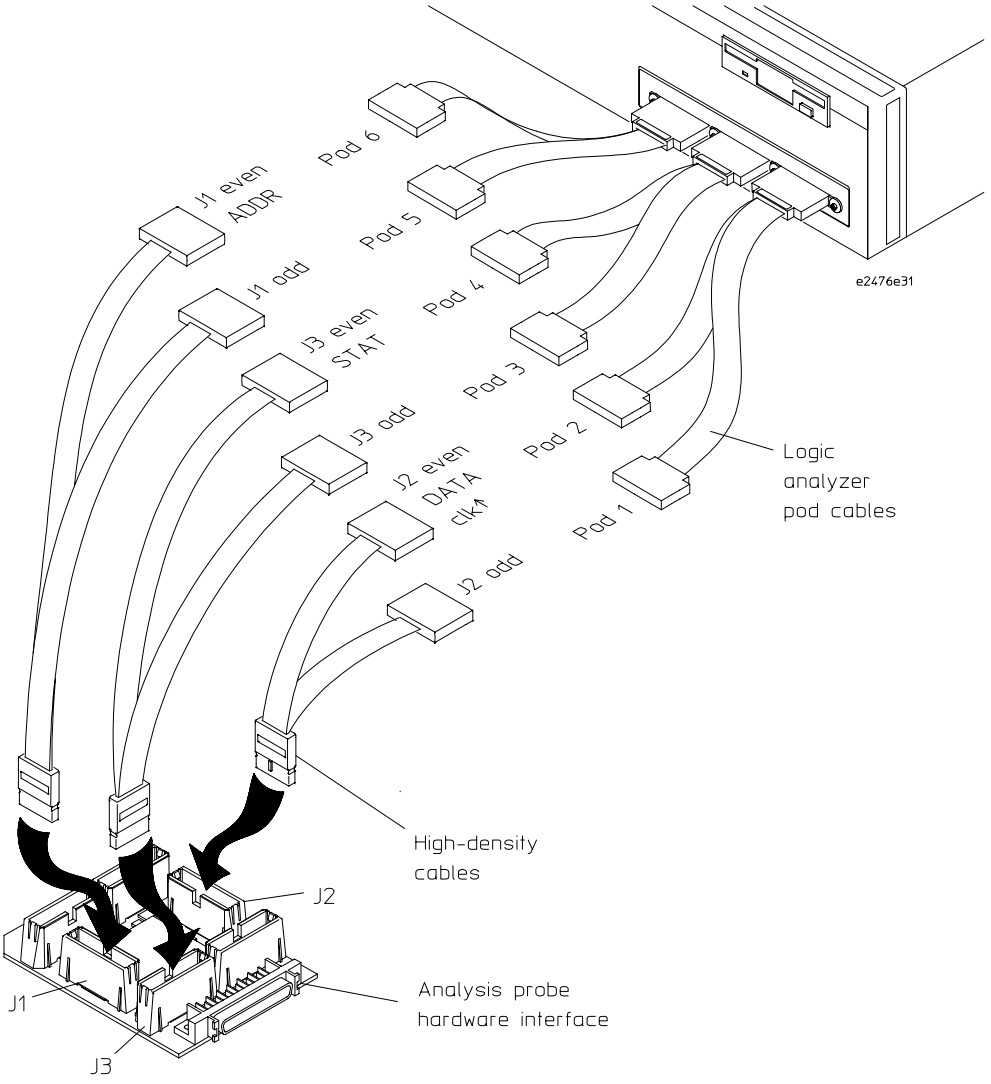
To connect to the HP 16601A logic analyzer

Use the figure below to connect the analysis probe to the HP 16601A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



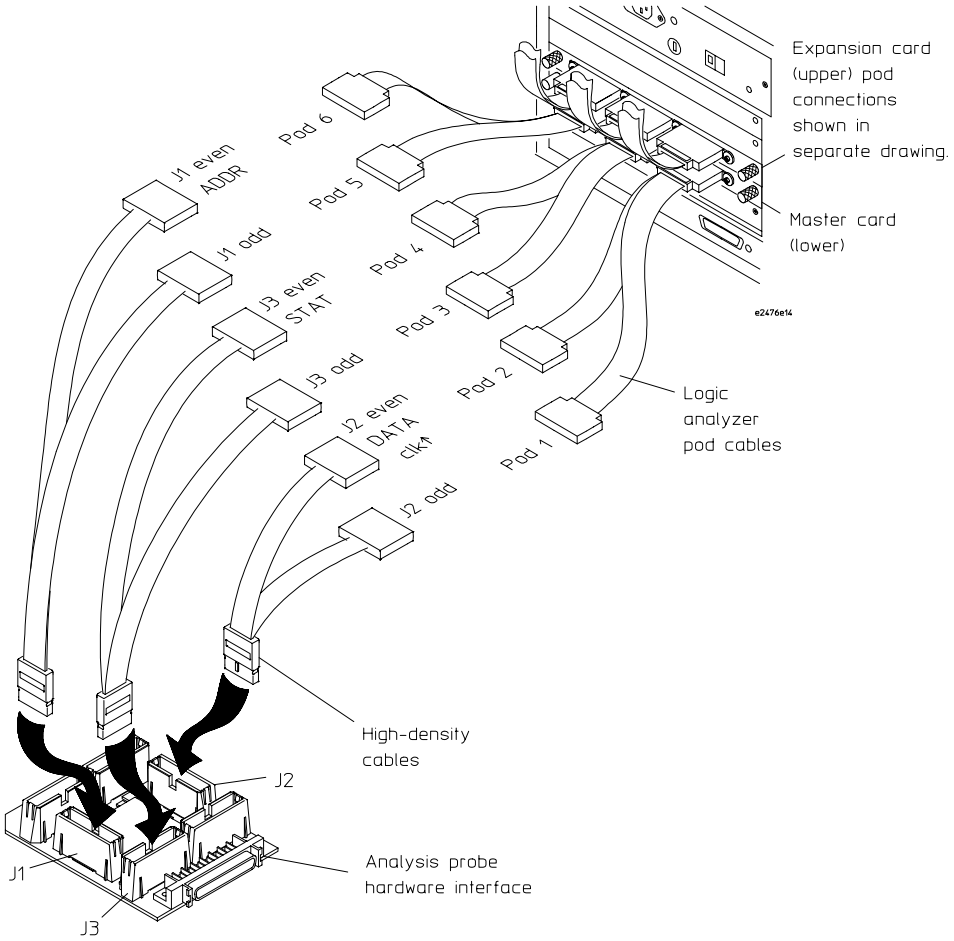
To connect to the HP 16602A logic analyzer

Use the figure below to connect the analysis probe to the HP 16602A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

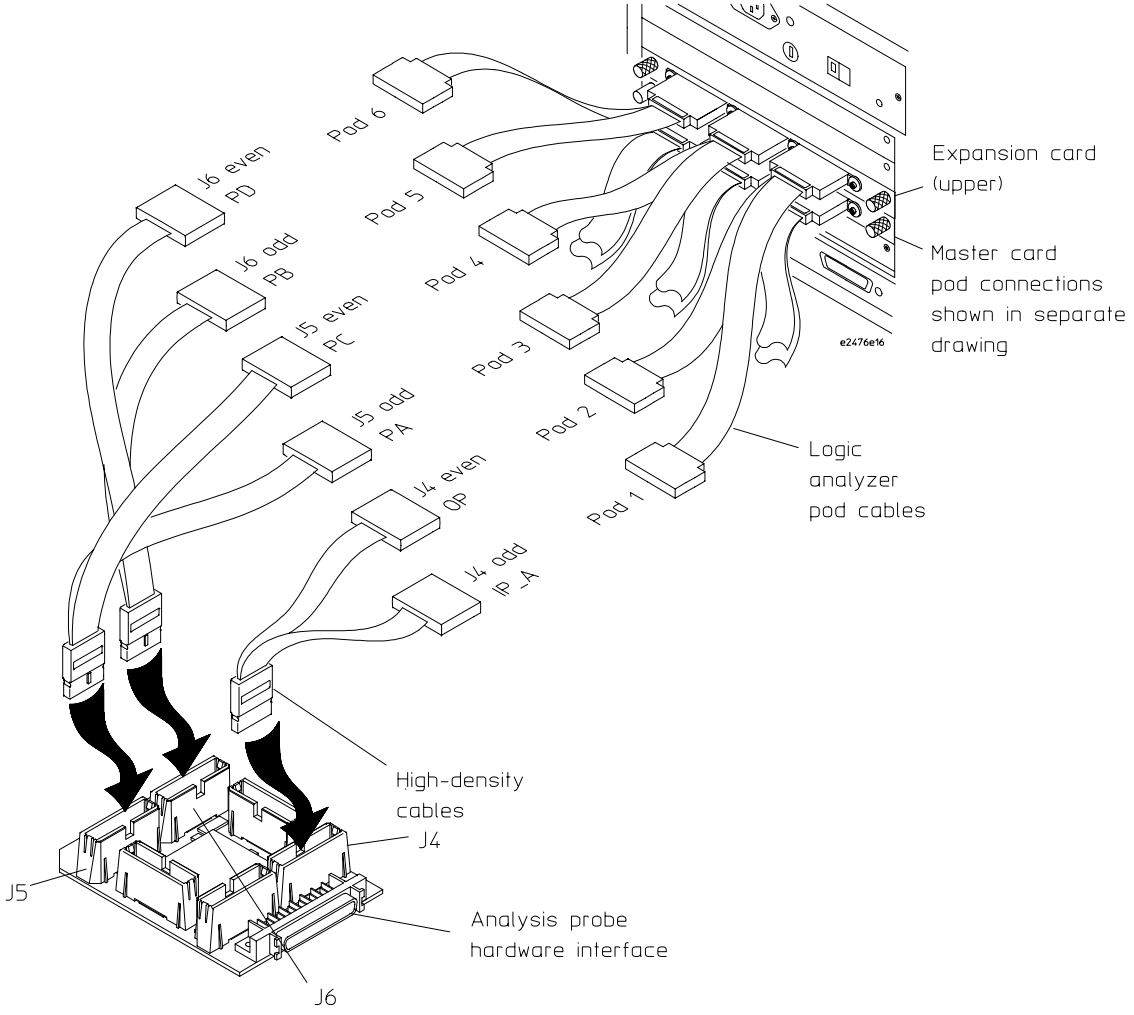


To connect to the HP 16550A analyzer (two-card)

Use the figure below (continued on next page) to connect the analysis probe to the two-card HP 16550A logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



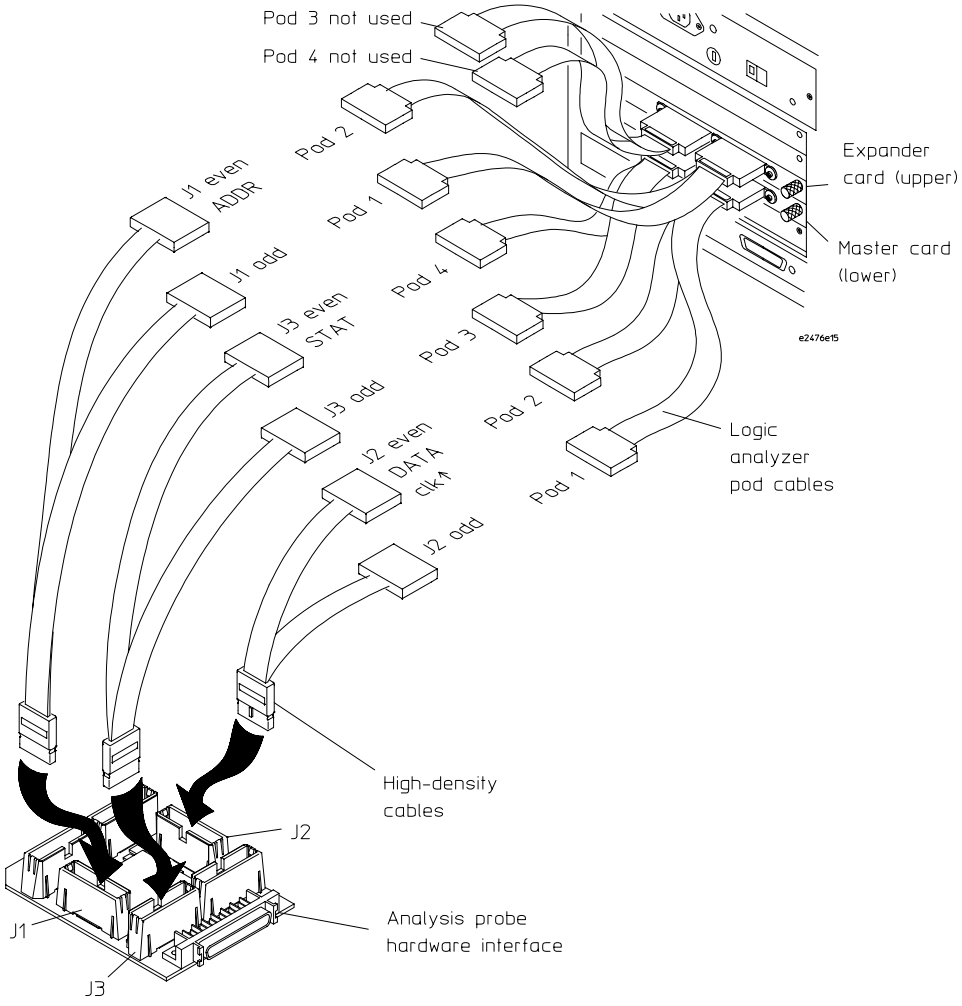
HP 16550A analyzer two-card connections, part 1



HP 16550A analyzer two-card connections, part 2

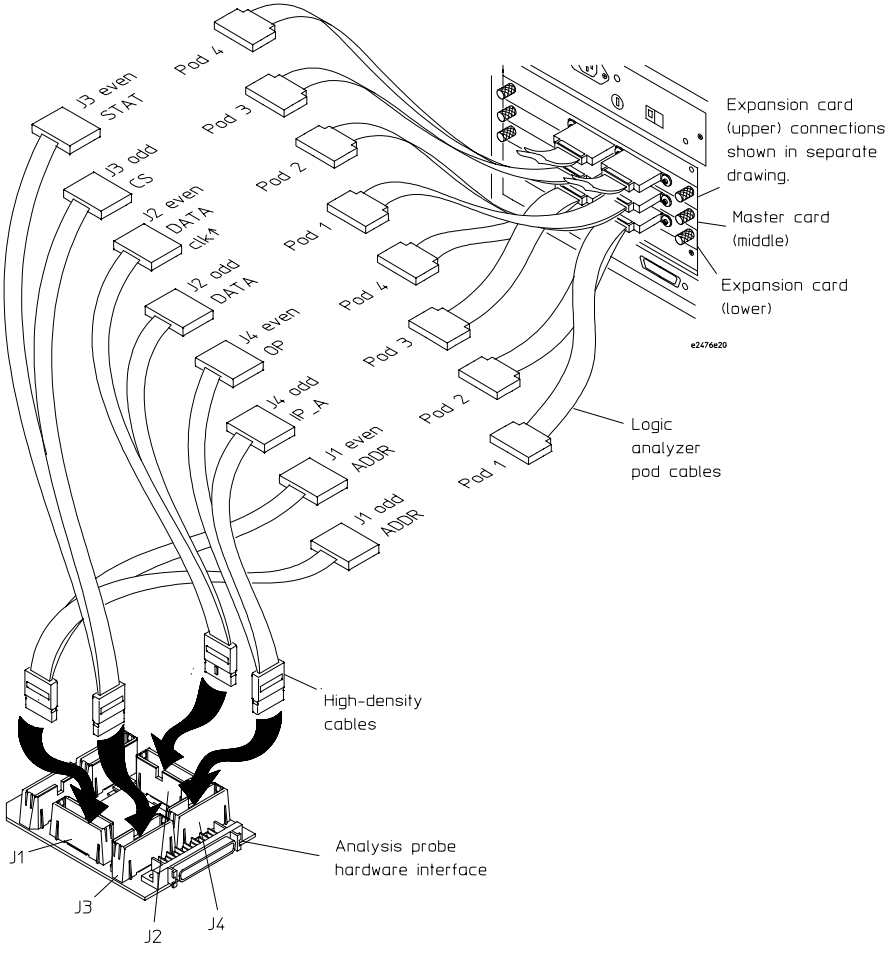
To connect to the HP 16554/55/56/57 analyzer (two-card)

Use the figure below to connect the analysis probe to the two-card HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

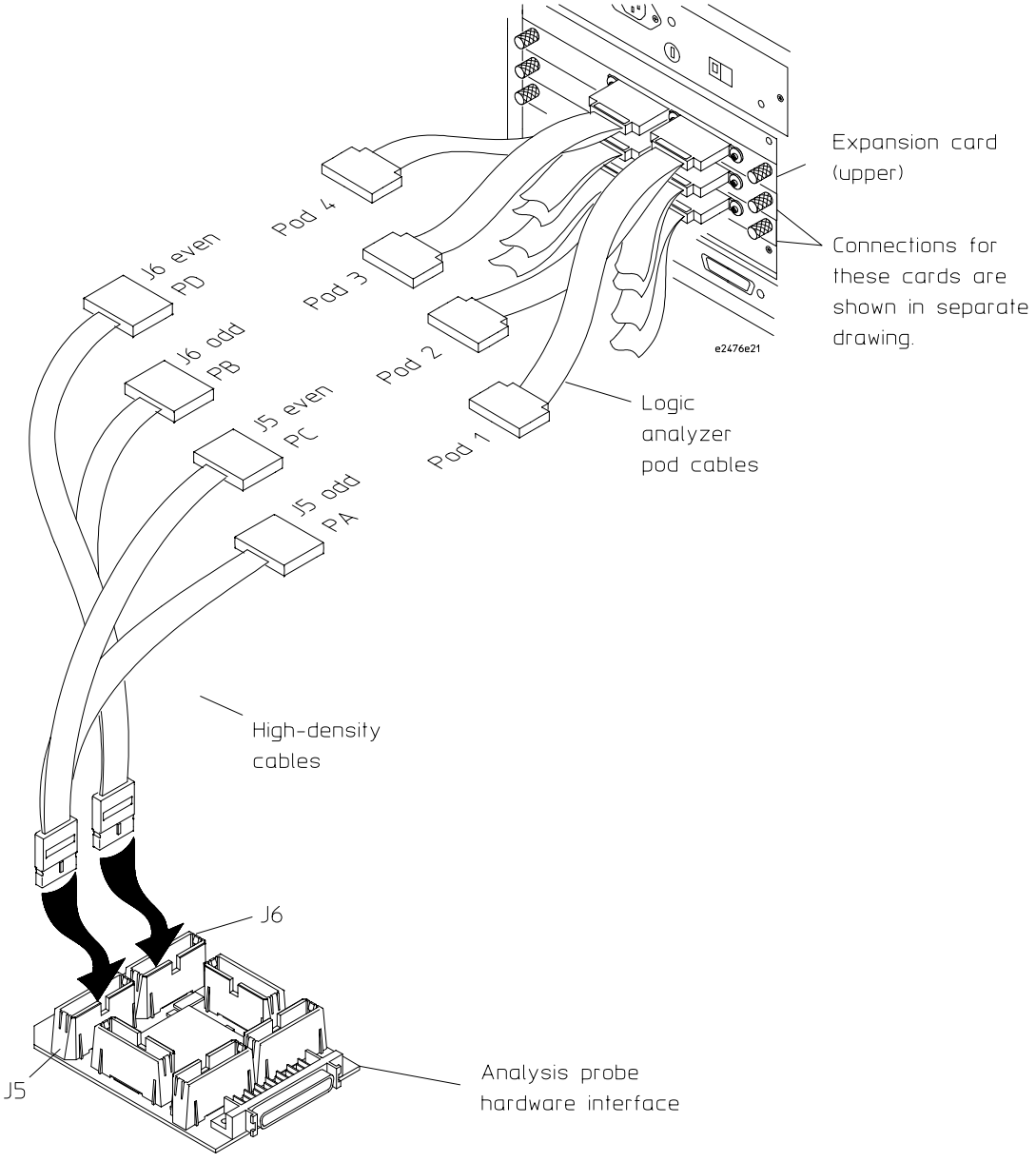


To connect to the HP 16554/55/56/57 analyzer (three-card)

Use the figure below to connect the analysis probe to the three-card HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



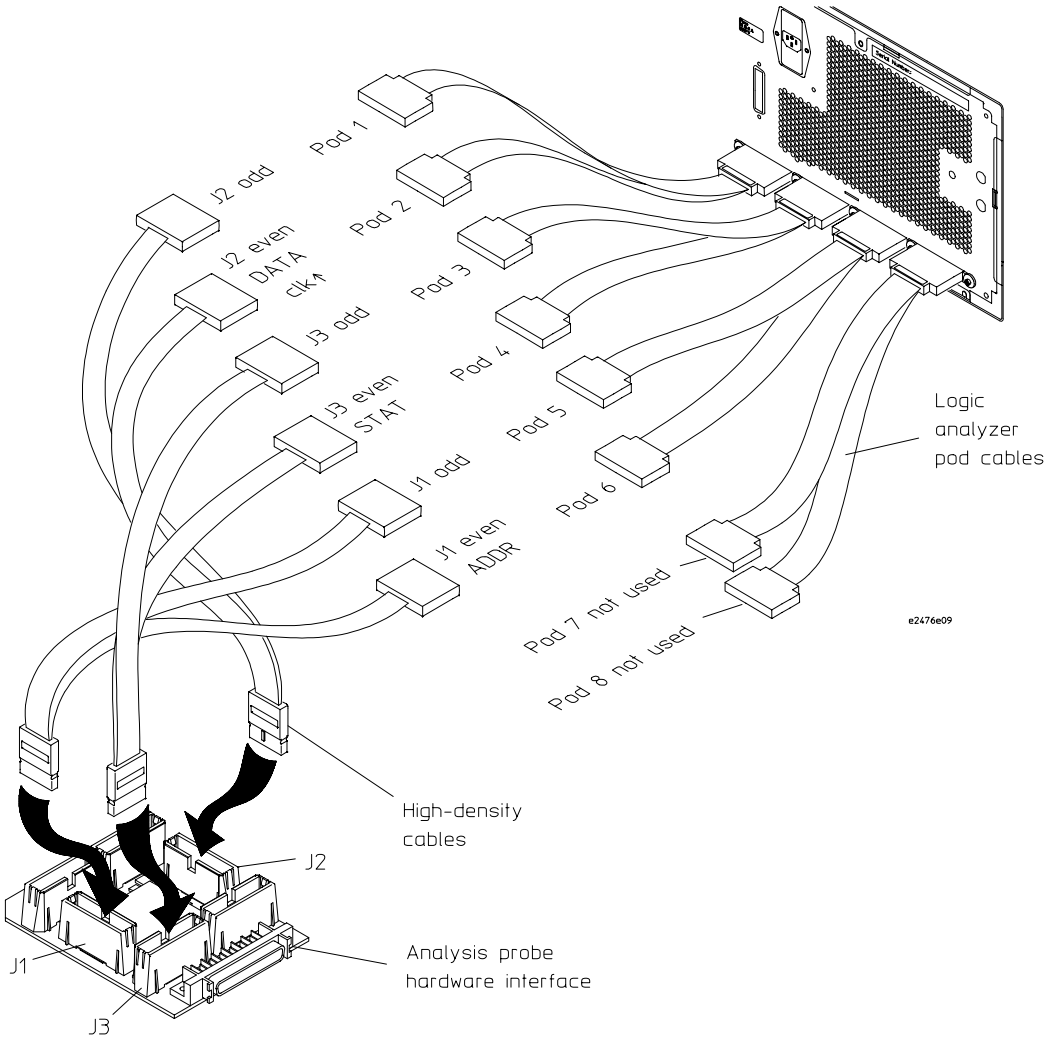
HP 16554/55/56/57 analyzer three-card analyzer connections, part 1



Three-card HP 16554/55/56/57 analyzer three-card connections, part 2

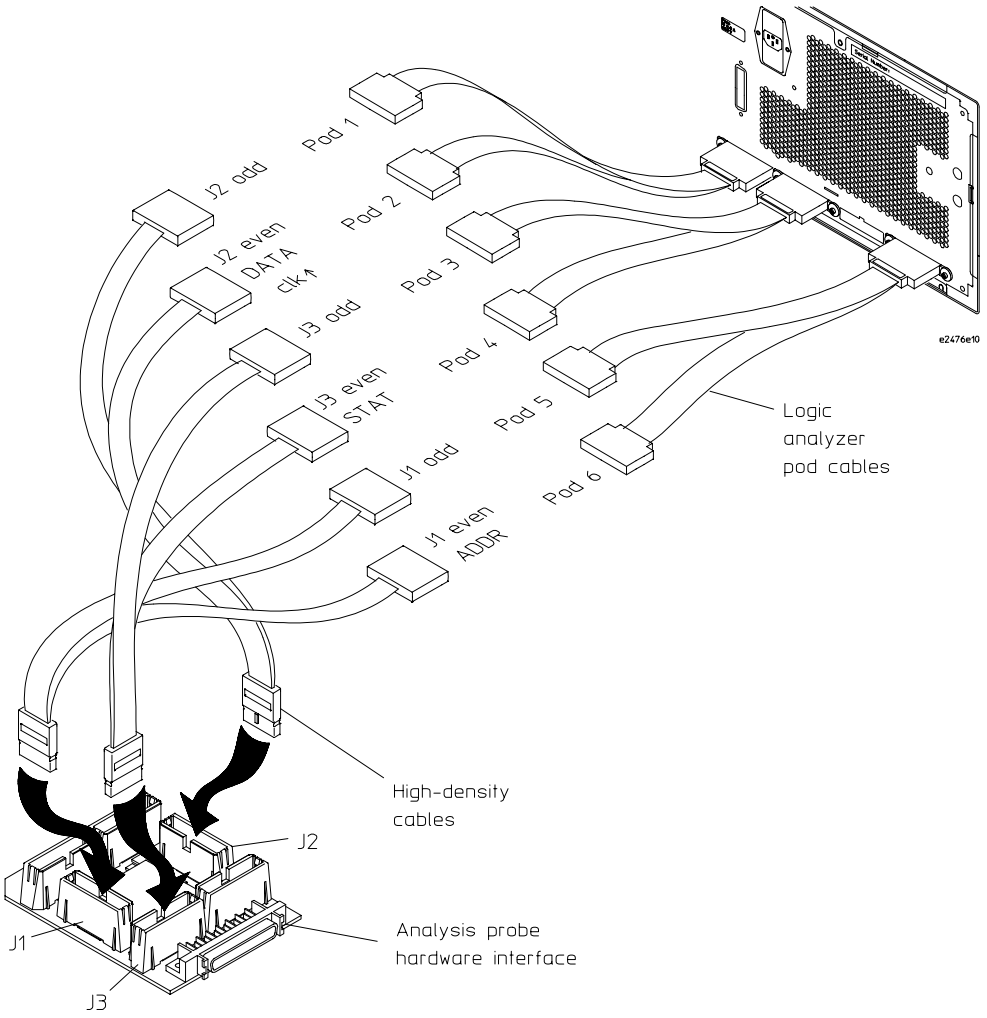
To connect to the HP 1660A/AS/C/CS/CP/E/ES/ EP logic analyzers

Use the figure below to connect the analysis probe to the HP 1660A/C/E logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



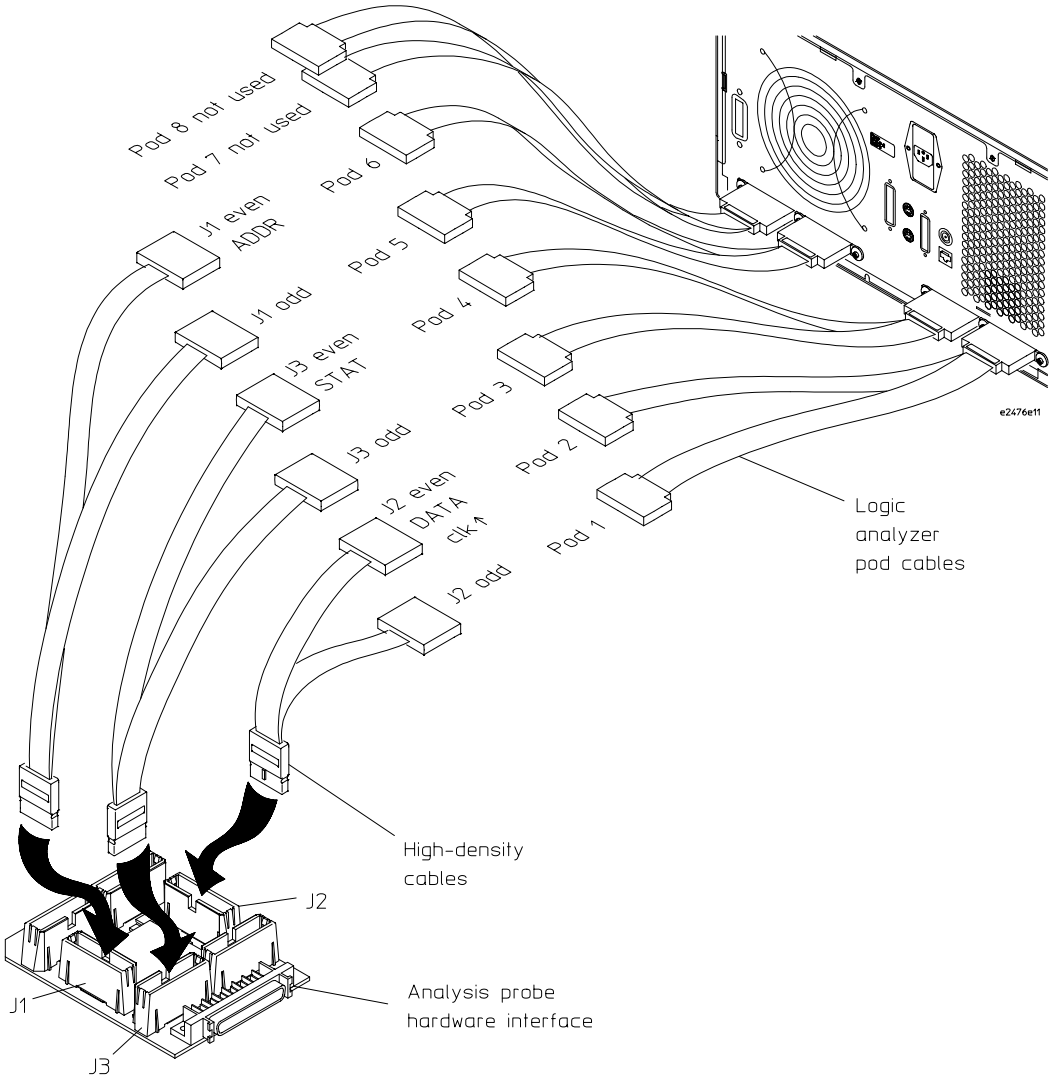
To connect to the HP 1661A/AS/C/CS/CP/E/ES/ EP logic analyzers

Use the figure below to connect the analysis probe to the HP 1661A/C/E logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



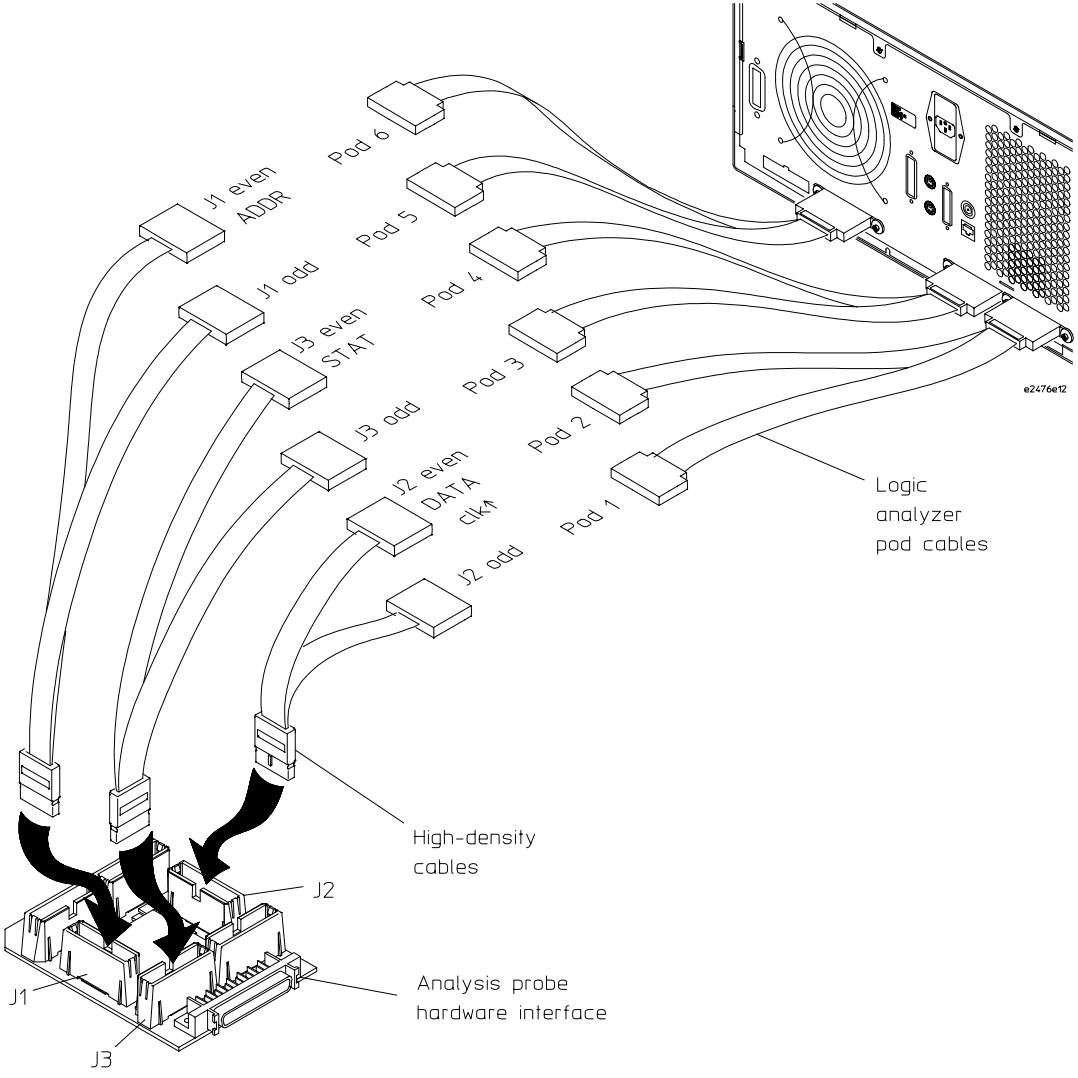
To connect to the HP 1670A/D/E logic analyzer

Use the figure below to connect the analysis probe to the HP 1670A/D/E logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



To connect to the HP 1671A/D/E logic analyzer

Use the figure below to connect the analysis probe to the HP 1671A/D/E logic analyzer. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



Configuring the Logic Analysis System

You configure the logic analyzer by loading a configuration file. The information in the configuration file includes:

- Label names and channel assignments for the logic analyzer
- Inverse assembler or execution tracker file name

The configuration file you use is determined by the logic analyzer you are using. The configuration file names are listed with the logic analyzer connection tables, and in a table at the end of this section.

The procedures for loading a configuration file depend on the type of logic analyzer you are using. There is one procedure for the HP 16600/700 series logic analysis systems, and another procedure for the HP 1660-series, HP 1670-series, and logic analyzer modules in an HP 16500B/C mainframe. Use the appropriate procedures for your analyzer.

To load configuration and inverse assembler files from hard disk—HP 16600/700 logic analysis systems

If you did not use Setup Assistant, you can load the configuration and inverse assembler files from the logic analysis system hard disk.

1 Click on the File Manager icon. Use File Manager to ensure that the subdirectory /hplogic/configs/hp/mpc8xx/ exists.

> If the above directory does not exist, you need to install the MPC8XX Processor Support Package. Close File Manager, then use the procedure on the CD-ROM jacket to install the MPC8XX Processor Support Package before you continue.

2 Using File Manager, select the configuration file you want to load in the /hplogic/configs/hp/mpc8xx/ directory, then click Load. If you have more than one logic analyzer installed in your logic analysis system, use the Target field to select the machine you want to load.

The logic analyzer is configured for MPC860/821 analysis by loading the appropriate MPC860/821 configuration file. Loading the indicated file also automatically loads the inverse assembler with cache-on trace reconstruction. The configuration file names are shown in the following table.

3 Close File Manager.

To load configuration and inverse assembler files from floppy disk—HP 16600/700 logic analysis systems

The preferred method is to install this functionality from the CD-ROM onto the hard disk and load from the hard disk.

To install a configuration and inverse assembler file from the floppy disk that was shipped with your HP analysis probe:

1. Install the floppy disk in the floppy drive on the HP 16600A/16700A-series logic analysis system mainframe.
2. In the Logic Analysis System window, click the **File Manager** icon.
3. In the File Manager window:
   - Set **Current Disk** to Flexible Disk.
   - Set Target to the analyzer you wish to configure.
   - Click the name of the desired configuration file in the Contents frame. The Contents frame lists the configuration files and inverse assembler files available on the floppy disk. These may be either DOS or LIF format files. Either format can be loaded directly into the appropriate logic analyzers.

   Note that the logic analyzers read both DOS and LIF formats. However, only DOS formatted floppy disks can be used to store configurations and data. LIF format floppy disks are read-only.
4. Click **Load**.

   The configuration file you choose will set up the logic analyzer and associated tools. You may see Information, Error, and Warning dialogs that say your configuration has been loaded, and advise you about making proper connections.
5. Click the **Workspace window** icon to see the arrangement of analysis tools in your configuration.
6. Right-click the logic analyzer icon in your configuration and choose its **Setup** button to see the way your configuration file defined the Config, Format, and Trigger options.

   Under the Format tab, buses are labeled, and bits included in each bus are identified by an asterisk "*".

This procedure restores the configuration that was in effect when the configuration file was saved. Because the file was not saved using your system, you may receive error messages about loading the enhanced inverse assembler or about pods that were truncated. Click the Config, Format, and Trigger tabs and modify the configuration to satisfy your measurement desires. Then you can save your customized configuration to DOS format using the **File→**Save Configuration selection in any of your tool windows, or clicking the Save tab in the File Manager. For details about how to save configuration files, open the Help window.

To load configuration and inverse assembler files—other logic analyzers

If you have an HP 1660-series, HP 1670-series, or logic analyzer modules in an HP 16500B/C mainframe use these procedures to load the configuration file and inverse assembler.

The first time you set up the logic analyzer, make a duplicate copy of the flexible disk. For information on duplicating disks, refer to the reference manual for your logic analyzer.

For logic analyzers that have a hard disk, you might want to create a directory such as MPC860 on the hard drive and copy the contents of the floppy onto the hard drive. You can then use the hard drive for loading files.

1. Insert the logic analyzer flexible disk in the front disk drive of the logic analyzer.
2. Select the "Flexible Disk" menu.
3. Configure the menu to "Load" the analyzer configuration from disk.
4. Select the appropriate module (such as "100/500 MHz LA" or "Analyzer") for the load.
5. Use the knob to select the appropriate configuration file.

   Your configuration file choice depends on which analyzer you are using. See table on following page.

6. Execute the load operation to load the file into the logic analyzer.

   The logic analyzer is configured for MPC860 analysis by loading the appropriate MPC860 configuration file. Loading the indicated file also automatically loads the pipelined-systems inverse assembler. For information on the difference between the pipelined and nonpipelined inverse assemblers, refer to “To select a different inverse assembler.”

7. If you are using the HP 16505A Prototype Analyzer, insert the appropriate “16505 Prototype Analyzer” flexible disk (pipelined or nonpipelined) into the disk drive of the prototype analyzer, and update the HP 16505A from the Session Manager. You must close your workspace to run the update.

   The HP 16505A Prototype Analyzer requires software version A.01.30 or higher to work with the inverse assembler.

Logic Analyzer Configuration Files for HP 16600/700-Series Logic Analysis Systems

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I8XXE)
16710/11/12A (1 card)	100 MHz STATE 250 MHz TIMING	c860F_1
16710/11/12A (2 card)	100 MHz STATE 250 MHz TIMING	c860F_2
16715A/16A (2 card)	167 MHz STATE 333/667 MHz TIMING	c860M_2
16715A/16A (3 card)	167 MHz STATE 333/667 MHz TIMING	c860M_3
16717A (2 card)	167/333 MHz STATE 333/667 MHz TIMING	c860M_2
16717A (3 card)	167/333 MHz STATE 333/667 MHz TIMING	c860M_3
16600A	na	c860F_2
16601A	na	c860F_2
16602A	na	c860F_1
16550A (1 card)	100 MHz STATE 250 MHz TIMING	c860F_1
16550A (2 card)	100 MHz STATE 250 MHz TIMING	c860F_2
16554A (2 card)	0.5 M SAMPLE 70/250 MHz LA	c860M_2
16555A/D (2 card)	1.0 M SAMPLE 110/250 MHz LA	c860M_2
16556A/D (2 card)	1.0 M SAMPLE 100/400 MHz LA	c860M_2
16557D (2 card)	2.0 M SAMPLE 135/250 MHz LA	c860M_2

Logic Analyzer Configuration Files for HP 16600/700-Series Logic Analysis Systems (continued)

Logic Analyzer Configuration Files for HP 16600/700-Series Logic Analysis Systems (continued)

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I8XXE)
16554A (3 card)	0.5 M SAMPLE 70/250 MHz LA	c860M_3
16555A/D (3 card)	1.0 M SAMPLE 110/250 MHz LA	c860M_3
16556A/D (3 card)	1.0 M SAMPLE 100/400 MHz LA	c860M_3
16557D (3 card)	2.0 M SAMPLE 135/250 MHz LA	c860M_3

Logic Analyzer Configuration Files for HP 16500/1660/1670-Series Logic Analysis Systems

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I860E)	Configuration File for Cache-on Execution Tracking (I860ET)
16550A (1 card)	100 MHz STATE 250 MHz TIMING	CM860F6	CM860ET1
16550A (2 card)	100 MHz STATE 250 MHz TIMING	CM860F12	CM860ET2
16554A (2 card)	0.5 M SAMPLE 70/250 MHz LA	CM860M8	CM860ET3
16555A/D (2 card)	1.0 M SAMPLE 110/250 MHz LA	CM860M8	CM860ET3
16556A/D (2 card)	1.0 M SAMPLE 100/400 MHz LA	CM860M8	CM860ET3
16557D (2 card)	2.0 M SAMPLE 135/250 MHz LA	CM860M8	CM860ET3

Logic Analyzer Configuration Files for HP 16500/1660/1670-Series Logic Analysis Systems (continued)

Logic Analyzer Configuration Files for HP 16500/1660/1670-Series Logic Analysis Systems (continued)

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I860E)	Configuration File for Cache-on Execution Tracking (I860ET)
16554A (3 card)	0.5 M SAMPLE 70/250 MHz LA	CM860M12	CM860ET4
16555A/D (3 card)	1.0 M SAMPLE 110/250 MHz LA	CM860M12	CM860ET4
16556A/D (3 card)	1.0 M SAMPLE 100/400 MHz LA	CM860M12	CM860ET4
16557D (3 card)	2.0 M SAMPLE 135/250 MHz LA	CM860M12	CM860ET4
1660A/AS/C/CS/E/ES/EP	na	CM860M8	na
1661A/AS/C/CS/E/ES/EP	na	CM860F6	na
1670A/D/E	na	CM860M8	na
1671A/D/E	na	CM860F6	na

4

Designing Connectors for Custom Probing for the Inverse Assembler

The HP E2477A Inverse Assembler /Execution Tracker Software uses the same software as the HP E2476A Analysis Probe. This chapter shows you how to design logic analyzer connectors on your target system for use with the HP E2477A.

If you are using an HP E2476A analysis probe, skip this chapter.

This chapter consists of the following section:

- Using High-Density Connectors
- High-Density Connector Mechanical Specifications

Using High-Density Connectors

High-density Mictor (Matched Impedance ConnecTOR) connectors are recommended for connecting the target system to the logic analyzer because they require less board space and provide higher signal integrity than medium-density connectors. Each connector carries 32 signals and two clocks.

- Each 32-signal high-density header connector requires approximately 1.1" x 0.4" of printed-circuit board space.
- The part number for the high-density Mictor connector is: AMP P/N 2-767004-2 or HP: 1252-7431.
- Each Mictor connector requires one HP E5346A high-density termination adapter cable to attach to the logic analyzer. This is a Y-cable where the single end connects to the high-density header connector, and each of the two opposite ends connects to a logic analyzer pod.
- Any probed signal line must be able to supply a minimum of 600 mV to the probe tip and handle a minimum loading of 90 KOhms shunted by 10 pF. The maximum input voltage for the logic analyzer is +/- 40 volts peak.
- If a printed-circuit board already has a header connector attached, but the signal pinouts do not match the requirement, an adapter (HP part number E5346-60002) can be used to route the signals to the correct pods.
- A plastic shroud (HP part number E5346-44701) is available to secure the mechanical connection of the high-density cable to the Mictor header connector.

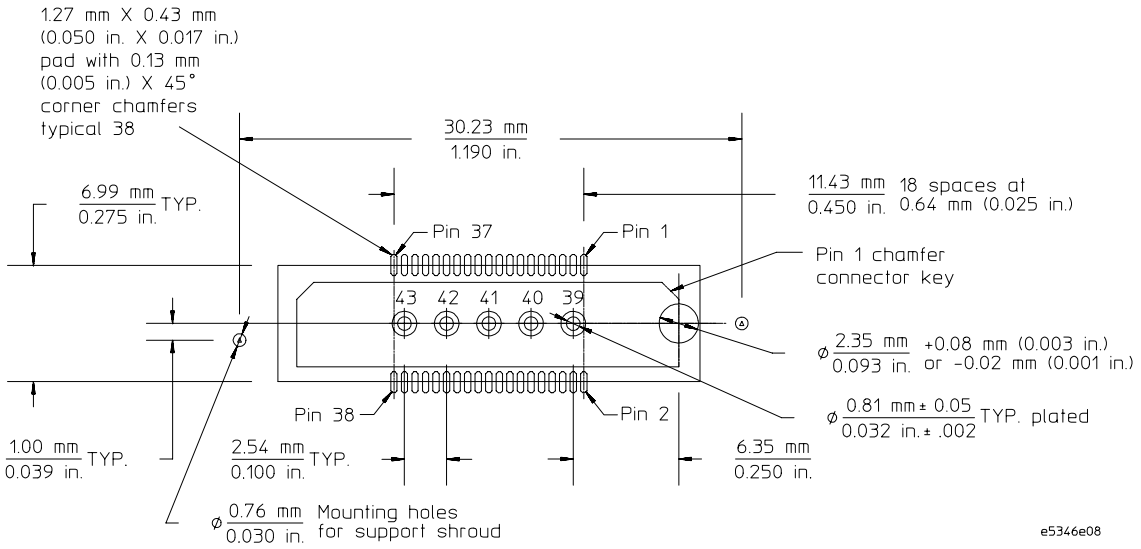
See Also

More information on this connector is available in the document *HP E5346A High-Density Termination Adapter*, HP part number 5965-5475E. This document is available in Portable Document Format (PDF) from the web site:

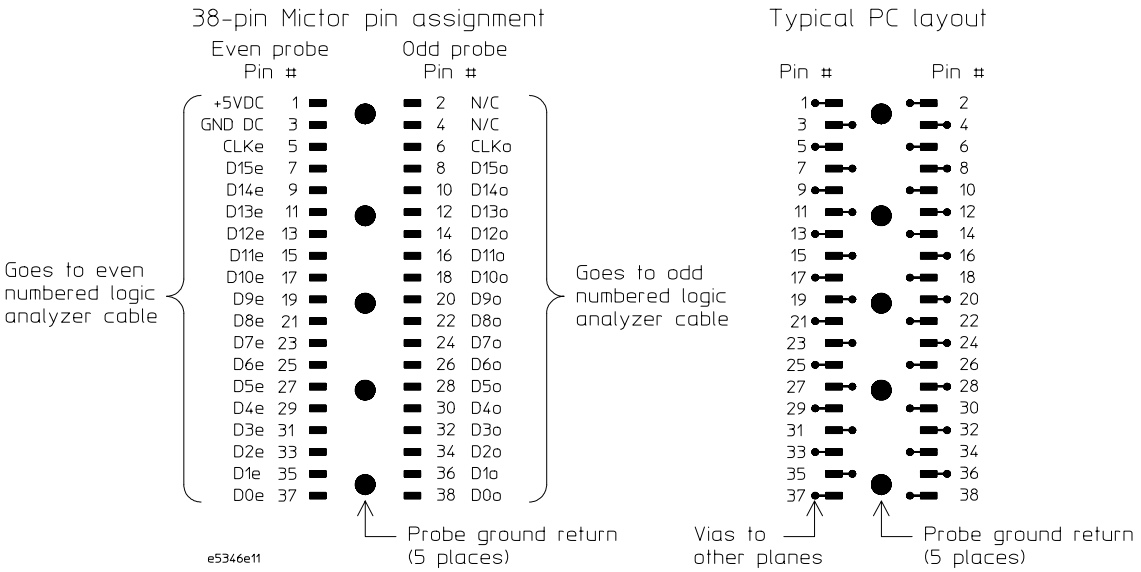
http://www.tmo.hp.com/tmo/datasheets/English/HPE5346A.html

High-Density Connector Mechanical Specifications

Dimensions of the AMP Mictor 2-767004-2 surface mount connector are shown below. The holes for mounting a support shroud are off-center to allow 0.40 in (1.20 mm) centers when using multiple connectors.



The high-density connector pin assignment and recommended circuit board routing are shown below.



Five center inline pins on the connector are the signal ground returns and must be connected to ground.

5

Analyzing the MPC860/821 with an HP 16600A/16700A-series Logic Analyzer

The information in this chapter is specific to systems using HP 16600A/16700A-series logic analyzers. For systems using HP 1660A/1670A/16500B/C-series logic analyzers, see Chapter 6, “Analyzing the MPC860/821 with an HP 1660A/1670A/16500B/C-series Logic Analyzer,” beginning on page 117.

This chapter describes modes of operation for the HP E2476A analysis probe. It also describes data, symbol encodings, and information about cache-on trace reconstruction using Motorola S-records. Except for the modes of operation, all sections apply to both the HP E2476A analysis probe and the HP E2477A inverse assembler software.

The information in this chapter is presented in the following sections:

- Modes of operation
- Modes of analysis
- Logic analyzer configuration
- Using the inverse assembler
- Using cache-on trace reconstruction

Modes of Operation

The HP E2476A analysis probe can be used in three different operating modes: State-per-ack, State-per-clock, or Timing. The HP E2477A inverse assembler software can be used for State-per-ack and State-per-clock analysis. The following sections describe these operating modes and how to configure the logic analyzer for each mode.

State-per-ack mode

In State-per-ack mode, the logic analyzer uses clock store qualification to capture only address and data-acknowledge cycles. This is the default mode set up by the configuration files.

State-per-ack mode provides the greatest information density in the logic analyzer acquisition memory.

State-per-clock mode

In State-per-clock mode, every clock cycle is captured by the logic analyzer, including idle and wait states between and during tenures. To configure your logic analyzer for State-per-clock mode:

1. Click on the logic analyzer icon.
2. Select “Setup...” from the menu. The “Sampling” tab will be active on the window that appears.
3. Set the clocks as follows: M=off, L=off, K=rising edge, J=off.
4. Select the Trigger tab and ensure that step 2 of the flow diagram says “Store anystate”.

Timing mode

In Timing mode, the logic analyzer samples the microprocessor pins asynchronously. To configure your logic analyzer for timing analysis:

1. Click on the logic analyzer icon.
2. Select “Setup...” from the menu. The “Sampling” tab will be active on the window that appears.
3. Select the Timing Mode button.

Modes of Analysis

The inverse assembler offers two modes of analysis for MPC860/821 microprocessors: traditional inverse assembly, and inverse assembly with cache-on trace reconstruction. The mode is set in the **Invasm Preferences** window using the **External Bus Decoding** dialog.

Inverse assembly analysis

The inverse assembler lets you obtain displays of MPC860/821 operations in PowerPC instruction mnemonics, as described in *PowerPC Microprocessor Family: The Programming Environments for 32-Bit Microprocessors*. In addition, information that is processed in cache may be displayed using the cache-on trace reconstruction feature of the inverse assembler.

The inverse assembler requires the HP 16600A/700A-series logic analyzers. It provides much more information when used together with the HP B4620B Source Correlation Tool Set. Source correlation performs a correlation of the addresses from cache with the high-level code execution.

Cache-on trace reconstruction

Cache-on trace reconstruction lets you track instructions executed in the cache. The logic analyzer displays the data in instruction-type (branch, sequential, etc.) format. If an S-record is loaded, the data is inverse assembled into mnemonics.

For cache-on trace reconstruction, set the operating mode to State-per-clock and enable show cycle disassembly.

Logic Analyzer Configuration

The following sections describe the logic analyzer configuration as set up by the configuration files.

NOTE: It is strongly recommended that you do not change the setup related to the MPC860/821 sampling, format, pod assignment or configuration dialogs. The Setup Assistant will configure the logic analyzer for making measurements of the MPC860/821.

Trigger sequence

The Trigger sequence is set up by the software to store all states.

NOTE: If you modify the trigger sequence to store only selected bus cycles, incorrect or incomplete disassembly may be displayed.

Using Labels

The configuration files contain predefined format specifications. These format specifications include all labels for monitoring the microprocessor. The tables on the following pages show the signals used in the STAT label and the predefined symbols set up by the configuration files.

Labeling Conventions

The HP logic analyzers and the PowerPC use opposite conventions to designate individual signals on a bus. In PowerPC nomenclature, bit 0 is the most significant; in the logic analyzers, bit 0 is the least significant. In PowerPC, A0 is the most significant bit of the address bus; on the analyzer, this bit is called ADDR31.

Most Significant	Least Significant	
A0	**A31**	***PowerPC***
ADDR31	**ADDR0**	***Logic Analyzer***

This may cause confusion in the waveform window when using Channel Mode Sequential or Individual.

Do not modify the ADDR, DATA, or STAT labels in the format specification if you want inverse assembly. Changes to these labels may cause incorrect or incomplete inverse assembly.

Status Encoding

Each bit of the STAT label is described in the table below. The inverse assembler uses STAT bits BR, BG, BB, TS, TA, TEA, R/W, BURST, BI, and TSIZ. The signal-to-connector tables in the "Hardware Reference" chapter list all the MPC860/821 signals probed and their corresponding analyzer channels.

Status Bit Description

Status Bit	Description
BR	A slave controller asserts Bus Request to indicate that it wants access to the memory bus.
BG	The memory system asserts Bus Grant to allow the MPC860/821 onto the address bus.
BB	Bus Busy indicates that the an external master controller has the bus.
TS	The MPC860/821 asserts TS for one cycle to commence a transaction.
TA	The memory system asserts TA to acknowledge a data transaction.
TEA	The memory system may assert TEA to indicate a transfer error, such as an unmapped part of the address space.
R/W	R/W is high for a read, low for a write.
STS	Special Transfer Start is an internal transfer indicator.
BURST	BURST indicates that the current initiated transfer is a burst one. The BURST signal is negated prior to the expected last data beat of the burst transfer.
BDIP	Burst Data In Progress indicates the last beat of a burst transfer.
BI	Burst Inhibit indicates that the slave device addressed in the current burst transaction is unable to support burst transfers.
AT[0, 2, 3]	Address Type provides further information about the current transfer. For a read, they indicate whether instructions or operands are being fetched.
TSIZ [0:1]	Indicates the size for the data transfer.
VF[0:2]	Visible Instruction Queue Flushes Status tracks program flow.
VFLS[0:1]	Visible History Buffer Flushes Status tracks program flow.

Predefined Logic Analyzer Symbols

The configuration software sets up symbol tables on the logic analyzer. The tables define a number of symbols which make several of the STAT fields easier to interpret. The following table lists the symbol descriptions.

Symbol Description

Label	Symbol	Encoding
R/W	rd	1
	wr	0
TSIZ	4 byte	00
	1 byte	01
	2 byte	10
TEA	(blank)	1
	tea	0
TA	(blank)	1
	ta	0
BURST	(blank)	1
	burst	0
BI	(blank)	1
	bi	0
BR	(blank)	1
	br	0
BG	(blank)	1
	bg	0
VFLS	0 flsh	00
	1 flsh	01
	2 flsh	10
	debug	11

Label	Symbol	Encoding
BB	(blank)	1
	bb	0
BDIP	(blank)	1
	bdip	0
STS	(blank)	1
	sts	0
AT0	CPM	1
	CPU	0
AT2	data	1
	instr	0
AT3	trace	1
	resrv	0
TS	(blank)	1
	ts	0
VF	000	000
(see note)	010	001
	100	010
	110	011
	001	100
	011	101
	101	110
	111	111

Note: The VF symbols display the proper order of these pins. These symbols compensate for the MPC860/821 multiplexing scheme.

To qualify stored data

The configuration file sets up the logic analyzer clock.

The logic analyzer identifies valid states based on the clock signals. The logic analyzer refers to the system clock as the K clock, and the $\overline{TA}$ signal as the M clock.

The default clocking combination “K rising and M low” latches address and data when the system clock is rising and $\overline{TA}$ is asserted.

The memory controller in the MPC860/821 will allow data to be latched on the falling edge of the system clock (K falling). If your data appears incorrect, check the processor’s memory controller registers to ensure that the clocking configurations of the processor and the logic analyzer match.

For State-per-clock acquisition, change the clock qual Q1 to Off.

Using the Inverse Assembler

This section discusses the general output format of the inverse assembler and processor-specific information.

Traditional inverse assembly, in which the external processor bus states are captured and decoded, may be implemented by disabling the target's cache. However, this will slow the target significantly, and may induce timing related problems. The target system's performance will be much better if the cache-on trace reconstruction feature is enabled when using the inverse assembler.

Using Cache-On Trace Reconstruction

The inverse assembler uses show cycles, the $\overline{\text{STS}}$/VF/VFLS signals, and program image files to decode captured MPC860/821 execution into complete program trace. When compared to data that comes straight from the logic analyzer, the data from the inverse assembler:

- Contains code that executes out of cache
- Has unexecuted prefetches removed or shown with prefetch marking
- Shows the actual execution times of instructions

The data from the inverse assembler contains only code that has been executed by the microprocessor. Read and write cycles captured by the logic analyzer are unchanged.

An instruction in the listing may be preceded by an asterisk to indicate prefetch, or by a question mark to indicate that it may have been prefetched.

Inverse Assembler Modes of Operation

The table below describes the various modes in which the inverse assembler can operate. An explanation of how to set up the inverse assembler to operate in these modes follows.

Inverse Assembler Modes of Operation

IA Cache Decoding	Data Bus Connected	S-Record Loaded	Result
off	no	no	Error message: opcode retrieval requires that the data bus is connected or an S-Record is loaded.
off	no	yes	Opcodes are fetched from the S-Record and decoded into instruction mnemonics. R/W data will not be displayed.
off	yes	no	Traditional Inverse Assembly Opcodes are fetched from the data bus and decoded into instruction mnemonics. R/W data will be displayed.
off	yes	yes	Opcodes are fetched from the S-Record and decoded into instruction mnemonics. R/W data will be displayed.
on	no	no	Show cycle messages are displayed. R/W data will not be displayed.
on	no	yes	Show cycles provide the address so opcodes can be fetched from the S-Record and decoded into instruction mnemonics. R/W data will not be displayed.
on	yes	no	Show cycle messages are displayed. R/W data will be displayed.
on	yes	yes	Show cycles provide the address so opcodes can be fetched from the S-Record and decoded into instruction mnemonics. R/W data will be displayed.

NOTE: Read and write states are always indicated regardless of whether the data bus is connected. When the data bus is connected, read/write data will also be displayed.

To use the Invasm menu

The Invasm menu provides four choices: Load, Preferences, Filter, and Options. Access the Invasm menu in the listing window.

You must use the Preferences dialog to configure the inverse assembler to match the microprocessor memory controller configuration. The other dialogs assist in analyzing and displaying data. The following sections describe these dialogs.

Loading the Inverse Assembler

The Load dialog lets you load a different inverse assembler and apply it to the data in the Listing window. In some cases you may have acquired raw data; you can use the Load dialog to apply an inverse assembler to that data.

Setting the Inverse Assembler Preferences

Why the configuration is necessary

Because critical information about what type of data is being accessed through a memory bank is stored in internal registers, the inverse assembler needs to be given some information about how the memory system is set up.

The memory controller operates by mapping every address to one of eight memory banks. Each memory bank can be set up to drive different external signals, to have different write permissions, etc. The memory banks are numbered from 0 to 7. Memory bank 0 has the highest priority and bank 7 has the lowest.

The base register and option register for each memory bank hold information that describes the width of the memory accessed through that bank, the type of data, and the addresses that will be accessed through that bank. Since this information is not given on external signals, the inverse assembler provides a preferences window to enter this information so that the data decode can be as accurate as possible.

Finding memory bank information using a debugger

You can use a debugger to examine the base register and option register to determine what values to enter in the preferences window.

The Base Address fields should be set to match the upper 17 bits of the base registers.

The Address Mask fields should be set to match the upper 17 bits of the option registers.

The memory port sizes can be determined by looking at bits 20-21 (assuming bit 31 is least significant) of the base registers.

The address type bits can be used to limit access to instructions or data. Your target may or may not be configured to use address type comparisons. Look at bit 19 of the option registers (this is the address type mask), if this bit is cleared (0) then your target is not configured to do address type comparisons. If the bit is set then bit 19 of the base register will be set to 0 if the memory bank accesses instructions and to 1 if the memory bank accesses data.

Finding memory bank information at compile time

NOTE: If the AT2 signal is used for instruction/data decoding, cycle type information is not necessary.

When compiling the code that will be analyzed, direct the linker to locate all instructions and all data (constants, variables, the stack) in separate memory "blocks". "Blocks" of memory can be differentiated as long as one of the upper 17 address bits differ. Then, set up the preferences window so that memory bank 0 will decode the instruction states and memory bank 1 will decode the data states.

Example

Assume that the code will run from DRAM that has a 32 bit port size. Compile the code specifying to the linker to place text or code at address 0x00020000 and data at address 0x00100000.

Set up the first two memory banks in the preferences window as follows:

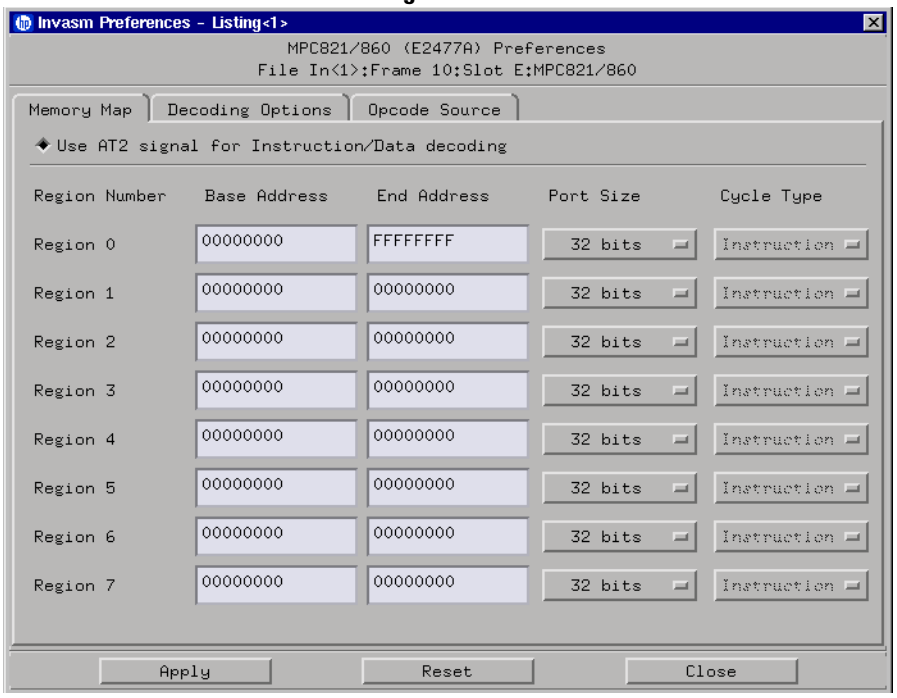
Region	Base Address	End Address	Port Size	Cycle Type
Region 0	00020000	0002FFFF	32 bit	instruction
Region 1	00100000	001FFFFF	32 bit	data

The inverse assembler will now interpret any read from addresses 0x00020000 - 0x0002FFFF as instruction reads and any reads from 0x00100000 - 0x001FFFFF as data reads.

To set the memory map preferences

It is necessary to configure the memory map in the Preferences dialog before using the inverse assembler.

Inverse Assembler Preferences Dialog



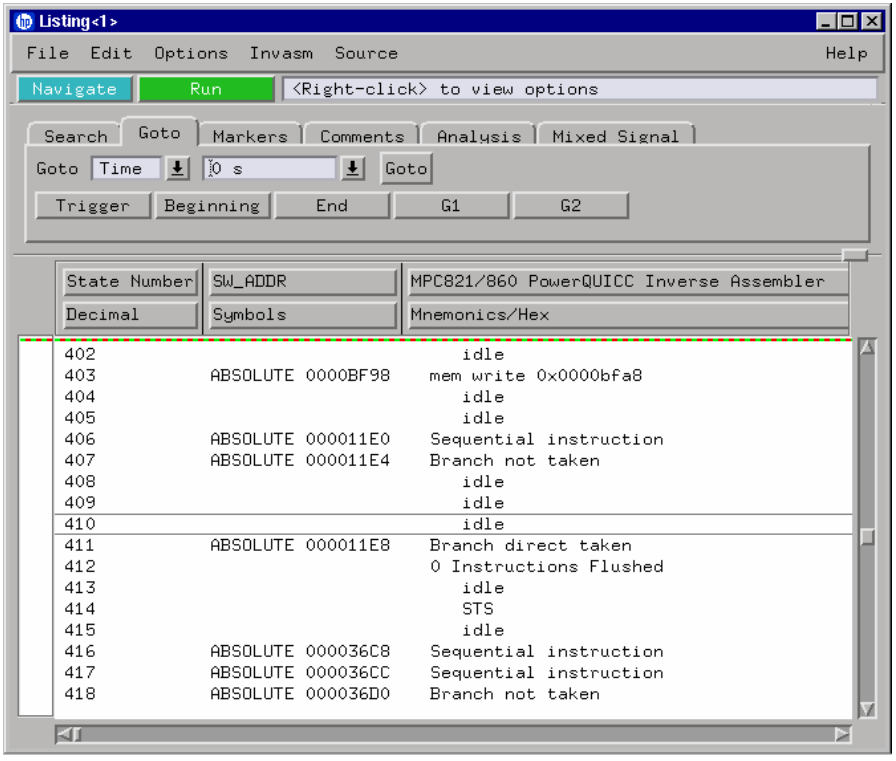
Click **Apply** when you have finished configuring the memory map.

Cache-on trace reconstruction uses the AT2 signal to distinguish instructions from data on the target's internal bus. Select the **Use AT2 signal for Instruction/Data decoding** button when using cache-on trace reconstruction.

Enabling show cycle disassembly

To enable show cycles write 0, 1, or 5 to register ICTRL [ISCT_SER]. Also, $\overline{\text{STS}}$ functionality of OP2/MODCK1/$\overline{\text{STS}}$ must be enabled by writing 01 or 11 to register SIUMCR[DBGC]. This also enables IP_B2/AT2 to function as AT2.

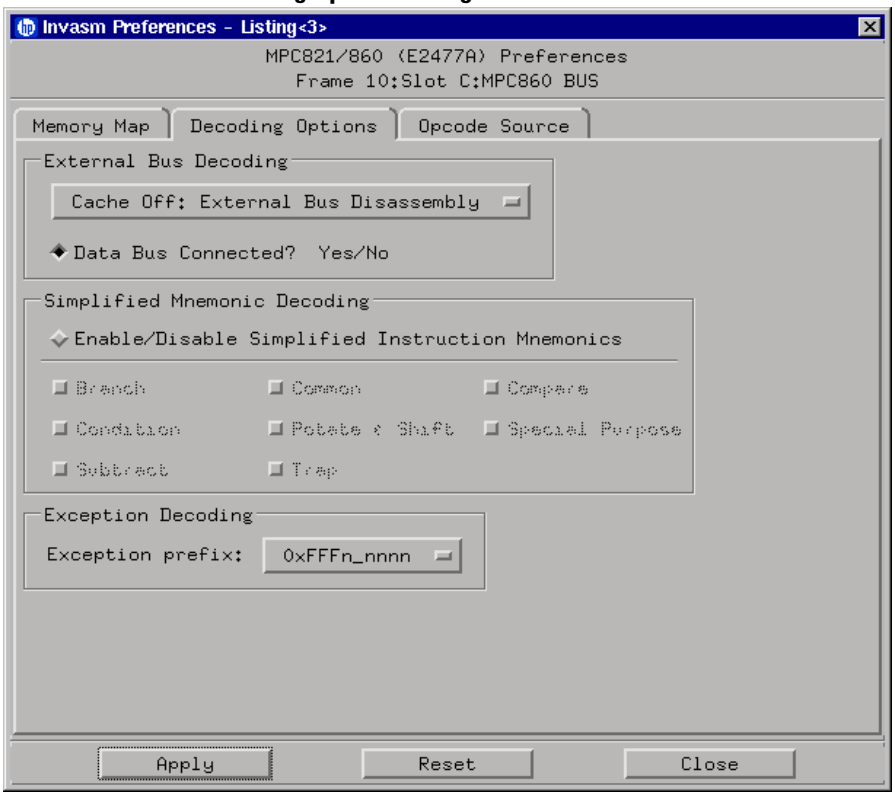
Show Cycle Messages are Displayed: Cache On, No S-Record Loaded



The AT2 signal is used to discern instructions from data on the target system's external bus. For traditional inverse assembly, if not using cache-on trace reconstruction and not using IP_B2/AT2 to function as AT2, it is necessary to fill out the cycle type column in the memory map.

To set the decoding options preferences

Inverse Assembler Decoding Options Dialog



External Bus Decoding. Choose **Cache Off: External Bus Disassembly** for traditional inverse assembly or **Cache On: Show Cycle Disassembly** for cache-on trace reconstruction.

Data Bus Connected. Read and write states are always indicated regardless of whether the data bus is connected. However, when the data bus is connected, read/write data will also be displayed. See “Inverse Assembler Modes of Operation” on page 100

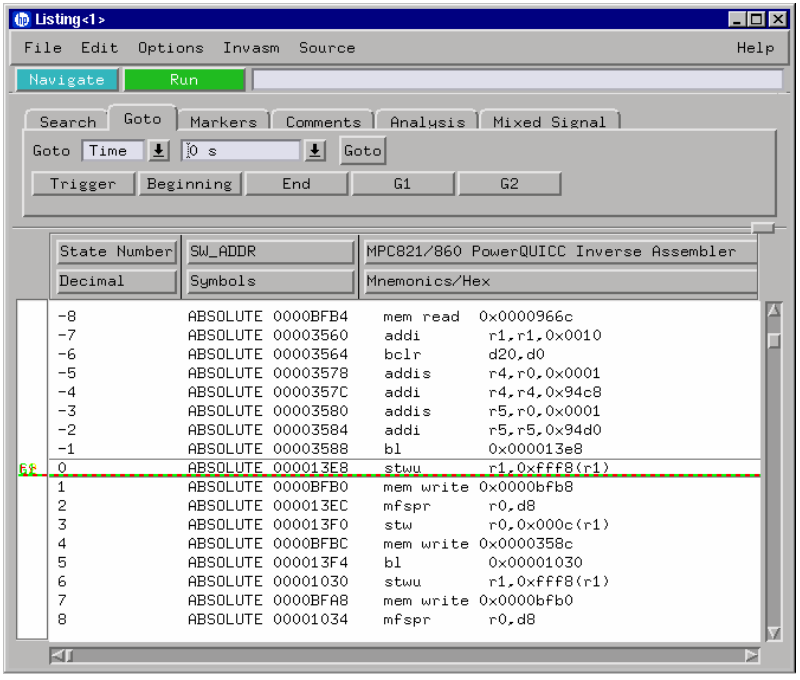
Simplified Mnemonic Decoding. PowerPC assemblers support a number of simplified mnemonics for some popular assembly language instructions, as described in Appendix F of the *PowerPC Microprocessor*

Family: The Programming Environments for 32-Bit Microprocessors. The inverse assembler will show those extensions if you wish to see them. By enabling the Simplified Mnemonic Decoding, you can select which types of simplified mnemonics will be shown. Click the options for the simplified mnemonics you desire.

Displaying the simplified mnemonics may help you to get a better idea of what a particular instruction is really doing. For example, an "or r1,r1,r1" instruction is simplified to a "nop."

Exception Decoding. the inverse assembler can output the types of exceptions that occur. The PowerPC architecture allows for two locations of the exception vector table. You can determine which location is set up for your target by looking at the IP bit (bit 25) of the MSR register. This can be done by examining the initialization code or by using an emulator to view the MSR register.

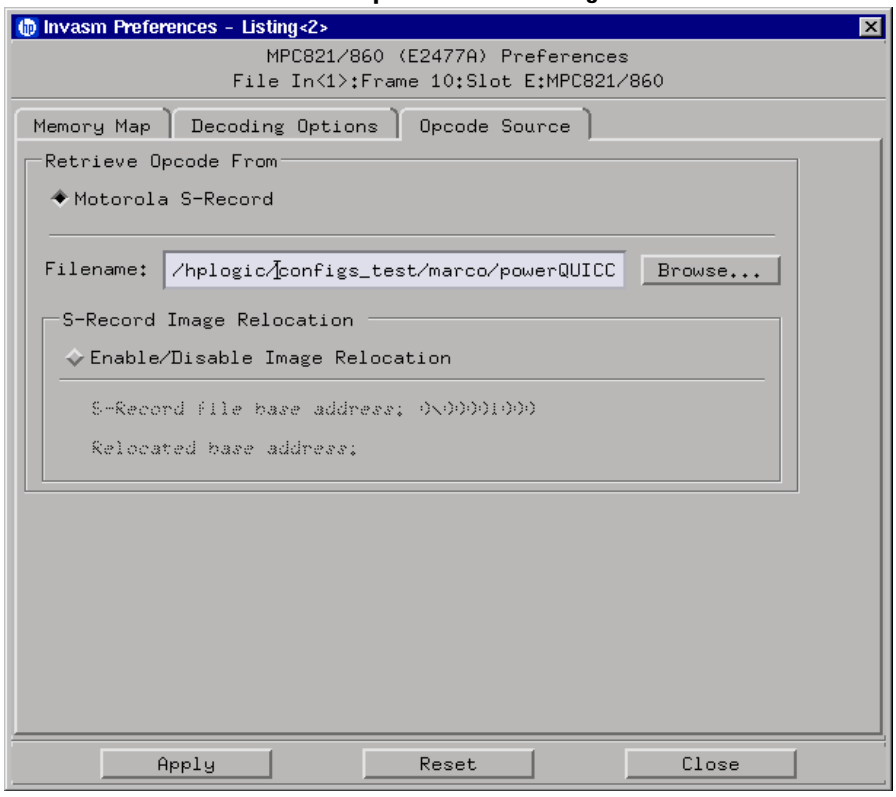
Listing Window Showing Trace with Data Bus Connected: Cache Off



Read and write data is displayed because the data bus is connected.

To set the opcode source preferences

Inverse Assembler Preferences Opcode Source Dialog

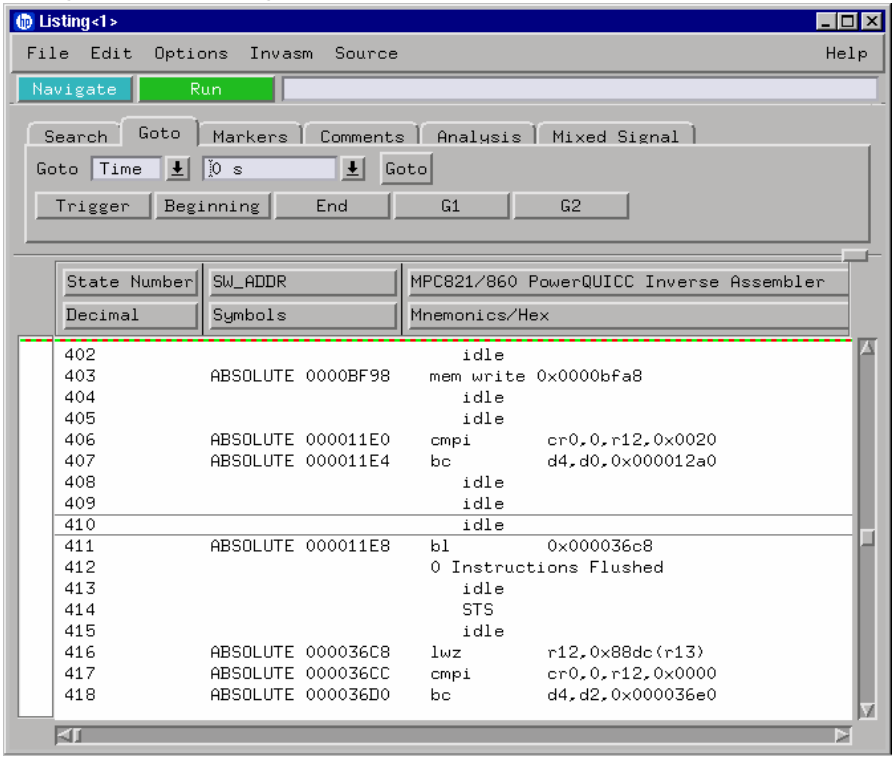


Specifying use of Motorola S-record. Select "**Motorola S-Record**" in the "**Retrieve opcode from**" dialog to have a Motorola S-record supply execution trace information to the cache-on trace reconstruction tool. Use the **Browse...** button to locate the S-record file.

S-Record Image Relocation. The Image relocation portion of the dialog box allows you to relocate the SREC file to some other location in memory. This is useful when the loaded file is moved to some other location in memory. For example, the starting address in the SREC file is 1000. However, memory starting at 1000 is relocated to 5000. In order for the inverse assembler to retrieve the correct data, the entire SREC file must be relocated to 5000. Enter the relocated base address; all the resulting offsets will be calculated by

the inverse assembler.

Listing Window Showing Instruction Mnemonics: Cache On, S-Record Loaded



Show cycles provide the information so the opcodes can be fetched from the S-record. The information is decoded into instruction mnemonics.

Display Filtering

The inverse assembler lets you Show or Suppress several types of states. This dialog is called display filtering. States can be filtered according to what type of cycle the state is, or according to which memory bank was accessed for the cycle.

The show/suppress settings do not affect the data that is stored by the logic analyzer; they only affect whether that data is displayed or not. You can examine the same data with different settings, for different analysis requirements.

This dialog allows faster analysis in two ways. First, you can filter unneeded information out of the display. For example, suppressing idle states will show only states in which a transaction was completed.

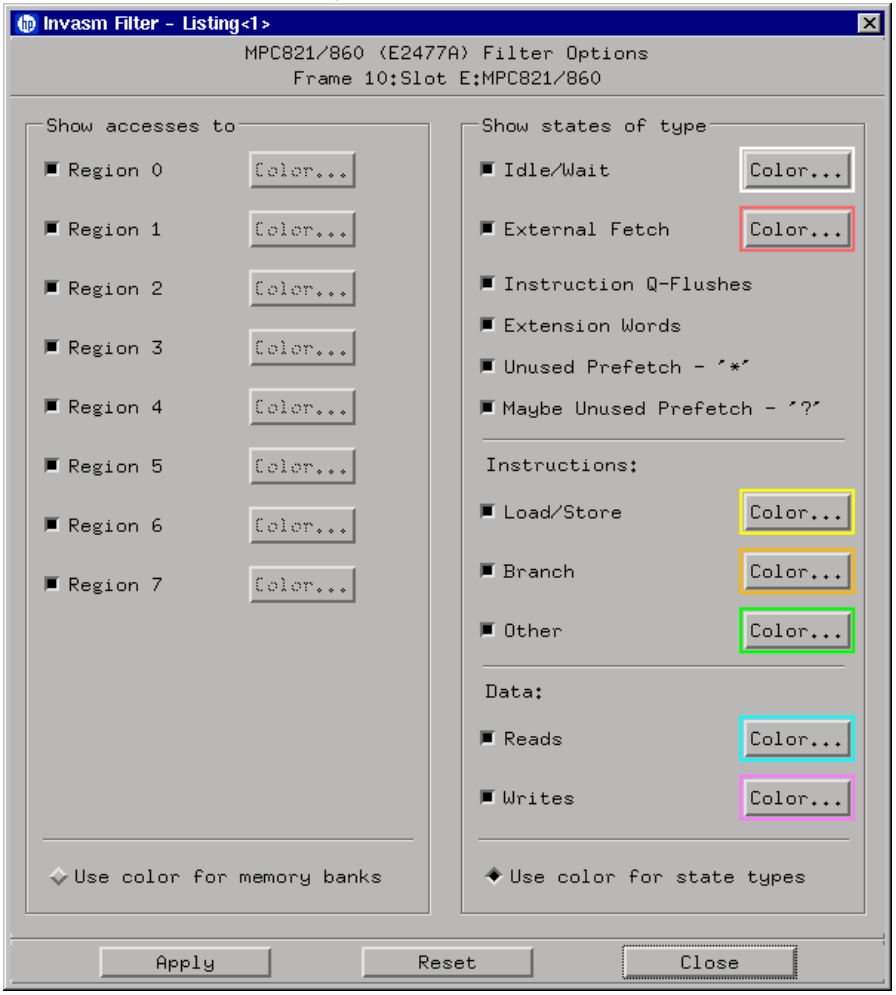
Second, you can isolate particular operations by suppressing all other operations. For example, you can show branches, with all other states suppressed, allowing quick analysis of branch instructions.

HP 16600A/700A-series logic analysis systems provide one additional feature for analyzing data. Instead of (or in addition to) showing or suppressing states, the selected states can also be shown in color.

> Color can only be used for distinguishing either memory bank accesses or cycle types, but not both at the same time.

The following figure shows the inverse assembler filter dialog.

Inverse Assembler Filter Dialog



Options

The options dialog lets you change the width of the symbols in the disassembly column. It also allows you to display symbols (globals), hex, or line numbers.

To disable the instruction cache on the MPC860/821 for traditional inverse assembly

When the instruction cache is enabled, many PowerPC instructions are executed from the cache and do not appear on the external bus. To get an execution trace on the bus, the instruction cache can be disabled. This must be done in supervisor mode.

To disable the cache with the emulation module:

Use your debugger or the Emulation Control Interface to set the IC_CST register (or the DC_CST register, to disable the data cache).

Register values for controlling the cache

Value	Meaning
0400 0000	Disable
0a00 0000	Unlock all
0c00 0000	Invalidate
0200 0000	Enable

To disable the cache with code:

- Disable the cache with the following code:

```
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprICCST, r0; disable Instr Cache
isync
```

- To also disable the data cache use:

```
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprDCCST, r0; disable Data Cache
isync
```

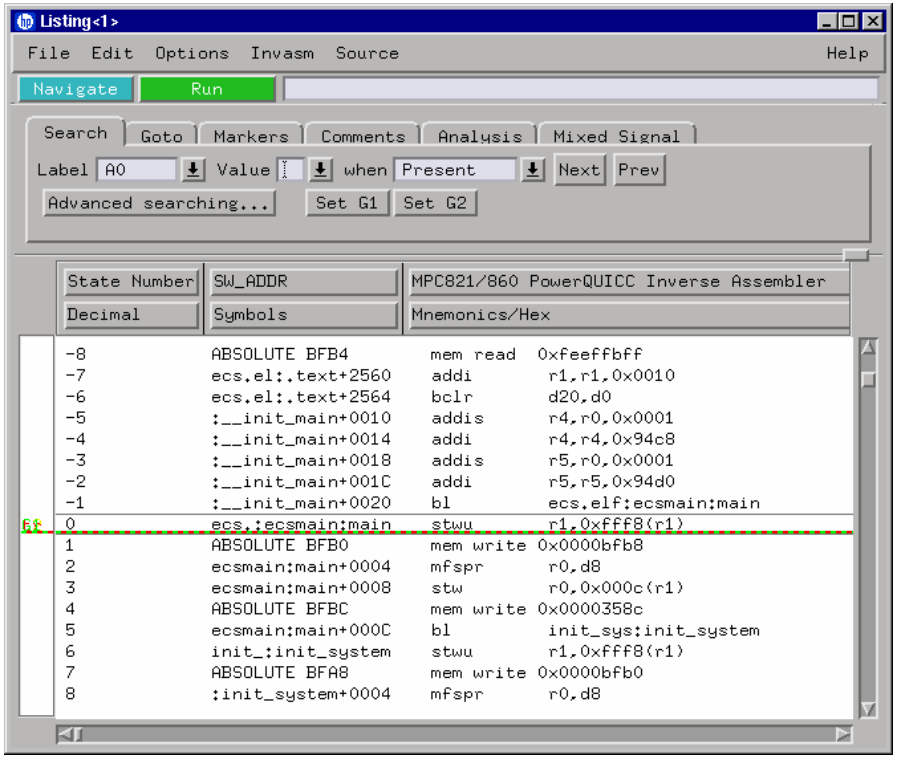
- To invalidate and disable the caches use:

```
addir2, 0, 0X019
addir1, 0, 0X06
slwr0, r1, r2
mtsprICCST, r0; invalidate instr cache
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprICCST, r0; disable Instr Cache
addir2, 0, 0X019
addir1, 0, 0X06
slwr0, r1, r2
mtsprDCCST, r0; invalidate Data cache
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprDCCST, r0; disable Data Cache
isync
```

To display captured state data

The logic analyzer displays captured state data in the Listing window. The inverse assembler display is obtained by setting the base for the DATA label to Invasm. The following figure shows a typical Listing window.

Listing window



The column on the left of the inverse assembly data display is the least significant hexadecimal digit of the current address. On the HP 16600A/700A-series logic analysis systems, the entire synthesized address appears under the label “SW_ADDR”. You can observe the actual address bits presented by the MPC860/821 under the ADDR label.

Inverse assembler output format

The following paragraphs explain the operation of the inverse assembler and the results you can expect under certain conditions.

Interpreting Data

General purpose registers are displayed as r0, r1, r2...r31. Special purpose registers are displayed using their mnemonic.

Most numerical data is displayed in hexadecimal, for example, “stwu r1,0xfff8(r1).” Bit numbers and shift counts are displayed in decimal with a dot suffix, for example, “cror 31. 31. 31.”

A few instructions display their operands in binary with a “%” prefix, for example, “mtfsfi 4 %0101.”

The inverse assembler decodes the full PowerPC instruction set architecture, including 64-bit mode instructions and optional instructions not implemented on the MPC860/821. When these unimplemented opcodes are encountered, the listing displays “illegal opcode.”

An instruction word of 00000000 is decoded as “illegal opcode.” Otherwise, if an opcode is invalid, it is shown as “unknown opcode.”

Branch Instructions

If the address of a branch relative instruction is known, its target is presented as an absolute hex address (or as a symbol if it matches an ADDR pattern or range symbol). If the address of a branch relative instruction is not known, its target is displayed as a hexadecimal offset such as +00000C30 or -00000048.

SW_ADDR Label

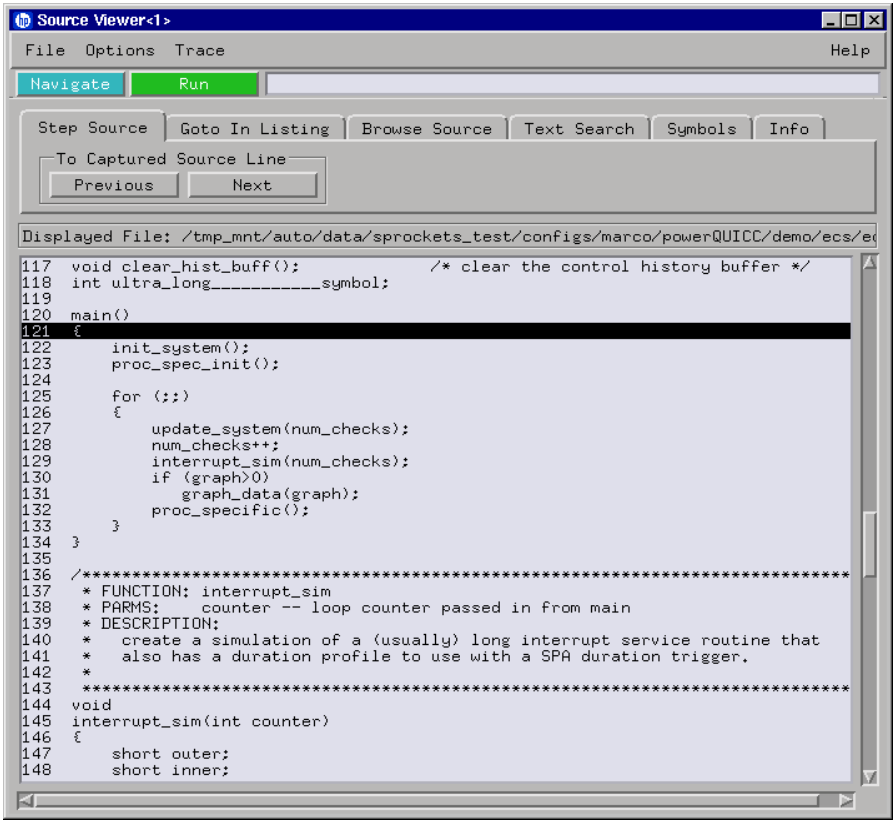
When an HP 16600A/700A-series logic analysis system is being used, the inverse assembler generates a “SW_ADDR” field. This field is the Software Address generated by the inverse assembler.

The SW_ADDR label cannot be used exactly like other labels. For example, when loading symbols, you will notice that the SW_ADDR label is not in the list of labels that the symbols can be loaded into. Symbols should still be loaded into the ADDR label. The main purpose of the SW_ADDR label is for correlation of the listing with source code using the HP B4620B Source Correlation Tool Set.

Displaying Data with the HP B4620B Source Correlation Tool Set

Source correlation correlates the addresses from cache with the high-level code execution. The figure below shows execution of data that is correlated to the data shown on page 113.

Source Correlation Tool Set Data



6

Analyzing the MPC860/821 with an HP 1660A/1670A/16500B/C-series Logic Analyzer

The information in this chapter is specific to systems using HP 1660A/1670A or 16500A/B-series logic analyzers. For systems using HP 16600/16700-series logic analyzers, see Chapter 5, “Analyzing the MPC860/821 with an HP 16600A/16700A-series Logic Analyzer,” beginning on page 89.

This chapter describes modes of operation for the HP E2476A analysis probe. It also describes data, symbol encodings, and information about the inverse assembler and cache-on execution tracker. Except for the modes of operation, all sections apply to both the HP E2476A analysis probe and the HP E2477A inverse assembler/execution tracker software.

The information in this chapter is presented in the following sections:

- Modes of operation
- Modes of analysis
- Logic analyzer configuration
- Using the inverse assembler
- Using the cache-on execution tracker

Modes of Operation

The HP E2476A analysis probe can be used in three different operating modes: State-per-ack, State-per-clock, or Timing. The HP E2477A inverse assembler/execution tracker software can be used for State-per-ack and State-per-clock analysis. The following sections describe these operating modes and how to configure the logic analyzer for each mode.

State-per-ack mode

In State-per-ack mode, the logic analyzer uses clock store qualification to capture only address and data-acknowledge cycles. This is the default mode set up by the configuration files.

State-per-ack mode provides the greatest information density in the logic analyzer acquisition memory.

State-per-clock mode

In State-per-clock mode, every clock cycle is captured by the logic analyzer, including idle and wait states between and during tenures. To configure the logic analyzer for State-per-clock mode, use the Format menu to change the store qualification to “anystate”, and change the clock qual Q1 to Off. For additional information, refer to the “Format menu” and “To qualify stored data” sections.

Timing mode

In Timing mode, the logic analyzer samples the microprocessor pins asynchronously, typically with 4-ns resolution. To configure the logic analyzer for timing analysis, select the Configuration menu of the logic analyzer, select the Type field for Analyzer 1, and select Timing.

Modes of Analysis

The inverse assembler offers two modes of analysis for MPC860/821 microprocessors: inverse assembly, and cache-on execution tracking. These two modes cannot operate at the same time. To change from one analysis mode to the other, load the appropriate configuration file.

If you have acquired data with the cache-on execution tracker configuration, you can view the data in a Listing window with the cache-on execution tracker or with the inverse assembler.

If you have acquired data with the inverse assembler configuration, you can view the data in a Listing window with the inverse assembler only.

Inverse assembly analysis

The inverse assembler lets you obtain displays of MPC860/821 operations in MPC860/821 mnemonics. Information that is processed in cache is not visible to the inverse assembler, and cannot be decoded.

Cache-on execution tracker

The cache-on execution tracker lets you track instructions executed in the cache. However, the data is not inverse assembled into MPC860/821 mnemonics. The logic analyzer displays the data in instruction-type format.

The cache-on execution tracker requires the HP 16600A/700A-series logic analyzers, or the HP 16505A prototype analyzer for the HP 16500B/C mainframe. It provides much more information when used together with the HP B4620B Source Correlation Tool Set. Source correlation performs a correlation of the addresses from cache with the high-level code execution.

For cache-on execution tracking, set the operating mode to State-per-clock.

Logic Analyzer Configuration

The following sections describe the logic analyzer configuration as set up by the configuration files.

Trigger specification

The trigger specification is set up by the software to store all states. If you modify the trigger specification to store only selected bus cycles, incorrect or incomplete disassembly may be displayed.

Format menu

This section describes the organization of MPC860/821 signals in the logic analyzer's Format menu.

The configuration software sets up the analyzer format menu to display either six, eight, or twelve pods of data, depending on the analyzer. The figure on the following page **shows the Format menu for the MPC860/821.**

The configuration files contain predefined format specifications. These format specifications include all labels for monitoring the microprocessor. The tables on the following pages show the signals used in the STAT label and the predefined symbols set up by the configuration files.

The HP logic analyzers and the PowerPC use opposite conventions to designate individual signals on a bus. In PowerPC nomenclature, bit 0 is the most significant; in the logic analyzers, bit 0 is the least significant. In PowerPC, A0 is the most significant bit of the address bus; on the analyzer, this bit is called ADDR31.

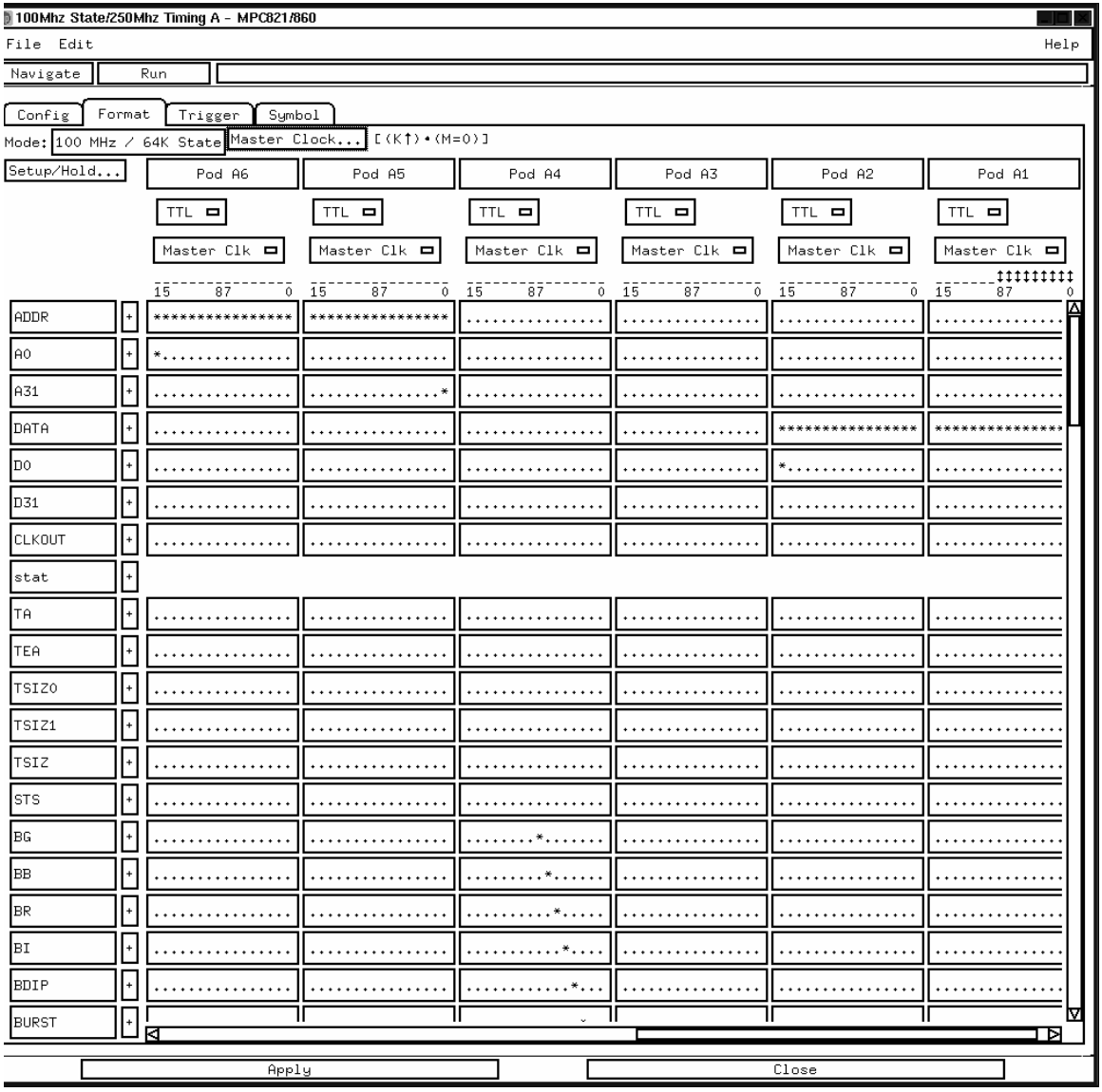
Most Significant	Least Significant	
A0	**A31**	***PowerPC***
ADDR31	**ADDR0**	***Logic Analyzer***

This may cause confusion in the waveform window when using Channel Mode Sequential or Individual.

Format Menu

Do not modify the ADDR, DATA, or STAT labels in the format specification if you want inverse assembly. Changes to these labels may cause incorrect or incomplete inverse assembly.

Format Window



Status Encoding

Each of the bits of the STAT label is described in the table below. The inverse assembler uses STAT bits BR, BG, BB, TS, TA, TEA, R/W, BURST, BI, and TSIZ. The signal-to-connector tables in the "Hardware Reference" chapter list all the MPC860/821 signals probed and their corresponding analyzer channels.

Status Bit Description

Status Bit	Description
BR	A slave controller asserts Bus Request to indicate that it wants access to the memory bus.
BG	The memory system asserts Bus Grant to allow the MPC860/821 onto the address bus.
BB	Bus Busy indicates that the an external master controller has the bus.
TS	The MPC860/821 asserts TS for one cycle to commence a transaction.
TA	The memory system asserts TA to acknowledge a data transaction.
TEA	The memory system may assert TEA to indicate a transfer error, such as an unmapped part of the address space.
R/W	R/W is high for a read, low for a write.
STS	Special Transfer Start is an internal transfer indicator.
BURST	BURST indicates that the current initiated transfer is a burst one. The BURST signal is negated prior to the expected last data beat of the burst transfer.
BDIP	Burst Data In Progress indicates the last beat of a burst transfer.
BI	Burst Inhibit indicates that the slave device addressed in the current burst transaction is unable to support burst transfers.
AT[0, 2, 3]	Address Type provides further information about the current transfer. For a read, they indicate whether instructions or operands are being fetched.
TSIZ [0:1]	Indicates the size for the data transfer.
VF[0:2]	Visible Instruction Queue Flushes Status tracks program flow.
VFLS[0:1]	Visible History Buffer Flushes Status tracks program flow.

Predefined Logic Analyzer Symbols

The configuration software sets up symbol tables on the logic analyzer. The tables define a number of symbols which make several of the STAT fields easier to interpret. The following table lists the symbol descriptions.

Symbol Description

Label	Symbol	Encoding
R/W	rd	1
	wr	0
TSIZ	4 byte	00
	1 byte	01
	2 byte	10
TEA	(blank)	1
	tea	0
TA	(blank)	1
	ta	0
BURST	(blank)	1
	burst	0
BI	(blank)	1
	bi	0
BR	(blank)	1
	br	0
BG	(blank)	1
	bg	0
VFLS	0 flsh	00
	1 flsh	01
	2 flsh	10
	debug	11

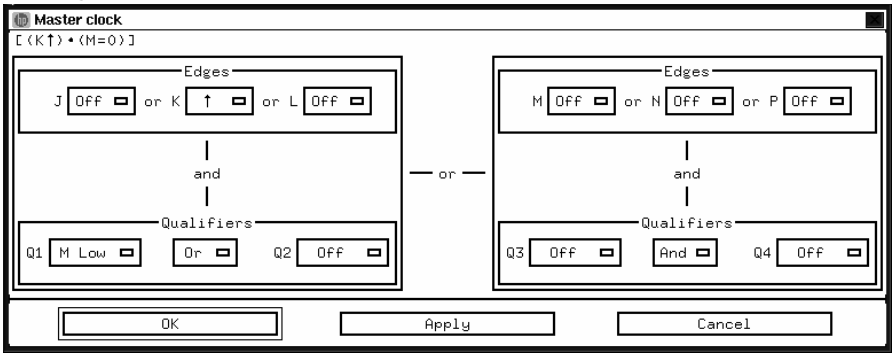
Label	Symbol	Encoding
BB	(blank)	1
	bb	0
BDIP	(blank)	1
	bdip	0
STS	(blank)	1
	sts	0
AT0	CPM	1
	CPU	0
AT2	data	1
	instr	0
AT3	trace	1
	resrv	0
TS	(blank)	1
	ts	0
VF	000	000
(see note)	010	001
	100	010
	110	011
	001	100
	011	101
	101	110
	111	111

Note: The VF symbols display the proper order of these pins. These symbols compensate for the MPC860/821 multiplexing scheme.

To qualify stored data

Store qualification is done by clock qualification. The M clock is the signal TA. The clocking combination “K rising and M low (TA asserted)” stores TA states. This setup latches address and data when the system clock is rising and TA is asserted.

Storage Qualification



The memory controller in the MPC860/821 will allow data to be latched on the falling edge of the system clock (K falling). If your data appears incorrect, check the memory controller registers for this type of configuration.

For State-per-clock acquisition, change the clock qual Q1 to Off.

Burst Mode

When transferring data in burst mode, the memory controller drives many of the bus signals. Since the memory controller can be configured to latch address and data in several different ways, the clocks may need to be configured differently for your setup. A slave/master clock setup can be used to latch either data or address early.

Using the Inverse Assembler

This section discusses the general output format of the inverse assembler and processor-specific information.

Before the inverse assembler will correctly disassemble information captured on the MPC860/821 address bus, you must:

- If you are not using the cache-on execution tracker, make sure the target system's cache has been disabled.
- Set up the inverse assembler preferences to give the inverse assembler information about the MPC860/821 memory controller and memory bank setup.

Inverse Assembly and Cache-on Execution Tracking
The inverse assembler is loaded with configuration files CM860Mxx and CM860Fxx. The cache-on execution tracker is loaded with configuration files CM860ETx. To change from inverse assembly to cache-on execution tracking, you must reload the appropriate configuration file into the logic analyzer.

To disable the instruction cache on the MPC860/821

When the instruction cache is enabled, many PowerPC instructions are executed from the cache and do not appear on the external bus. To get an execution trace on the bus, the instruction cache can be disabled. This must be done in supervisor mode.

To disable the cache with the emulation module:

Use your debugger or the Emulation Control Interface to set the IC_CST register (or the DC_CST register, to disable the data cache).

Register values for controlling the cache

Value	Meaning
0400 0000	Disable
0a00 0000	Unlock all
0c00 0000	Invalidate
0200 0000	Enable

To disable the cache with code:

- Disable the cache with the following code:

```
addir2, 0, 0X01A
 addir1, 0, 0X01
 slwr0, r1, r2
 mtsprICCST, r0; disable Instr Cache
 isync
```

- To also disable the data cache use:

```
addir2, 0, 0X01A
 addir1, 0, 0X01
 slwr0, r1, r2
 mtsprDCCST, r0; disable Data Cache
 isync
```

- To invalidate and disable the caches use:

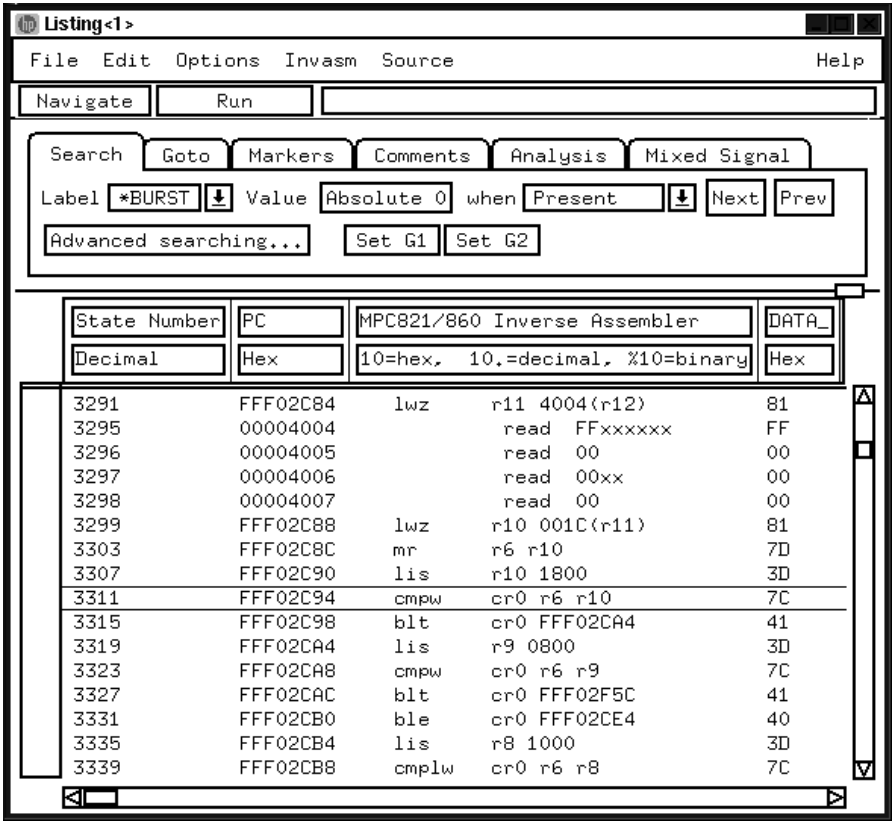
```
addir2, 0, 0X019
 addir1, 0, 0X06
 slwr0, r1, r2
 mtsprICCST, r0; invalidate instr cache
```

```
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprICCST, r0; disable Instr Cache
addir2, 0, 0X019
addir1, 0, 0X06
slwr0, r1, r2
mtsprDCCST, r0; invalidate Data cache
addir2, 0, 0X01A
addir1, 0, 0X01
slwr0, r1, r2
mtsprDCCST, r0; disable Data Cache
isync
```

To display captured state data

The logic analyzer displays captured state data in the Listing menu. The inverse assembler display is obtained by setting the base for the DATA label to Invasm. The following figure shows a typical Listing menu.

Listing Menu



The column on the left of the inverse assembly data display is the least significant hexadecimal digit of the current address. Because the MPC860/821 presents one address then reads 16 bytes during a burst, the least-significant digit is synthesized by the disassembler. On the HP 16600A/700A-series logic analysis systems, the entire synthesized address appears under the label “PC”. You can observe the actual address bits presented by the MPC860/821 under the ADDR label.

Inverse assembler output format

The following paragraphs explain the operation of the inverse assembler and the results you can expect under certain conditions.

Interpreting Data

General purpose registers are displayed as r0, r1...r31. Special purpose registers are displayed using their mnemonic.

Most numerical data is displayed in hexadecimal, for example, “lwzr28 0044(r1).”

Bit numbers and shift counts are displayed in decimal with a dot suffix, for example, “cror 31. 31. 31.”

A few instructions display their operands in binary with a “%” prefix, for example, “mtfsfi 4 %0101.”

The inverse assembler decodes the full PowerPC instruction set architecture, including 64-bit mode instructions and optional instructions not implemented on the MPC860/821. When these unimplemented opcodes are encountered, the instruction mnemonic has a “?” prefix. If a reserved bit is set in an instruction opcode field, a “?” is appended most often to the mnemonic, but in some cases to an operand.

An instruction word of 00000000 is decoded as “illegal.” Otherwise, if an opcode is invalid, it is shown as “Undefined Opcode.”

Branch Instructions

If the address of a branch relative instruction is known, its target is presented as an absolute hex address (or as a symbol if it matches an ADDR pattern or range symbol). If the address of a branch relative instruction is not known, its target is displayed as a hexadecimal offset such as +00000C30 or -00000048.

Extended Mnemonics

PowerPC assemblers support a number of extended mnemonics for some popular assembly language instructions as described in the MPC860/821 User's Manual. The E2476/7A disassembler supports the following extensions:

- Conditional traps and branches decode the condition mnemonically when possible. For some conditions which have no conventional mnemonics (for example, “signed less than or unsigned greater than”), the condition field

is displayed in binary.

- The L bit is omitted as a compare operand. Instead, compares are decoded as “cmpw” (or “?cmpd”).
- “Add immediate” instructions with a negative immediate operand are decoded as subtract immediate (“subi”).
- “Subtract from” instructions subf and subfc are decoded as subtract instructions sub and subc with the operands exchanged so that “sub r3 r4 r5” is mnemonically interpreted as “r3 = r4 - r5.”
- ori r0 r0 0000 is decoded as “nop”.
- Add immediate and add immediate shifted instructions, addi and addis, with a null source register are decoded as load immediate and load immediate shifted, li and lis.
- or instructions with identical source registers are decoded as move register, mr.
- nor instructions with identical source registers are decoded as not register, not.
- xor and eqv instructions with identical source and destination registers are decoded as clear and set, clr and set, respectively.
- The cror, crnor, crxor, and creqv instructions map analogously to crmv, crnot, crclr, and crset.
- When the mtcrf instruction field mask specifies the entire cr, it is decoded as mtcr.

- The PowerPC rotate-left instructions have extended mnemonics. The following listing shows the extended mnemonics for the integer rotate instructions.

Mnemonic	Decoded As
rlwimi (rotate left word immediate then mask insert)	inslwi insert from left immediate insrwi insert from right immediate
rlwinm (rotate left word immediate then AND with mask)	rotlwirotate left immediate rotrwirotate right immediate slwishift left immediate srwishift right immediate extlwiextract and left justify immediate extrwiextract and right justify immediate clrlwiclear left immediate clrrwiclear right immediate clrlslwiclear left and shift left immediate
rlwnm (rotate left word then AND with mask)	rotlwrotate left

PC Label

When an HP 16600A/700A-series logic analysis system is being used, the inverse assembler generates a “PC” field. The PC is displayed as another column in the listing tool. This field is the Software Address generated by the inverse assembler.

The PC label cannot be used exactly like other labels. For example, when loading symbols, you will notice that the PC label is not in the list of labels that the symbols can be loaded into. Symbols should still be loaded into the ADDR label. The main purpose of the PC label is for correlation of the listing with source code using the HP B4620B Source Correlation Tool Set.

To use the Invasm menu

The Invasm menu provides four choices: Load, Filter, Preferences, and Options. Access the Invasm menu in the listing window.

You must use the Preferences dialog to configure the inverse assembler to match the microprocessor memory controller configuration. The other dialogs assist in analyzing and displaying data. The following sections describe these dialogs.

Load

The Load dialog lets you load a different inverse assembler and apply it to the data in the Listing menu. In some cases you may have acquired raw data; you can use the Load dialog to apply an inverse assembler to that data.

Display Filtering

The inverse assembler lets you Show or Suppress several types of states. This dialog is called display filtering. States can be filtered according to what type of cycle the state is, or according to which memory bank was accessed for the cycle.

The show/suppress settings do not affect the data that is stored by the logic analyzer; they only affect whether that data is displayed or not. You can examine the same data with different settings, for different analysis requirements.

This dialog allows faster analysis in two ways. First, you can filter unneeded information out of the display. For example, suppressing idle states will show only states in which a transaction was completed.

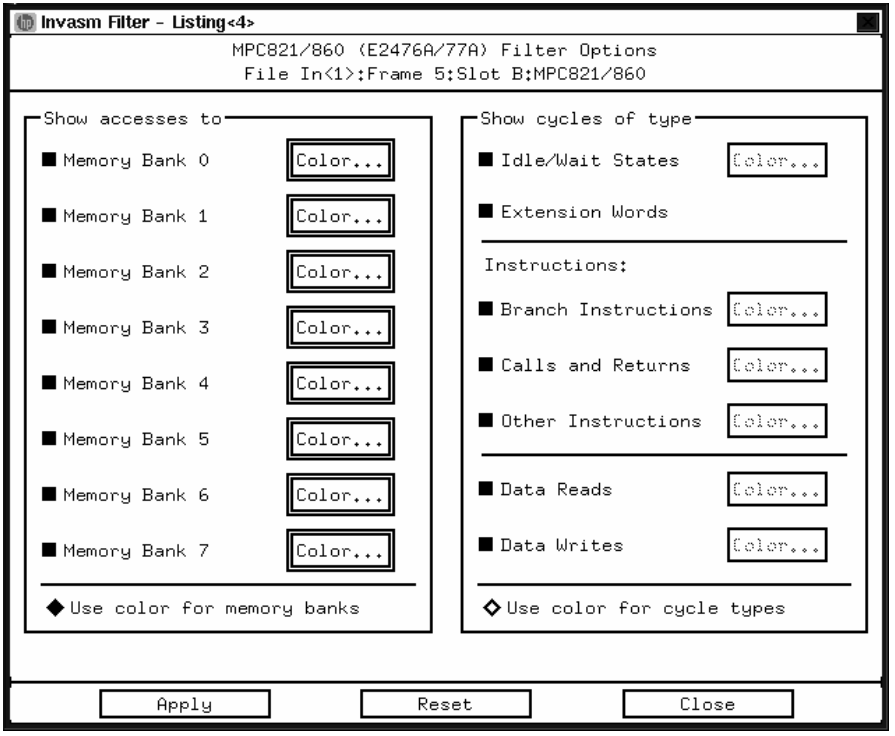
Second, you can isolate particular operations by suppressing all other operations. For example, you can show Calls and Returns, with all other states suppressed, allowing quick analysis of Calls and Returns.

HP 16600A/700A-series logic analysis systems provide one additional feature for analyzing data. Instead of (or in addition to) showing or suppressing states, the selected states can also be shown in color.

Color can only be used for distinguishing either memory bank accesses or cycle types, but not both at the same time.

The following figure shows the inverse assembler Filter menu.

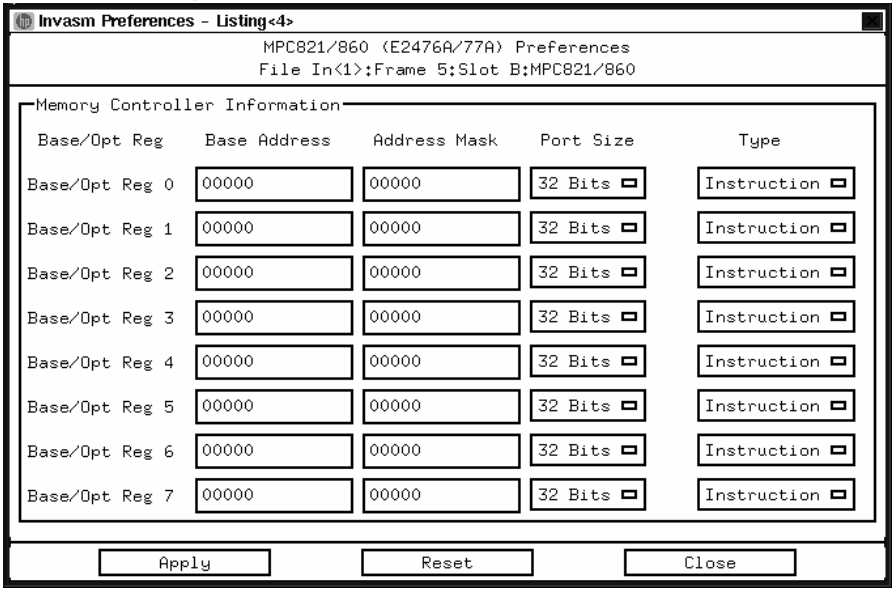
Inverse Assembler Filter Menu



Preferences

The MPC860/821 does not always provide a signal to distinguish instruction reads from data reads. Also, the width of the memory being accessed (port size) is not indicated on external signals. For accurate inverse assembly the inverse assembler must be configured to match the memory controller configuration on the microprocessor. The following figure shows the inverse assembler Preferences window, where the inverse assembler is configured.

Inverse Assembly Preferences Menu



The changes will apply to the top line of the Listing menu screen and down.

Base Address = Upper 17 bits of each base register.
Address Mask = Upper 17 bits of each option register.
Portsize = Width of memory being accessed, and is encoded in bits 20 and 21 of the Base Register.
Type =Information Type that is located in this range. If address type bits are used then AT2 indicates instruction or data. If address type bits are not used, the AT2 does not necessarily indicate instruction or data. In either case, the inverse assembler needs to know instruction or data.

The inverse assembler assumes all memory banks are valid, so lower-numbered banks should be used before higher-numbered banks. Bank 0 has the highest priority, and Bank 7 has the lowest priority.

If the inverse assembler returns "IA Error: Address not in table" then the address did not meet the specifications for any of the memory banks.

Why the configuration is necessary

Because critical information about what type of data is being accessed through a memory bank is stored in internal registers, the inverse assembler needs to be given some information about how the memory system is set up.

The memory controller operates by mapping every address to one of eight memory banks. Each memory bank can be set up to drive different external signals, to have different write permissions, etc. The memory banks are numbered from 0 to 7. Memory bank 0 has the highest priority and bank 7 has the lowest.

The base register and option register for each memory bank hold information that describes the width of the memory accessed through that bank, the type of data, and the addresses that will be accessed through that bank. Since this information is not given on external signals, the inverse assembler provides a preferences window to enter this information so that the data decode can be as accurate as possible.

Finding memory bank information using a debugger

You can use a debugger to examine the base register and option register to determine what values to enter in the preferences window.

The Base Address fields should be set to match the upper 17 bits of the base registers.

The Address Mask fields should be set to match the upper 17 bits of the option registers.

The memory port sizes can be determined by looking at bits 20-21 (assuming bit 31 is least significant) of the base registers.

The address type bits can be used to limit access to instructions or data. Your target may or may not be configured to use address type comparisons. Look at bit 19 of the option registers (this is the address type mask), if this bit is cleared (0) then your target is not configured to do address type comparisons.

If the bit is set then bit 19 of the base register will be set to 0 if the memory bank accesses instructions and a 1 if the memory bank accesses data.

Finding memory bank information at compile time

When compiling the code that will be analyzed, direct the linker to locate all instructions and all data (constants, variables, the stack) in separate memory 'blocks'. 'Blocks' of memory can be differentiated as long as one of the upper 17 address bits differ. Then, set up the preferences window so that memory bank 0 will decode the instruction states and memory bank 1 will decode the data states.

Example

For example, assume the code will run from DRAM that has a 32 bit port size. Compile the code specifying to the linker to place text or code at address 0x00020000 and data at address 0x00100000.

Set up the first two memory banks in the preferences window as follows:

Bank	Base Address	Address Mask	Port Size	Read Type
0	00020	FFFF0	32 bit	instruction
1	00100	FFF00	32 bit	data

The inverse assembler will now interpret any read from addresses 0x00020000 - 0x0002FFFF as instruction reads and any reads from 0x00100000 - 0x001FFFFF as data reads.

Options

The Options menu lets you change the width of the display.

Using the Cache-on Execution Tracker

This section discusses the general output format of the cache-on execution tracker and processor-specific information.

Inverse Assembly and Cache-on Execution Tracking
The cache-on execution tracker is loaded with configuration files C860ETx. The inverse assembler is loaded with configuration files C860Mxx and C860Fxx. To change from inverse assembly to cache-on execution tracking, you must reload the appropriate configuration file into the logic analyzer.

System Requirements

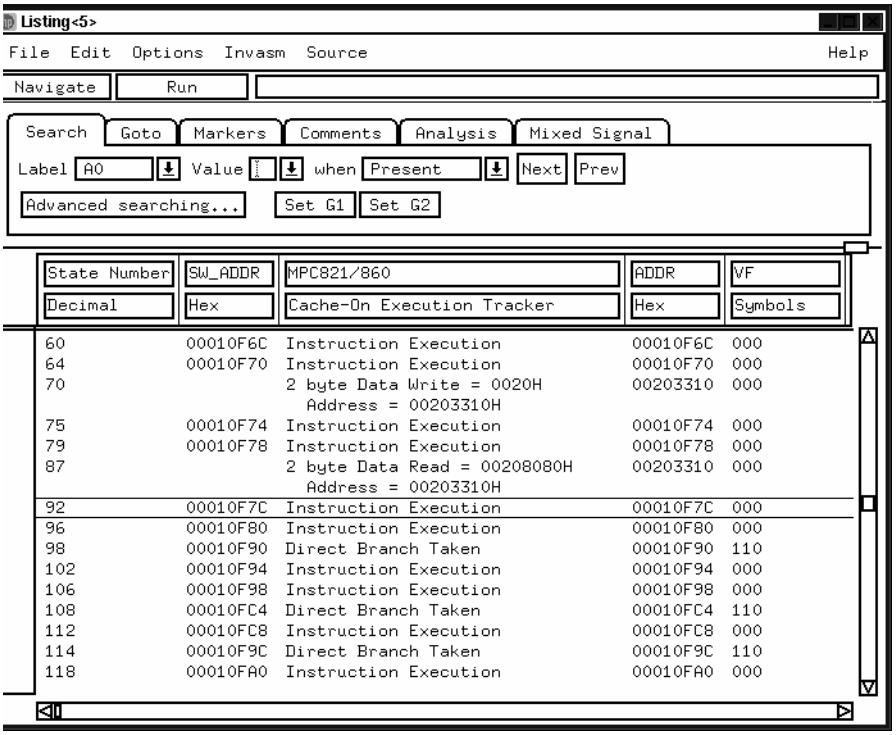
The cache-on execution tracker requires the HP 16600A series or HP 16700A-series logic analyzers, or the HP 16505A Prototype Analyzer with the HP 16500B/C mainframe. Although not required, the HP B4620B Source Correlation Tool Set is highly recommended. See pages page 27 and page 29 for the logic analyzers and logic analyzer software version requirements for using the cache-on execution tracker.

The analysis probe operating mode must be set to State-per-clock for the execution tracker to operate properly. Program trace cycles must be enabled, and the OP2/MODCK1/$\overline{\text{STS}}$ pin must be configured for STS.

To display captured state data

The logic analyzer displays captured state data in the Listing menu. The cache-on execution tracker display is obtained by setting the base for the DATA label to Invasm. The following figure shows a typical Listing menu.

State Data for Cache-on Execution Tracker

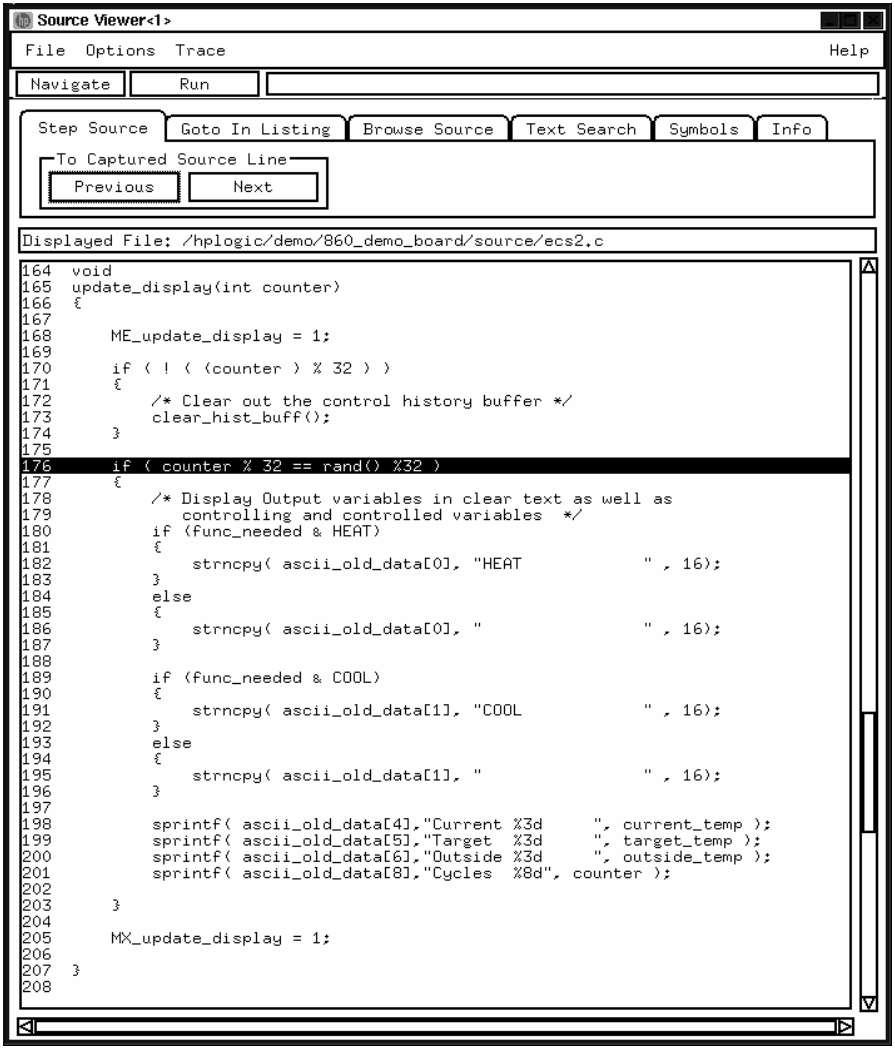


The entire synthesized address appears under the label “SW_ADDR”. The actual address bits presented by the MPC860/821 may be observed under the ADDR label.

Displaying Data with the HP B4620B Source Correlation Tool Set

Source correlation correlates the addresses from cache with the high-level code execution. The figure below shows execution of data that is correlated to the data shown on the previous page.

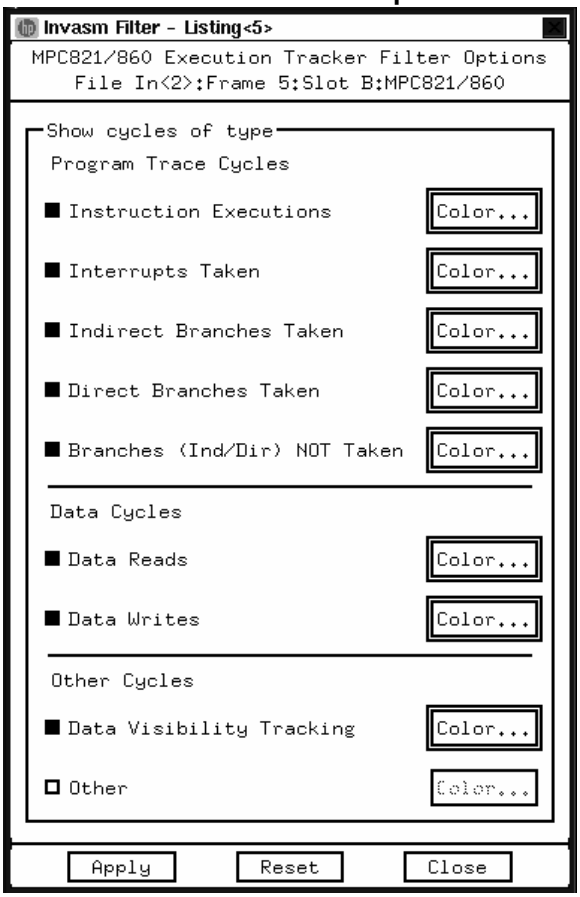
Source Correlation Tool Set Data



Display Filtering

The cache-on execution tracker lets you Show or Suppress several types of cycles. This dialog is called display filtering. States can be filtered according to the type of cycle. The figure below shows the menu.

Cache-on Execution Tracker Filter Options



The show/suppress settings do not affect the data that is stored by the logic analyzer; they only affect whether that data is displayed or not. You can examine the same data with different settings, for different analysis requirements.

This dialog allows faster analysis in two ways. First, you can filter unneeded information out of the display. For example, suppressing “Other” states will show only states that contain data information or code execution information.

Second, you can isolate particular operations by suppressing all other operations. For example, you can show Branches Taken, with all other states suppressed, allowing quick analysis of branches.

The cycles that can be shown or suppressed are described below.

Instruction Executions. This dialog lets you show or suppress sequential instruction executions.

Interrupts Taken . This dialog lets you show or suppress transition states that are the result of an interrupt or an exception. The cache-on execution tracker will also attempt to decode the type of interrupt based on the interrupt vector address. If Interrupts Taken is suppressed, the interrupt code executed will still be displayed.

Indirect Branches Taken. This dialog lets you show or suppress any branch to an address obtained from the MPC860/821 link or count registers.

Direct Branches Taken . This dialog lets you show or suppress any branch that is not indirect (such as to a specified address).

Branches (Ind/Dir) NOT Taken. This dialog lets you show or suppress any conditional branches NOT taken.

Data Reads. This dialog lets you show or suppress data reads.

Data Writes. This dialog lets you show or suppress data writes.

Other. This dialog lets you show or suppress all other states not covered by the above program trace cycle filters.

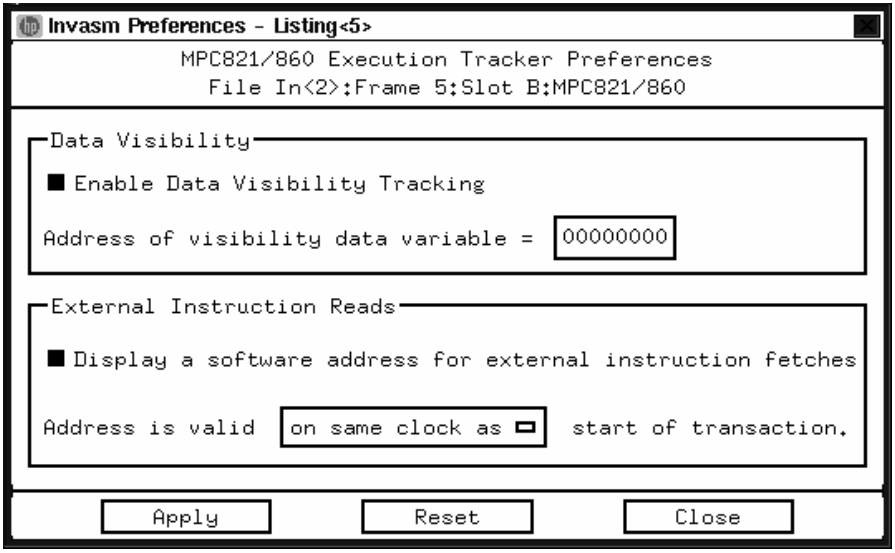
Color

This dialog provides one additional feature for analyzing data. Instead of (or in addition to) showing or suppressing states, the selected states can also be shown in color, for quick visual identification.

Preferences

The Preferences pop-up menu provides two choices: Data Visibility Tracking (for 32-bit memory systems only), and Display Software Address.

Execution Tracker Preferences Menu



Enable Data Visibility Tracking. This field enables the Data Visibility Tracking function. Data Visibility Tracking allows you to see the data values during debug with the data cache enabled. To use this function, you must insert data visibility macros into your code, and specify a starting address for a non-cachable memory location in which the data will be stored. The Data Visibility macros and an application note can be obtained at ftp site hpcsos.col.hp.com. To access this site, log on as anonymous, and use your email address as your password. The files are located in dist/logic/data_vis/MPC860/.

Data visibility tracking should be used only with systems which have a 32-bit bus.

Display Software Address . Enabling this dialog allows correlation of instruction fetches external to the cache with the source code. When this dialog is enabled, you must specify the number of wait states (clocks) before the address is valid. The number of wait states is typically dependent on the type of memory hardware being accessed.

To enable Program Trace cycles

For the cache-on execution tracker to operate properly, program trace cycles must be enabled, and the OP2/MODCK1/$\overline{\text{STS}}$ pin must be configured for $\overline{\text{STS}}$. The procedures below perform these operations. This must be done in supervisor mode.

- Enable the Program Trace cycles with the following code:

```
addir1, 0, 0
 mtsprICTRL, r1
```

- To enable the STS pin, verify that bits 9 and 10 of the SIUMCR are set to 01 or 11.

7

Symbols and Source Code in the Analyzer

Symbols are more easily recognized than hexadecimal address values in logic analyzer trace displays, and they are easier to remember when setting up triggers.

HP logic analyzers let you assign user-defined symbol names to particular label values.

Also, you can download symbols from certain object file formats into HP logic analyzers.

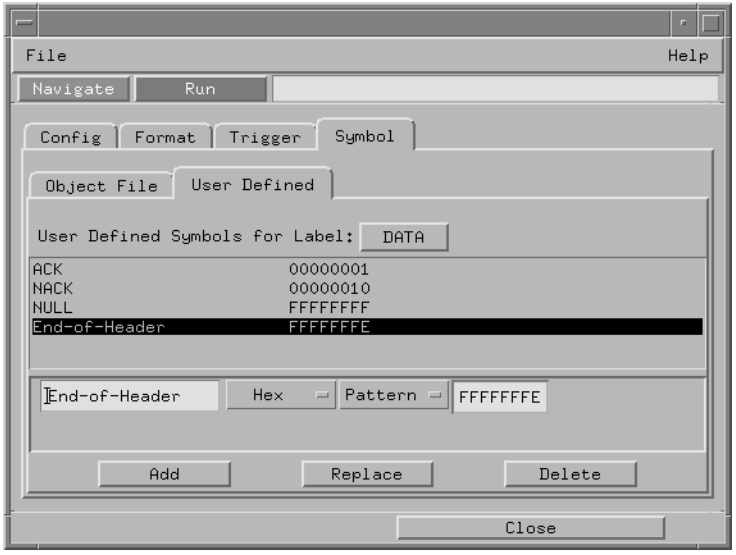
When source file line number symbols are downloaded to the logic analyzer, you can set up triggers on source lines. The HP B4620B Source Correlation Tool Set also lets you display the high-level source code associated with captured data.

After describing user-defined symbols, the rest of this chapter describes the requirements and considerations for displaying object file symbols and source code for MPC860/821 address values captured by a logic analyzer.

User-Defined Symbols

User-defined symbols are symbols you create from within the logic analyzer user interface by assigning symbol names to label values. Typically, you assign symbol names to address label values, but you can define symbols for data, status, or other label values as well.

User-defined symbols are saved with the logic analyzer configuration.



User-Defined Symbols Dialog

Predefined MPC860/821 Symbols

If you are using an analysis probe for the MPC860/821 microprocessor, the logic analyzer configuration files include predefined symbols.

These symbols appear along with the other user-defined symbols in the logic analyzer.

The predefined MPC860/821 symbols are listed on page 125.

Object File Symbols

The most common way to load program symbols into the logic analyzer is from an object file that is created when the program is compiled.

Requirements

In order for object file symbols and source code to be accurately assigned to address values captured by the logic analyzer, you need:

An accurate bus trace

Typically, an HP analysis probe is used to capture MPC860/821 microprocessor data. However, it is also possible to design connections for the logic analyzer into a prototype target system. Refer to the previous chapters on analysis probes and designing connections for custom probing.

An inverse assembler

Typically, the inverse assembler software is included with HP analysis probes, but it can also be purchased separately when custom probing connections are designed into a prototype target system. The MPC860/821 inverse assembler decodes captured data into program counter (PC) addresses (also known as software addresses) and assembly language mnemonics. Refer to the previous chapter on MPC860/821 inverse assembly.

A symbol file

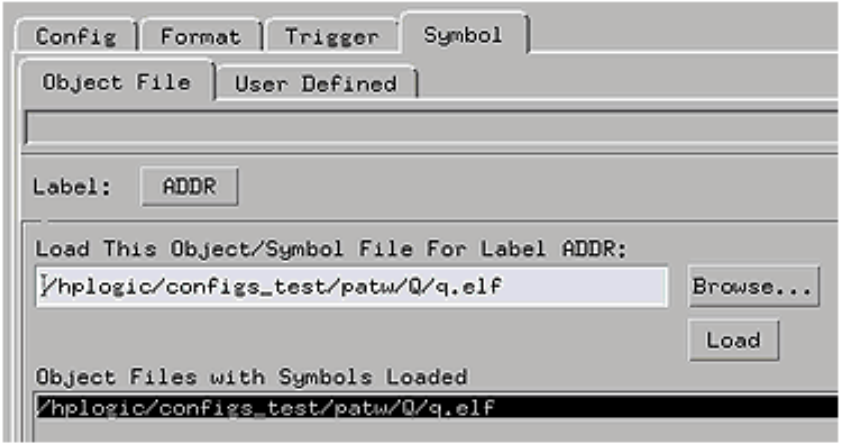
You need an object file containing symbolic debug information in a format the logic analyzer understands. Alternatively, you can use a General Purpose ASCII (GPA) symbol file (see page 267).

See Also

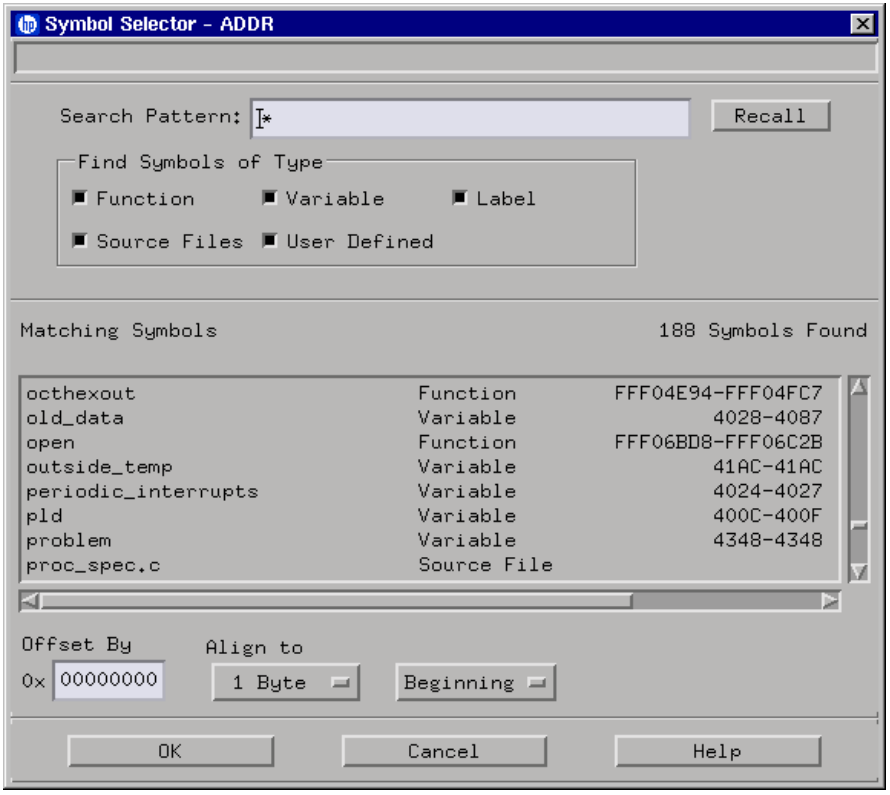
This chapter does not give you task-based instructions for loading object file symbols into a logic analyzer. Refer to your logic analyzer documentation or online help for these instructions.

To use object file symbols in the HP 16600A/ 700A

To load symbols in the HP 16600A/16700A-series logic analysis system, open the logic analyzer module's Setup window and select the Symbol tab; then, select the Object File tab. Make sure the label is ADDR. From this dialog you can select object files and load their symbol information.



When you load object file symbols into a logic analyzer, a database of symbol/line number to address assignments is generated from the object file. The Symbol Selector dialog allows you to use a symbol in place of a hexadecimal value when defining trigger patterns, trigger ranges, and so on.



If your language tool is not one of those listed on page 153, you can create a symbol file in the General-Purpose ASCII (GPA) file format (refer to the "General-Purpose ASCII (GPA) File Format" chapter).

See Also

Refer to your logic analyzer documentation or online help for information on how to load symbol files.

Compilers for MPC860/821

The following MPC821/860 compilers and their ELF/DWARF format object files can be used with HP logic analyzers and the HP B4620B Source Correlation Tool Set:

Object File Formats

Language System & Version	Format
Diab Data version 4.1a	ELF/DWARF
Green Hills version 1.8.8	ELF/DWARF
Microtec Research, Inc. version 1.4	ELF/DWARF

In order to use symbols in the logic analyzer, file name and line number information must be present in the object file. Your compiler may have options that include or exclude this information.

Limitations: For C++ files, symbols are not demangled. Mangled names are available for use and the trace listing will still correctly correlate to the appropriate source file lines.

When compiling code, if possible, specify that code and data be put in different memory “blocks”. A “block” is 32 Kbytes. 32 Kbytes is the smallest area of memory that can be distinguished by each memory block.

It is also useful to put the stack in the data block.

By separating the code and data in this way, the inverse assembler can be configured to properly decode both code and data.

See Also

Contact your Hewlett-Packard sales engineer to find out if there are other compilers for the MPC860/821 microprocessor that can be used with HP logic analyzers.

Diab Data Compiler Options

The following options should be used:

-g	Specifies to generate symbolic debugger information (same as -g2).
-WDDOBJECT=E	Specifies the ELF/DWARF file format.
-WDDENVIRON=cross	Specifies the cross development environment.
-WDDTARGET=PPC860 (or -WDDTARGET=PPC821)	Specifies the type of processor.
-Xdebug-mode=0xff	Turns off Diab Data extensions to the file format.

Diab Data provides a utility that you can use to generate the compiler options you need. Enter "dctrl -t" and follow the instructions. When it is finished, it will present you with a string that you can use for the compiler options.

Please refer to the language tool supplier's documentation for more information about the options available.

More information is available on the World Wide Web at: **http://www.diabdata.com**

Green Hills Compiler Options

The following options should be used:

-dwarf	Generates DWARF debugging information.
-G	Generates extended debugging information.
-cpu=ppc860 (or -cpu=ppc821)	Specifies code generation for the PPC860 (or PPC821) processor.

If you are using the Green Hills MULTI builder interface, use the following selections:

Options→Advanced, enable "Output DWARF on ELF targets"	Generates DWARF debugging information.
Options→File Options, select "Debugging Level MULTI"	Generates extended debugging information.
Options→CPU, select processor	Specifies code generation for the PPC860 (or PPC821) processor.

Please refer to the language tool supplier's documentation for more information about the options available. More information is available on the World Wide Web at: **http://www.ghs.com**

Microtec Research Inc. Compiler Options

The following options should be used:

-g	Specifies to generate debugging information.
-p860 (or -p821)	Specifies code generation for the PPC860 (or PPC821) processor.

Please refer to the language tool supplier's documentation for more information about the options available.

More information is available on the World Wide Web at: **http://www.mentorg.com/microtec**

Source Code

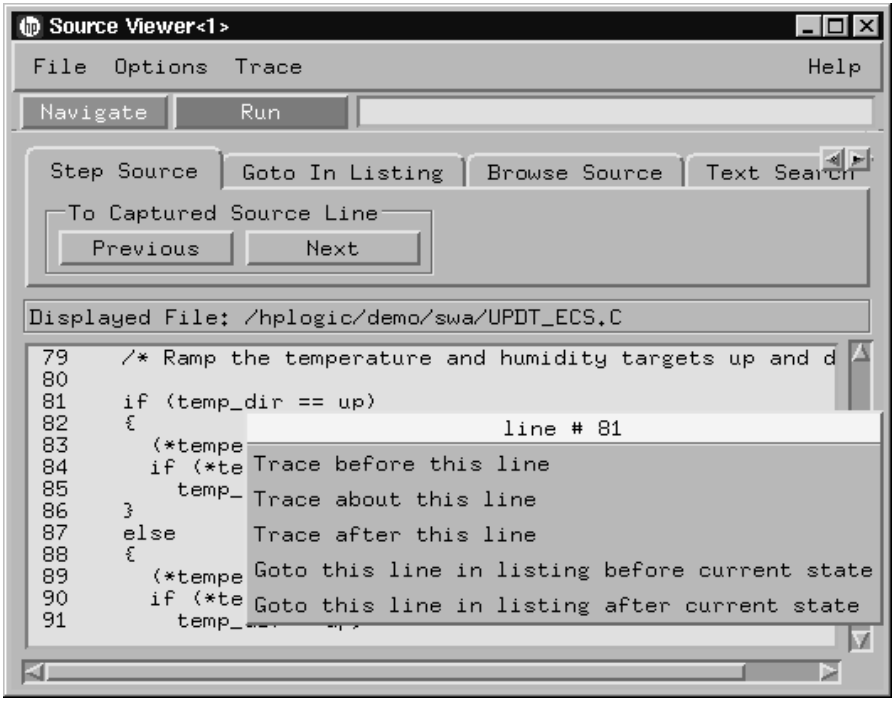
The HP B4620B Source Correlation Tool Set lets you:

- View the high-level source code associated with captured data.
- Set up triggers based on source code.

The source correlation tool set correlates the logic analyzer's address label with a line of high-level source code whose address, symbol name, file name, and line numbers are described in a symbol file downloaded to the logic analyzer.

To display the Source Viewer window, click on the logic analyzer module icon in the System window, and choose **Source Viewer...**.

The first time you display the Source Viewer window, it will probably be blank. To see the source code click the Browse Source tab and choose a file to display. To see source code which corresponds to a particular state in the listing, select that state in the Listing window.



If you purchased a solution, the HP B4620B Source Correlation Tool Set was included. Otherwise, the source correlation tool set is available as an add-on product for the HP 16600A/16700A-series logic analysis system and must be licensed before you can use it (see the System Admin dialogs for information on licensing).

See Also

More information on configuring and using the source correlation tool set can be found in the online help for your logic analysis system.

Requirements for source correlation

The source correlation tool set works with many microprocessors and their embedded software development environments.

However, the overall effectiveness of the source correlation tool set will vary to some degree depending on the specific development environment it is being used in. The following areas affect the performance of the source correlation tool set for different development environments:

- Analysis probe and inverse assembler.

  All the information needed to reconstruct the complete address bus of the target system must be acquired by the logic analyzer. The HP E2476A analysis probe meets this requirement.

  The logic analyzer's inverse assembler may need to reconstruct any incomplete address bus information and/or filter out any unexecuted instructions.

  When displaying the next or previous instances of a source line, the Source Viewer display uses the PC or SW_ADDR (Software Address) label generated by the inverse assembler.

- Object file symbols.

  The source correlation tool set requires that symbols be loaded into the logic analyzer (refer to the "Object File Symbols" section earlier in this chapter).

  The compiler needs to produce an object file format that is readable by the logic analyzer; otherwise; a general-purpose ASCII (GPA) format file needs to be generated.

- Access to source code files.

  The source correlation tool set requires that you give the logic analysis system access to your program's high-level source files (either by NFS mounting the file system that contains the source files or by copying source files to the logic analysis system disk).

Inverse Assembler Generated PC (Software Address) Label

In the HP 16600A/16700A-series logic analysis system, the MPC860/821 inverse assembler generates a “PC” label. The PC label is displayed as another column in the Listing tool. This label is also known as the Software Address generated by the inverse assembler.

The “Goto this line in listing” commands in the HP 16600A/16700A-series logic analysis system perform a pattern search on the PC label in the Listing display (when an inverse assembler is loaded). Because the inverse assembler is called for each line that is searched, the search can be slow, especially with a deep memory logic analyzer.

Also, a single source code line will generate many assembly instructions. The “Goto this line in listing” commands will not find a given source code line unless the first assembly instruction generated from the source line has been acquired by the logic analyzer.

For example, if the compiler unrolls a loop in the source code, the trace could begin after the first assembly instruction of the loop has been executed. A “Goto this line in listing” command would not find the source line.

Access to Source Code Files

The source correlation tool set must be able to access the high-level source code files referenced by the symbol information so that these source files can be displayed next to and correlated with the logic analyzer's execution trace acquisition. This requires you to be aware of a number of issues.

Source File Search Path

Verify that the correct file search paths for the source code have been entered into the source correlation tool set. The HP B4620B Source Correlation Tool Set can often read and access the correct source code file from information contained in the symbol file, if the source code files have not been moved since they were compiled.

Network Access to Source Files

If source code files are being referenced across a network, the HP logic analyzer networking must be compatible with the user's network environment. HP logic analyzers currently support Ethernet networks running a TCP/IP protocol and support ftp, telnet, NFS client/server and X-Window client/server applications. Some PC networks may require extensions to the normal LAN protocols to support the TCP/IP protocol and/or these networking applications. Users should contact their LAN system administrators to help setup the logic analyzer on their network.

Source File Version Control

If the source code files are under a source code or version control utility, check the file names and paths carefully. These utilities can change source code file paths and file names. If these names are changed from the information contained in the symbol file, the source correlation tool set will not be able to find the proper source code file. These version control utilities usually provide an "export" command that creates a set of source code files with unmodified names. The source correlation tool set can then be given the correct path to these files.

Triggering on Symbols and Source Code

When setting up trigger specifications to capture MPC860/821 execution:

- Use the logic analyzer address offset to compensate for relocated code.
- Use the logic analyzer storage qualification to capture the software execution you're interested in and filter out library code execution (whose source file lookups can take a long time if the library source code is not available).

Using the Address Offset

You need to adjust the source correlation tool set to compensate for relocatable code segments or memory management units that produce fixed code offsets.

The logic analyzer has an address offset field to help facilitate this.

Entering the appropriate address offset will cause the source correlation tool set to reference the correct symbol information for the relocatable or offset code.

Using Storage Qualification

You should configure the logic analyzer's storage qualification capabilities to store only those cycles that correspond to software execution (non-idle, etc.).

The source correlation tool set can exhibit long responses to requests for the next source line if the current trace listing corresponds to code from a library that is not in the source code search path. Logic analyzer storage qualification can be used to avoid capturing library code routines.

8

Connecting and Configuring the Emulation Module

This chapter shows you how to connect the emulation module to the target system and how to configure the emulation module and target processor.

Overview

Here is a summary of the steps for connecting and configuring the emulation module:

1. Make sure the target system is designed to work properly with the emulation module. (page 169.)
2. Install the emulation module in your logic analysis system, if necessary. (page 171.)

   If you are connecting to an HP 16600A/700A-series logic analysis system, use the Setup Assistant to guide you through steps 3-6 (see page 23). Use this manual for additional information, if desired.
3. Connect the emulation module to your target system using the 50-pin cable and the TIM or an analysis probe. (page 176.)
4. Update the firmware of the emulation module, if necessary. (page 181.)
5. Verify communication between the emulation module and the target. (page 182.)
6. Configure the emulation module. (page 183.)
7. Test the connection between the emulation module and the target. (page 196.)
8. Connect a debugger to the emulation module, if applicable. (page 199.)

See Also

"Using the Emulation Module with a Debugger" beginning on page 199 for information on configuring the emulation module with a debugger, and for information on configuring LAN port numbers.

Using the Emulation Control Interface

The Emulation Control Interface in your HP 16600A/700A-series logic analysis system allows you to control an emulator (an emulation module or an emulation probe).

As you set up the emulation module, you will use the Emulation Control Interface to:

- Update firmware (which reloads or changes the processor-specific personality of the emulator).
- Change the LAN port assignment (rarely necessary).
- Run performance verification tests on the emulator.

The Emulation Control Interface allows you to:

- Run, break, reset, and step the target processor.
- Set and clear breakpoints.
- Read and write registers.
- Read and write memory.
- Read and write I/O memory.
- View memory in mnemonic form.
- Read and write the emulator configuration.
- Download programs (in Motorola S-Record or Intel Hex format) to the target system RAM or ROM.
- View emulator status and errors.
- Write and play back emulator command script files.

If you have an emulation probe, this interface also allows you to configure the LAN address of the emulation probe.

Using the logic analysis system's intermodule bus does not require the Emulation Control Interface to be running. If the emulation module icon is in the Intermodule window, then it will be able to send and receive signals. Therefore if you are using a debugger, you can use an analyzer to cause a break.

Using a debugger with the Emulation Control Interface is not recommended because:

- The interfaces can get out of synchronization when commands are issued from both interfaces. This causes windows to be out-of-date and can cause confusion.
- Most debuggers cannot tolerate another interface issuing commands and may not start properly if another interface is running.

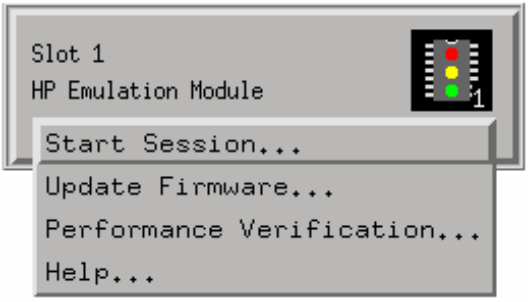
See Also

All of the Emulation Control Interface windows provide online help with a **Help** button or a **Help→On this window** menu selection. Refer to the online help for complete details about how to use a particular window.

To start the Emulation Control Interface from the main System window

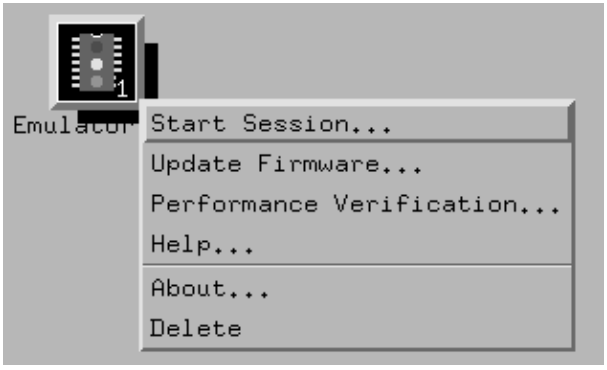
1 In the System window, click the emulation module icon.

2 Select **Start Session...**.



To start the Emulation Control Interface from the Workspace window

1. Open the Workspace window.
2. Drag the Emulator icon onto the workspace.
3. Right-click on the Emulator icon, then select **Start Session...**.



To start the Emulation Control Interface from the Workspace window for an emulation probe

If you have a stand-alone emulation probe connected to the logic analysis system via LAN, use the Emulation Probe icon instead of the Emulator icon.

1 Open the Workspace window.

2 Drag the Emulation Probe icon onto the workspace.

3 Right-click on the Emulation Probe icon, then select **Start Session...**.

4 In the Session window, enter the IP address or LAN name of the emulation probe, then click **Start Session**.

Designing a Target System for the Emulation Module

For your target system to work properly with the emulation module, it must meet the following requirements:

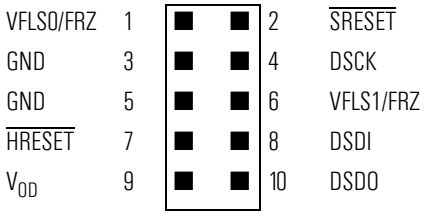
- The DSDI and DSCK signals must not be actively driven by the target system when the debug port is being used.
- The $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ signals from the debug connector must be ORed with the $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ signals that connect to the processor on the target system. They can be logically ORed or “wire-ORed” on the board. HP recommends “wire-ORing” the signals so that the emulation module can detect when the target is in reset. The emulation module will drive $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ through a 100Ω resistor.

The emulation module adds about 40 pF to all target system signals routed to the debug connector. This added capacitance may reduce the rise time of the $\overline{\text{SRESET}}$ or the $\overline{\text{HRESET}}$ signal beyond the processor specifications. If so, the target may need to increase the pull-up current on these signal lines.

Debug port connections

If you plan to connect the emulation module directly to the target system, the target system should have a debug port (BDM) connector.

The connector should be a dual row header strip ("Berg connector"), 10 pins per inch, with 25 mil pins.



Pins 1 and 6 may be connected to VFLS0 and VFLS1 respectively, or, if a single freeze line is used, to the FRZ line.

Debug port signals

Header Pin Number	Signal Name	I/O	MPC860/ 821 Pin No.*	Board Resistor
1	VFLS0	Out	H2	
2	$\overline{\text{SRESET}}$	In/Out	P2	10KΩ pullup
3	GND			
4	DSCK	In	H16	
5	GND			
6	VFLS1	Out	J3	
7	$\overline{\text{HRESET}}$	In/Out	N4	10KΩ pullup
8	DSDI	In	H17	
9	3.3 v			
10	DSDO	In	G17	

*Pin numbers are for 357 BGA packages.

See Also Chapter 18, "Development Support," of the Motorola *MPC860 User's Manual*.

Installing the Emulation Module

Your emulation module may already be installed in your logic analysis system. If you need to install an emulation module yourself, follow the instructions on the pages which follow.

CAUTION: Electrostatic discharge can damage electronic components. Use grounded wrist straps and mats when you handle modules.

To install the emulation module in an HP 16700A-series logic analysis system or an HP 16701A expansion frame

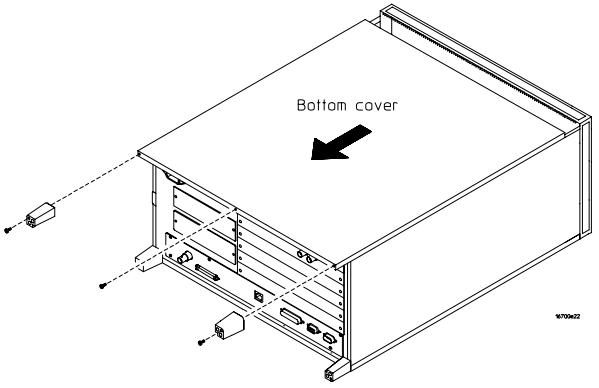
You will need a T-10 and T-15 Torx screw driver.

1 Turn off the logic analysis system and REMOVE THE POWER CORD.

Remove any other cables (including mouse or video monitor cables).

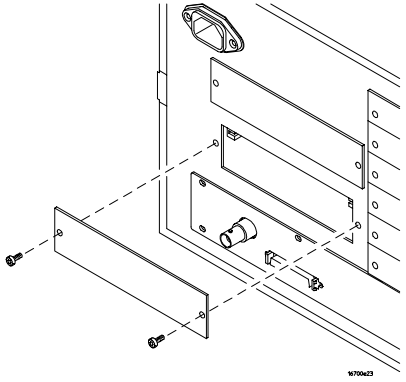
2 Turn the logic analysis system frame upside-down.

3 Remove the bottom cover.

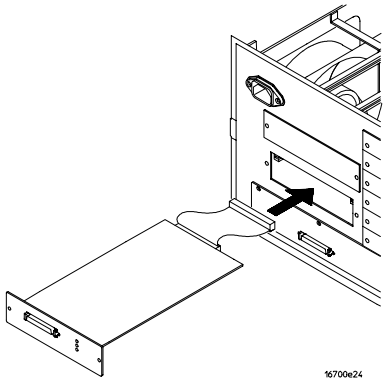


4 Remove the slot cover.

You may use either slot.

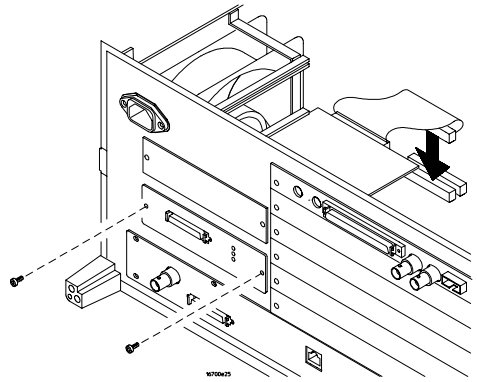


5 Install the emulation module.



6 Connect the cable and re-install the screws.

You may connect the cable to either of the two connectors. If you have two emulation modules, note that many debuggers will work only with the "first" module: the one toward the top of the frame ("Slot 1"), plugged into the connector nearest the back of the frame.



7 Reinstall the bottom cover, then turn the frame right-side-up.

8 Plug in the power cord, reconnect the other cables, and turn on the logic analysis system.

The new emulation module will be shown in the system window.

See Also

See page 181 for information on giving the emulation module a “personality” for your target processor.

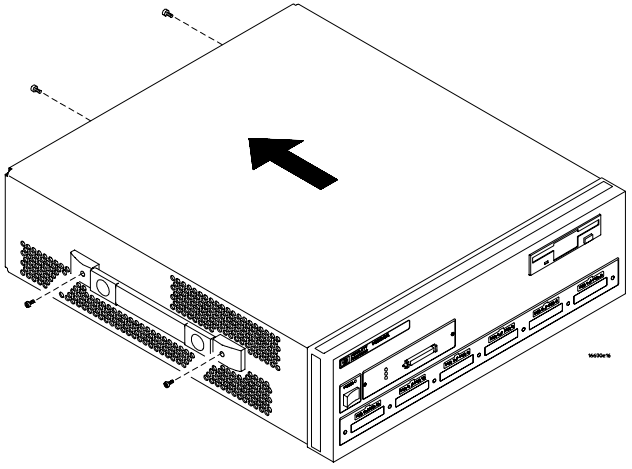
To install the emulation module in an HP 16600A-series logic analysis system

You will need a T-10 Torx screw driver.

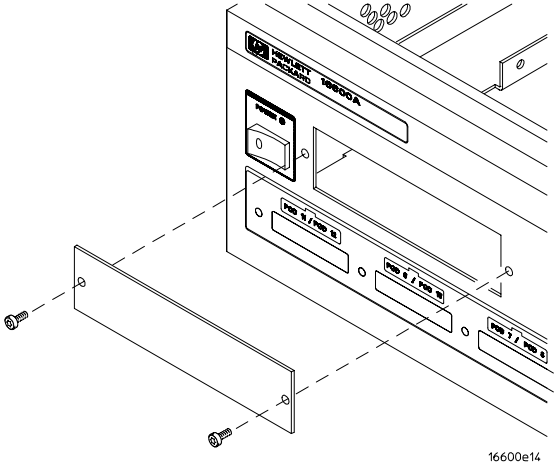
1 Turn off the logic analysis system and REMOVE THE POWER CORD.

Remove any other cables (such as probes, mouse, or video monitor).

2 Slide the cover back.

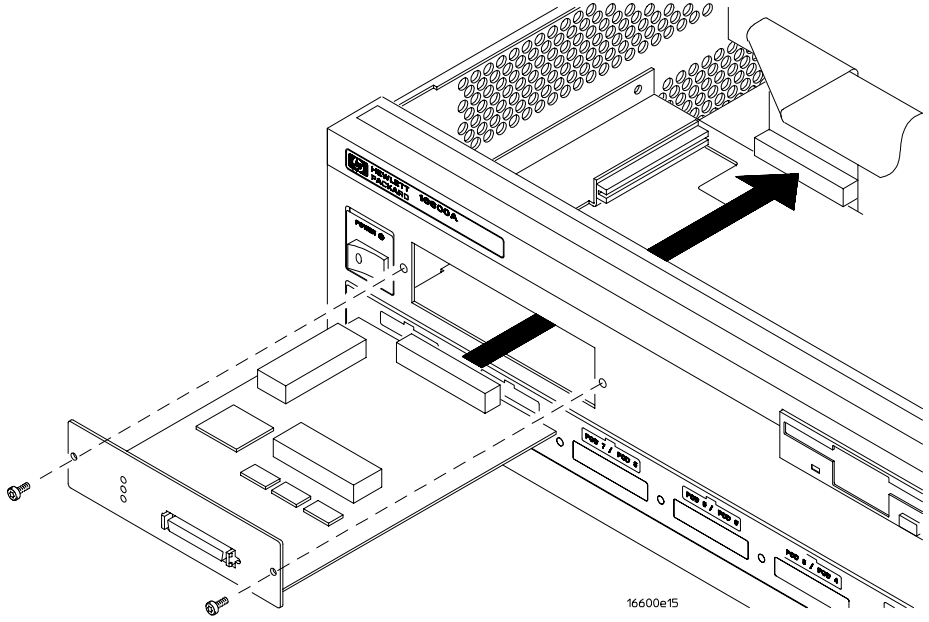


3 Remove the slot cover.



4 Install the emulation module.

5 Connect the cable and re-install the screws.



6 Reinstall the cover.

Tighten the screws snugly (2 N•m or 18 inch-pounds).

7 Plug in the power cord, reconnect the other cables, and turn on the logic analysis system.

The new emulation module will be shown in the system window.

See Also See page 181 for information on giving the emulation module a “personality” for your target processor.

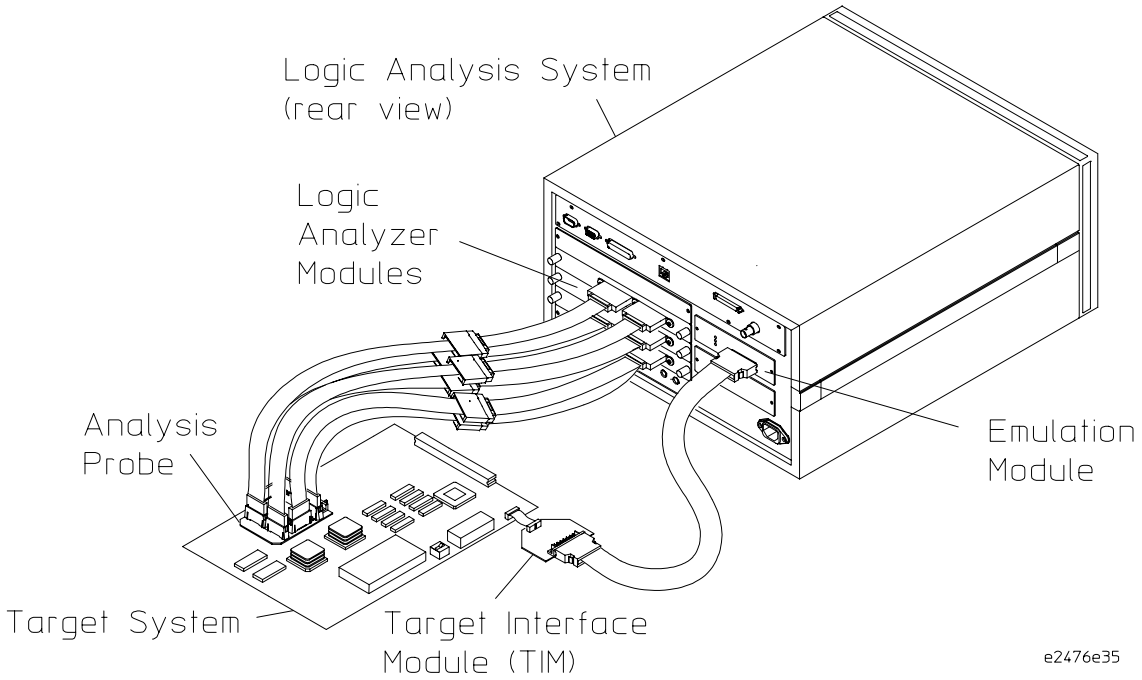
To test the emulation module

If this is the first time that you have used the emulation module, you should run the built-in performance verification test before you connect to a target system. Refer to page 312 in the “Troubleshooting” chapter for information on performance verification.

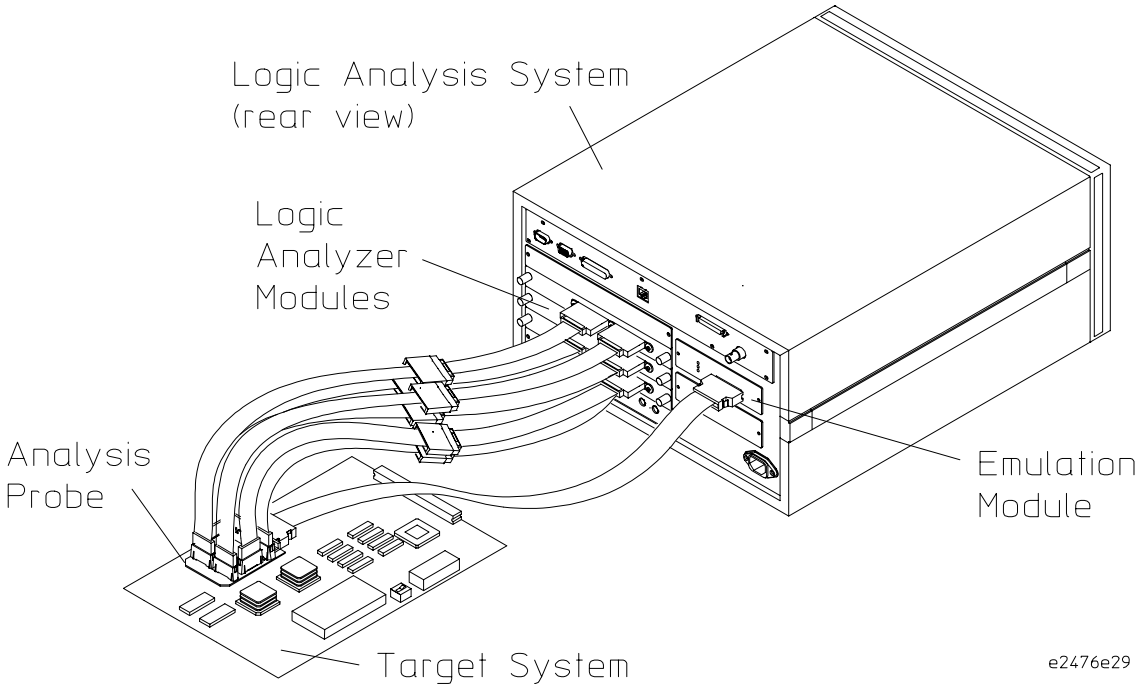
Connecting the Emulation Module to the Target System

Choose one of the following methods for connecting the emulation module to a target system.

- Directly through a debug port connector on the target board.



- Through an HP E2476A analysis probe, which provides a direct connection to the debug port pins.



After you have connected the emulation module to your target system, you may need to update the firmware in the emulation module.

See Also

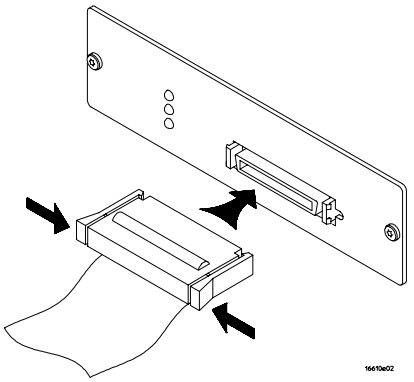
For information on designing a debug port on your target board, see page 169.

For a list of the parts supplied with the emulation module, see page 30.

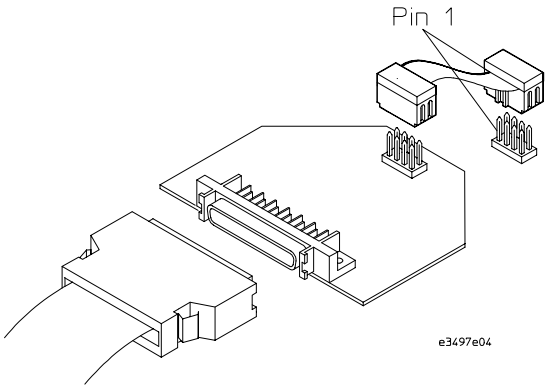
To connect to a target system using a debug port

The emulation module can be connected to a target system through a 10-pin debug port (BDM connector). The emulation module should be connected to a 10-pin male 2x5 header connector on the target system using the 10-conductor cable assembly provided.

1. Turn off the target system and disconnect it from all power sources.
2. Plug one end of the 50-pin cable into the emulation module.



3. Plug the other end of the 50-pin cable into the target interface module.
4. Plug one end of the 10-pin cable into the target interface module.
5. Plug the other end of the 10-pin cable into the debug port on the target system.



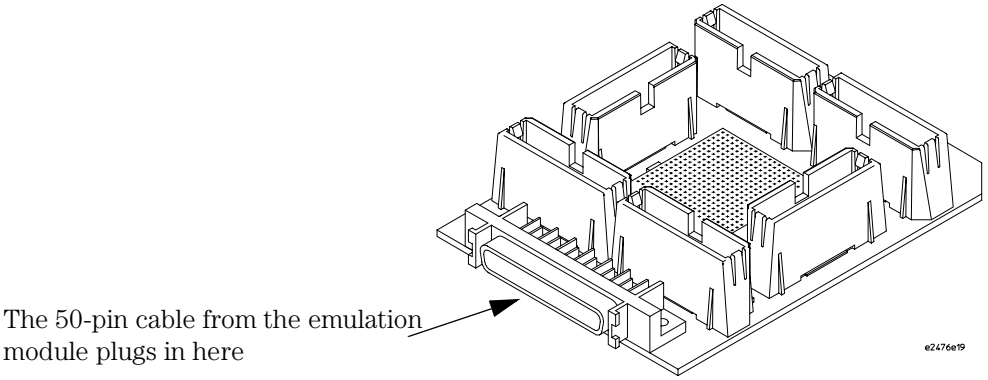
6 Turn on the power to the logic analysis system and then the target system.

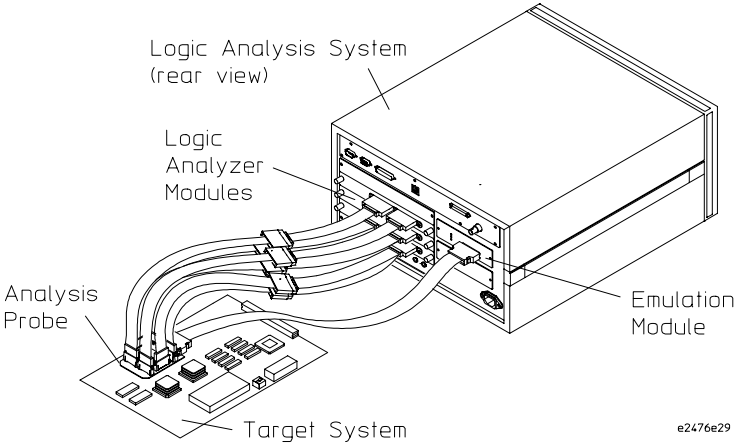
See Also

"Designing a Target System" (page 169) for information on designing a target system for use with the emulation module.

To connect to a target system using an analysis probe

1. Remove power from the target system.
2. Plug one end of the 50-pin cable into the emulation module.
3. Plug the other end of the 50-pin cable into the connector on the analysis probe.





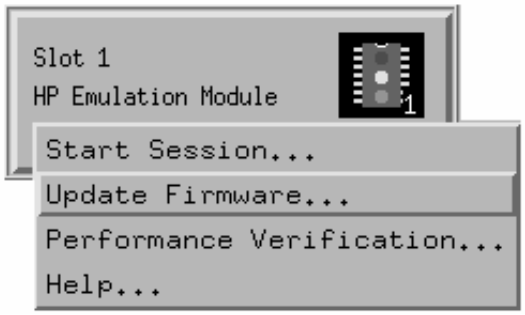
To update firmware

After you have connected the emulation module to your target system, you may need to update the firmware to give it the right "personality" for your processor. You must update the firmware if:

- The emulation module is being connected to a new analysis probe or TIM, or
- The emulation module was not shipped already installed in the logic analysis system, or
- You have an updated version of the firmware from HP.

To update the firmware:

1. End any run control sessions which may be running.
2. In the Workspace window, remove any Emulator icons from the workspace.
3. Install the firmware onto the logic analysis system's hard disk, if necessary.
4. In the system window, click the emulation module and select **Update Firmware**.



5. In the Update Firmware window, select the firmware version to load into the emulation module.
6. Click **Update Firmware**.

   In about 20 seconds, the firmware will be installed and the screen will update to show the current firmware version.

See Also

"Installing Software" beginning on page 35 for instructions on how to install the firmware files on the hard disk.

To display current firmware version information

- In the Update Firmware window, click **Display Current Version**.

  There are usually two firmware version numbers: one for "Generics" and one for the personality of your processor.

To verify communication between the emulator and target system

1. Turn on the target system.
2. Start the Emulation Control Interface.

   If the electrical connections are correct, and if the emulator firmware and analysis probe or TIM match your target processor, the Run Control window should be displayed:

Configuring the Emulation Module

The emulation module has several user-configurable options. These options may be customized for specific target systems and saved in configuration files for future use.

The easiest way to configure the emulation module is through the Emulation Control Interface in an HP 16600A or HP 16700A logic analysis system.

If you use the Emulation Control Interface, please refer to the online help in the Configuration window for information on each of the configuration options.

Other ways to configure the emulation module are by using:

- the emulation module's built-in terminal interface
- your debugger, if it provides an "emulator configuration" window which can be used with this HP emulation module

What can be configured

The following options can be configured using the Emulation Control Interface or using built-in commands:

- Processor type.
- Processor clock speed.
- Default reset level.
- "Break In" type.
- The emulation module's copy of the IMMR register.
- The emulation module's copy of the SYPCR register.
- The emulation module's copy of the DER register.
- The emulation module's copies of other internal registers.

  You may need to set up the emulator copies of the target processor registers. If you have a boot ROM that initializes these registers, you don't need to configure these registers because you can simply run the boot ROM. If you do not have a boot ROM, then you will need to initialize these registers so that you can communicate with the memory of the processor. Once these register copies are defined, then every reset followed by a break will write the emulator copies of the configuration registers to the processor.

  Once you have configured the register copies, it is a good idea to save a configuration. Loading the configuration will restore the values of all configuration options, including the register copies.

  The default values for the other options will allow the emulator to work with most target systems.

The following option can be configured using built-in commands:

- Restriction to real-time runs.

The built-in "help cf" command also lists the following options, which are provided only for compatibility with standalone emulation probes:

- BNC break in behavior.
- BNC trigger out behavior.

To configure using the Emulation Control Interface

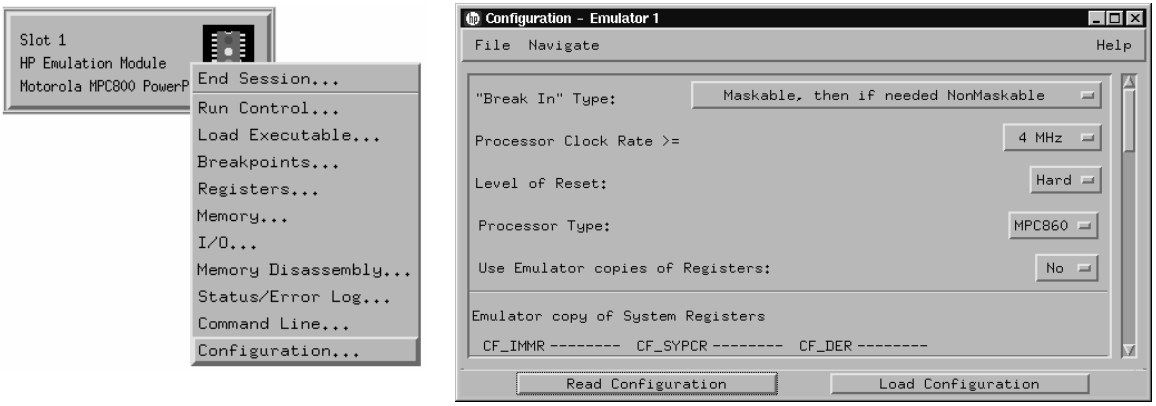
The easiest way to configure the emulation module is to use the Emulation Control Interface.

1 Start an Emulation Control Interface session.

In the system window, click the Emulation Control Interface icon, and then select "Start Session...".

2 Open a Configuration window.

Select "Configuration..." from the Emulation Control Interface icon or from the Navigate menu in any Emulation Control Interface window.



3 Set the configuration options, as needed.

The configuration selections will take effect when you close the configuration window or when you move the mouse pointer outside the window.

4 Save the configuration settings.

To save the configuration settings, open the File Manager window and click **Save...**.

See Also

Help →Help on this window in the Configuration window for information on each of the configuration options.

Help in the Emulation Control Interface menu for help on starting an Emulation Control session.

To configure using the built-in commands

If you are unable to configure the emulation module with the Emulation Control Interface or a debugger interface, you can configure the emulation module using the built-in "terminal interface" commands.

1. Connect a telnet session to the emulation module over the LAN.

   For example, on a UNIX system, for an emulation module in Slot 1 enter:

   `telnet` *`LAN_address`* `6472`

2. Enter `cf` to see the current configuration settings.
3. Use the `cf` command to change the configuration settings.

See Also

Enter `help cf` for help on the configuration commands.

For information on connecting using telnet, and for information on other built-in commands, see page 299.

Example

To see a complete list of configuration items, type “help cf”. This command displays:

```
cf - display or set emulation configuration

   cf                - display current settings for all config items
   cf <item>         - display current setting for specified <item>
   cf <item>=<value> - set new <value> for specified <item>
   cf <item> <item>=<value> <item> - set and display can be combined

 help cf <item>    - display long help for specified <item>

 --- VALID CONFIGURATION <item> NAMES ---

   proc     - Set type of target processor
   procck   - Set clock speed of target processor
   dprocck  - Display default clock speed of target processor
   bnchardbrk - Set BNC break type
   breakin  - BNC break in control
   cfreg    - Config Register Enable
   fastdnld - Silicon fast download status
   reset    - Set level of 'rst' command
   rrt      - Set restriction to real time runs
   trigout  - Trigger out control
M>
```

To configure using a debugger

Because the HP emulation module can be used with several third-party debuggers, specific details for sending the configuration commands from the debugger to the emulation module cannot be given here. However, all debuggers should provide a way of directly entering terminal mode commands to the emulation module. Ideally, you would create a file that contains the modified configuration entries to be sent to the emulation module at the beginning of each debugger session.

See Also

Information about specific debuggers in the “Using the Emulation Module with a Debugger” chapter (page 199).

Your debugger manual.

To configure the processor type

If you are using an MPC821, you need to set the PowerPC processor type.

Processor type configuration

Value	Emulation module configured for	Built-in command
MPC860	MPC860 (Default)	cf proc = MPC860
MPC850	MPC850	cf proc = MPC850
MPC821	MPC821	cf proc = MPC821
MPC801	MPC801	cf proc = MPC801

The `cfsave -s` command will store this configuration in the emulation module's flash memory. The `cfsave -r` command will restore this configuration.

To configure the processor clock speed

The BDM communication speed will be 1/3 of the configured processor clock speed. You may set the processor clock speed to a speed lower than the actual clock speed of your target system. Use the 25 MHz option for microprocessors running faster than 25 MHz.

Processor clock speed configuration

Value	Processor clock is at least	Built-in command
25	25 MHz	cf procck = 25
20	20 MHz	cf procck = 20
16	16 MHz	cf procck = 16
8	8 MHz	cf procck = 8
4	4 MHz (default)	cf procck = 4
1	1 MHz	cf procck = 1

Value	Processor clock is at least	Built-in command
512	512 kHz	cf procck=512
32	32 kHz	cf procck=32

Use the `cf dprocck` command to display the default clock speed.

To configure restriction to real-time runs

Real-time runs configuration

Value	Emulation module configuration	Built-in command
no	Allows commands which break to the monitor. Examples include commands which display memory or registers. (Default)	cf rrt=no
yes	No commands are allowed which break to the monitor, except "break," "reset," "run," or "step."	cf rrt=yes

To configure the default reset level

Reset level configuration

Value	The "reset" command will	Built-in command
hard	Cause a hard reset (Default)	cf reset=hard
soft	Cause a soft reset	cf reset=soft

To view the download speed

Download speed is automatically determined. The `cf fastdnld` command displays the download speed configuration.

Download speed configuration

Value	Meaning	Built-in command
yes	On-chip acceleration logic is functional.	cf fastdnld
no	On-chip logic is not functioning correctly. You are using one of the older processor revisions which had a bug in the fast download logic.	

To configure the "Break In" type

This option affects how the emulation module will react to a trigger in an intermodule measurement.

"Break In" type configuration

Value	What happens when the emulation module is triggered
Maskable	A trigger will immediately cause a maskable break. If the maskable break fails, a non-maskable break will be attempted. The delay between an attempted maskable break and the non-maskable break will allow many instructions to be executed. (Default)
NonMaskable	A trigger will immediately cause a non-maskable break. Use this value if you are trying to halt the processor in an ISR. The processor may not be able to continue running after the break.

To configure the emulation module's copy of the IMMR register

The IMMR register specifies the location of memory-mapped registers.

To set the emulation module's copy of the IMMR register using the built-in terminal interface, use the `reg cf_immr=`*value* command. Use a 32-bit hexadecimal value. Only the upper 16 bits are programmed into the IMMR—the lower 16 bits are read-only.

After a break from reset, the processor is programmed with the value from the emulation module's copy.

To display the value of the emulation module's copy using the built-in terminal interface, use the `reg cf_immr` command.

To configure the emulation module's copy of the SYPCR register

The processor's SYPCR register is automatically programmed from the emulation module's copy after a break from reset.

To set the emulation module's copy of the SYPCR register using the built-in terminal interface, use the `reg cf_sypcr=`*value* command. Use a 32-bit hexadecimal value. The value you enter will be ORed with 0x00000080 to ensure that the watchdog timer is disabled in background.

To display the value of the emulation module's copy using the built-in terminal interface, use the `reg cf_sypcr` command.

To configure the emulation module's copy of the DER register

The DER register (debug enable register) controls which exceptions cause the processor to enter debug mode. The processor's DER register is automatically programmed from the emulation module's copy after a break from reset.

To set the emulation module's copy of the DER register using the built-in terminal interface, use the `reg cf_der=`*value* command.

To display the value of the emulation module's copy using the built-in terminal interface, use the `reg cf_der` command.

The default value for this register is 0x3082400f.

To disable configuration registers

By default, the emulation module's copies of the memory-mapped registers and the IMMR, SYPCR, and DER registers are automatically programmed into the processor after a break from reset. You can disable this automatic initialization using the `cfreg` command.

Configuration register initialization

Value	Meaning	Built-in command
yes	Automatic register initialization is enabled. The processor probe's copies will be used to initialize the target. (Default)	cfreg = yes
no	Automatic register initialization is disabled. The values in the emulation module's copies of the configuration registers will be ignored.	cfreg = no

To configure a Motorola 860ADS target system

- Create an initialization file for your debugger or for the Emulation Control Interface.

The sample initialization file performs the following tasks:

- Disabling the watch-dog timer
- Maximizing the clock multiplier, and maximizing the debug clock for efficient download
- Programming the UPM registers
- Programming the chip selects

The initialization commands to do the above should be run after every reset;break that is done (because the reset will disrupt many of the register values).

```
# Configuration File for the Motorola MPC860 ADS Board
#
# This file will program the memory system after a
# target reset has occurred.

reg immr=ff000000             # Memory mapped registers
                              #  base address
reg msr=1002

m -a4 -d4 0@reg=01612440      # SIUMCR
m -a4 -d4 4@reg=ffffff88      # SYPCR  (disable watch-dog
                              #  timer)

m -a4 -d4 284@reg=0x00500000   # PLPRCR - Maximize the
clock
                              #  multiplier
cf procck=25                  # Maximize the debug clock
                              #  for download

# Configure the UPM
m -a4 -d4 0@upm=0fffcc24      # Single Read
m -a4 -d4 4@upm=0fffcc04
m -a4 -d4 8@upm=0cffcc04
m -a4 -d4 c@upm=00ffcc04
m -a4 -d4 10@upm=00ffcc00
m -a4 -d4 14@upm=37ffcc47
m -a4 -d4 18@upm=ffffffff
m -a4 -d4 1c@upm=ffffffff
m -a4 -d4 20@upm=0fffcc24     # Burst Read
```

```
m -a4 -d4 24@upm=0fffcc04
m -a4 -d4 28@upm=08ffcc04
m -a4 -d4 2c@upm=00ffcc04
m -a4 -d4 30@upm=00ffcc08
m -a4 -d4 34@upm=0cffcc44
m -a4 -d4 38@upm=00ffec0c
m -a4 -d4 3c@upm=03ffec00
m -a4 -d4 40@upm=00ffec44
m -a4 -d4 44@upm=00ffcc08
m -a4 -d4 48@upm=0cffcc44
m -a4 -d4 4c@upm=00ffec04
m -a4 -d4 50@upm=00ffec00
m -a4 -d4 54@upm=3fffec47
m -a4 -d4 58@upm=ffffffff
m -a4 -d4 5c@upm=ffffffff
m -a4 -d4 60@upm=0fafcc24    # Single Write
m -a4 -d4 64@upm=0fafcc04
m -a4 -d4 68@upm=08afcc04
m -a4 -d4 6c@upm=00afcc00
m -a4 -d4 70@upm=37ffcc47
m -a4 -d4 74@upm=ffffffff
m -a4 -d4 78@upm=ffffffff
m -a4 -d4 7c@upm=ffffffff

m -a4 -d4 80@upm=0fafcc24    # Burst Write
m -a4 -d4 84@upm=0fafcc04
m -a4 -d4 88@upm=08afcc00
m -a4 -d4 8c@upm=07afcc4c
m -a4 -d4 90@upm=08afcc00
m -a4 -d4 94@upm=07afcc4c
m -a4 -d4 98@upm=08afcc00
m -a4 -d4 9c@upm=07afcc4c
m -a4 -d4 a0@upm=08afcc00
m -a4 -d4 a4@upm=37afcc47
m -a4 -d4 a8@upm=ffffffff
m -a4 -d4 ac@upm=ffffffff
m -a4 -d4 b0@upm=ffffffff
m -a4 -d4 b4@upm=ffffffff
m -a4 -d4 b8@upm=ffffffff
m -a4 -d4 bc@upm=ffffffff
m -a4 -d4 c0@upm=e0ffcc84    # Refresh
m -a4 -d4 c4@upm=00ffcc04
m -a4 -d4 c8@upm=00ffcc04
m -a4 -d4 cc@upm=0fffcc04
m -a4 -d4 d0@upm=7fffcc04
m -a4 -d4 d4@upm=ffffcc86
m -a4 -d4 d8@upm=ffffcc05
m -a4 -d4 dc@upm=ffffffff
m -a4 -d4 e0@upm=ffffffff
m -a4 -d4 e4@upm=ffffffff
m -a4 -d4 e8@upm=ffffffff
```

```
m -a4 -d4 ec@upm=ffffffff

m -a4 -d4 f0@upm=33ffcc07    # Exception
m -a4 -d4 f4@upm=ffffffff
m -a4 -d4 f8@upm=ffffffff
m -a4 -d4 fc@upm=40004650
# Program the Chip Selects for the following memory map
# 0..400000         - DRAM
# 2100000..         - ADS Board Registers
# 2800000..2900000 - Flash
# ff00000           - CPU

m -a4 -d4 100@reg=02800001   # BR0 - Flash
m -a4 -d4 104@reg=ffe00d34   # OR0
m -a4 -d4 108@reg=02100001   # BR1 - ADS Registers
m -a4 -d4 10c@reg=ffff8110   # OR1
m -a4 -d4 110@reg=00000081   # BR2 - DRAM
m -a4 -d4 114@reg=ffc00800   # OR2

m -a2 -d2 17a@reg=0400
m -a4 -d4 170@reg=9ca21114

m -a2 -d2 200@reg=00c2
m -a2 -d2 240@reg=0082

reg srr1=1002
reg der=ffe7400f
```

Testing the emulator and target system

After you have connected and configured the emulator, you should perform some simple tests to verify that everything is working.

See Also

"Troubleshooting the Emulation Module" on page 295 for information on testing the emulator hardware.To test memory accesses

To test memory accesses

1. Start the Emulation Control Interface and configure the emulator, if necessary.
2. Open the Memory window.
3. Write individual locations or fill blocks of memory with patterns of your choosing.

   The access size is the size of memory access that will be used to write or read the memory values.
4. Use the Memory I/O window to stimulate I/O locations by reading and writing individual memory locations.

To test with a running program

To more fully test your target, you can load simple programs and execute them.

1. Compile or assemble a small program and store it in a Motorola S-Record or Intel Hex file.
2. Use the Load Executable window to download the program into RAM or flash memory.

3 Use the Breakpoints window to set breakpoints. Use the Registers window to initialize register values.

   The new register or breakpoint values are sent to the processor when you press the Enter key or when you move the cursor out of the selected register field.

4 In the Run Control window, click **Run**.

5 Use the Memory Mnemonic window to view the program and use the Memory window to view any output which has been written to memory.

9

Using the Emulation Module with a Debugger

Several prominent companies design and sell state-of-the-art source debuggers that work with the HP emulation module and emulation probe.

Benefits of using a debugger

The debugger will enable you to control the execution of your processor from the familiar environment of your debugger. Using a debugger lets you step through your code at the source-code level.

With a debugger connection, you can set breakpoints, single-step through source code, examine variables, and modify source code variables from the debugger interface. The debugger can also be used to download executable code to your target system.

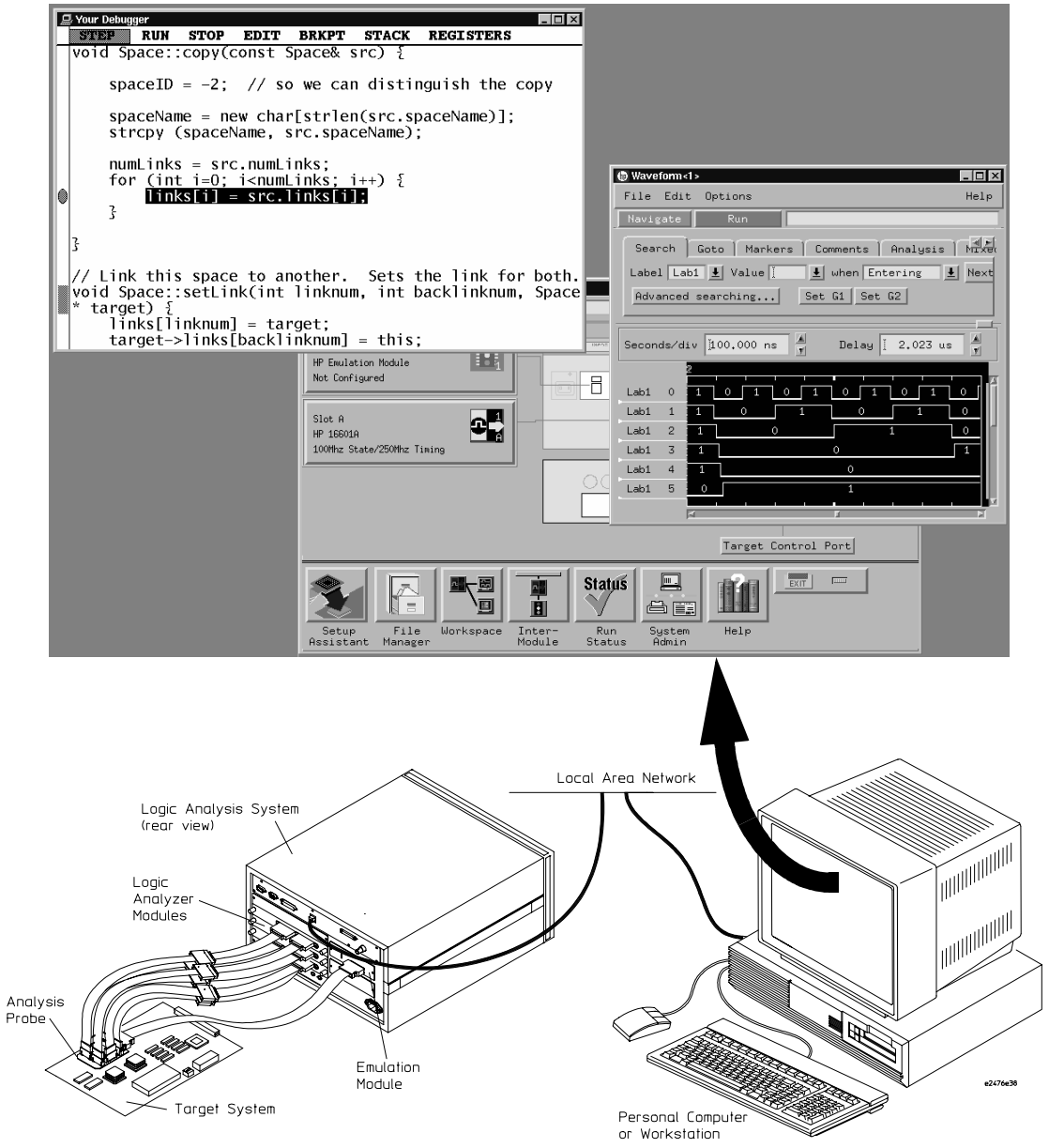
Using a debugger to connect the emulation module allows the entire design team to have a consistent interface from software development to hardware/software integration.

Debugger interfaces must be ordered directly from the debugger vendor.

Compatibility with other logic analysis system tools

You can use your logic analysis system to collect and analyze trace data while you use your debugger. If you are using an X windows workstation or a PC with an X terminal emulator, you can display the logic analyzer windows right next to your debugger.

Here is an example of what the display on your PC or workstation might look like.



Minimum requirements

To use a debugger with the emulation module, you will need:

- A debugger which is compatible with the emulation module
- A LAN connection between the PC or workstation that is running the debugger, and the HP 16600A or HP 16700A logic analysis system
- X windows or an X terminal emulator, such as Reflection X on a PC. This is required only if you wish to have the logic analysis system user interface displayed on your PC or workstation screen, along with the debugger.

Is your debugger compatible with the emulation module?

Ask your debugger vendor whether the debugger can be used with an HP emulation module or HP emulation probe (also known as a "processor probe" or "software probe").

LAN connection

You will use a LAN connection to allow the debugger to communicate with the emulation module.

Compatibility with the Emulation Control Interface

Do not use the logic analysis system's Emulation Control Interface and your debugger at the same time.

Setting up Debugger Software

The instructions in this manual assume that your PC or workstation is already connected to the LAN, and that you have already installed the debugger software according to the debugger vendor's documentation.

To use your debugger with the emulation module, follow these general steps:

- Connect the emulation module to your target system (page 164).
- Connect the logic analysis system to the LAN (page 204).
- Export the logic analysis system's display to your PC or workstation (page 207).
- Configure the emulation module (page 183).
- Begin using your debugger.

If you use the Emulation Control Interface to configure the emulation module, remember to end the Emulation Control Interface session before you start the debugger.

CAUTION: Do not use the Emulation Control Interface at the same time as a debugger.

The Emulation Control Interface and debuggers do not keep track of commands issued by other tools. If you use both at the same time, the tools may display incorrect information about the state of the processor, possibly resulting in lost data.

See Also Refer to the documentation for your debugger for more information on connecting the debugger to the emulation module.

To connect the logic analysis system to the LAN

Information on setting up a LAN connection is provided in the online help or installation manual for your logic analysis system.

Your debugger will require some information about the LAN connection before it can connect to the emulation module. This information may include:

- IP address (Internet address) or LAN name of the logic analysis system.
- Gateway address of the logic analysis system.
- Port number of the emulation module.

Port numbers for emulation modules

Port number	Use for
Debugger connections	
6470	Slot 1 (First emulation module in an HP 16600A/700A-series logic analysis system)
6474	Slot 2 (Second emulation module in an HP 16700A-series system)
6478	Slot 3 (Third emulation module in an expansion frame)
6482	Slot 4 (Fourth emulation module in an expansion frame)
Telnet connections	
6472	Slot 1 (First emulation module)
6476	Slot 2 (Second emulation module)
6480	Slot 3 (Third emulation module)
6484	Slot 4 (Fourth emulation module)

Write the information here for future reference:

IP Address of Logic Analysis System _____

LAN Name of Logic Analysis System _____

Gateway Address _____

Port Number of Emulation Module _____

To change the port number of an emulation module

Some debuggers do not provide a means to specify a port number. In that case, the debugger will always connect to port 6470 (the first emulation module). If you need to connect to another module, or if the port number of the first module has been changed, you must change the port number to be 6470.

To view or change the port number:

1. Click on the emulation module icon in the system window of the logic analysis system, then select Update Firmware.
2. Select Modify LAN Port....
3. If necessary, enter the new port number in the LAN Port Address field.

   The new port number must not be 0-1000 and must not already be assigned to another emulation module.

To verify communication with the emulation module

1. telnet to the IP address.

   For example, on a UNIX system, enter "telnet <*IP_address*> 6472". This connection will give you access to the emulation module's built-in terminal interface. You should see a prompt, such as "M>".

2. At the prompt, type:

   ```
   ver
   ```

   You should then see information about the emulation module and firmware version.

3. To exit from this telnet session, type <CTRL>D at the prompt.

See Also

The online help or manual for your logic analysis system, for information on physically connecting the system to the LAN and configuring LAN parameters.

"Troubleshooting," page 310, if you have problems verifying LAN communication.

To operate the logic analysis system using a web browser

HP 16600/700A-series logic analysis systems may be monitored and controlled using a web browser.

Information on connectivity is provided in the online help or installation manual for your logic analysis system.

To export the logic analysis system's display to a workstation

By exporting the logic analyzer's display, you can see and use the logic analysis system's windows on the screen of your workstation. To do this, you must have telnet software and X windows installed on your computer.

1. On the workstation, add the host name of the logic analysis system to the list of systems allowed to make connections:

   `xhost +<IP_address>`

2. Use telnet to connect to the logic analysis system.

   `telnet <IP_address>`

3. Log in as "hplogic".

   The logic analysis system will open a Session Manager window on your display.

4. In the Session Manager window, click Start Session on This Display.

Example

On a UNIX workstation, you could use the following commands to export the display of a logic analysis system named "mylogic":

```
$ xhost +mylogic
$ telnet mylogic
Trying...
Connected to mylogic.mycompany.com.
Escape character is '^]'.
Local flow control on
Telnet TERMINAL-SPEED option ON

HP Logic Analysis System
```

```
Please Log in as: hplogic [displayname:0]

login: hplogic
Connection closed by foreign host.
$
```

To export the logic analysis system's display to a PC

By exporting the logic analyzer's display, you can see and use the logic analysis system's windows on the screen of your PC. To do this, you must have telnet software and an X terminal emulator installed on your computer. The following instructions use the Reflection X emulator from WRQ, running on Windows 95, as an example.

1. On the PC, start the X terminal emulator software.

   To start Reflection X, click the Reflection X Client Start-up icon.

2. Start a telnet connection to the logic analysis system.

   Log in as "hplogic".

   For Reflection X, enter the following values in the Reflection X Client Start-up dialog:

   a. In the Host field, enter the LAN name or IP address of the logic analysis system.

   b. In the User Name field, enter "hplogic".

   c. Leave the Password field blank.

   d. Leave the Command field blank.

   e. Click Run to start the connection.

   The logic analysis system will open a Session Manager window on your display.

3. In the Session Manager window, click Start Session on This Display.

Using the Green Hills debugger

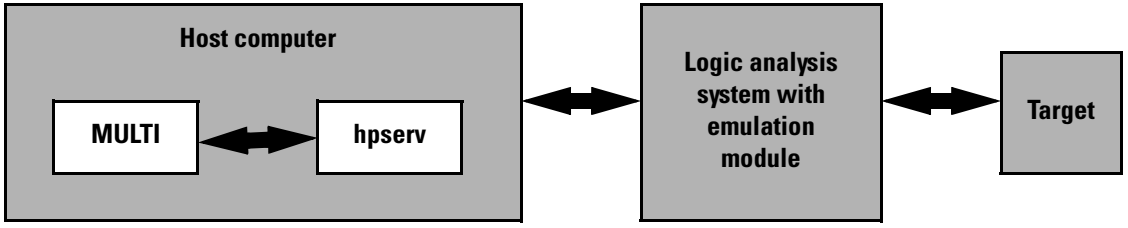
Compatibility

Version 1.8.8.A of the MULTI Development Environment from Green Hills Software, Inc. is one of several debuggers that connect to the HP emulation module.

This section provides information that is specific to using MULTI with the HP emulation module. It is intended to be used in conjunction with the MULTI documentation provided by Green Hills Software.

Overview

MULTI connects to an emulation module through the Green Hills host-resident program (hpserv).



Getting started

1 Check that your Emulation Module is programmed with firmware for an MPC8xx processor.

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a “Motorola MPC800 PowerPC Emulator”. If it is not, follow the instructions on page 181 to update the firmware.

2 Build the executable.

If you have the demo software shipped with the Green Hills debugger, follow these steps:

a Prepare the executable.

Go to the hpdemo subdirectory where you installed MULTI. Copy the mbx800.lnk file to user.lnk.

b Start MULTI.

On Unix, enter "multi".

On Windows, double-click the Green Hills icon.

c Set up the MULTI software environment:

- Replace the project default.bld (in the Builder dialog box next to the project button) with hpdemo/default.bld and press ENTER.
- Make sure the target button on the MULTI window says "PPC".
- In the Builder window, double-click ecs.bld.

The box next to the Debug button should display “ecs”. The window should list the names of the source code files.

d In the Builder menu bar, select Options→**CPU**, then set the processor type.

e In the Builder menu bar, select Options→**Advanced**, and make sure that "Output DWARF on ELF targets" option is enabled.

f Build the demo program:

- In the Builder window, click the Build icon. (Or, in the menu bar, select **Build→Build All**.)

- Close the Progress window when the "Build completed" message is displayed.

3 Connect MULTI to the emulation module.

There are two ways to connect to the emulation module:

- In the Remote box in the MULTI Builder window, enter:

  ```
  hpserv IP_address
  ```

  OR

- In the Builder window, click Debug to open the Debugger window, then in the Debugger window's command pane, enter:

  ```
  remote hpserv IP_address
  ```

Starting hpserv opens two windows: the Target window and the I/O window. Commands entered in the Target window are sent directly to the emulation module.

The I/O window sends input (stdin) to and receives output (stdout) from the target program while it is running.

Note that hpserv connects to the first emulation module (port 6470) in a logic analysis system frame. You may specify another port by using the -p option with hpserv. See page 204 for more information on port numbers.

4 Start the debugger.

If you have not opened the Debugger window yet, click **Debug** in the Builder window.

5 Configure the emulation module and target system.

Before running the target processor, you must configure the HP emulation module for your target system. For example, you may have to set the BDM clock speed, the reset operation, cache disabling, or other configuration parameters.

If you are unsure of the configuration needed for your emulation module, you can use the Configuration window in the logic analysis system's Emulation Control Interface to explore the configuration options.

For additional help on configuration commands, refer to the example configuration command file for a Motorola MPC860 ADS board on page 193.

Once you know the configuration settings needed for your target system, you may use one of the following methods to configure the emulation module and target system:

- Use the Configuration window in the logic analysis system's Emulation Control Interface.
- Enter "cf" commands in the Target window.
- Use an initialization script.

  See the "To configure the emulation module and target system using an initialization script" section on page 212 for information on saving the configuration commands in a script.

6 Specify an initialization address for the stack pointer.

This is required if the stack pointer is neither initialized when the processor is reset nor set in the start-up code generated by the compiler. If the stack pointer address needs to be initialized:

- In the debugger's command pane, enter:

  `_INIT_SP = <address>`

OR

- In the Target window, enter:

  `reg r1=<address>`

OR

- Include the following line in an initialization script:

  `target reg r1=<address>`

7 Download the code:

- In the Debugger window, select **Remote→LoadProgram**.

  The Debugger command pane indicates that the code has been downloaded to the target.

To configure the emulation module and target using an initialization script

You can use an initialization script to configure the emulation module and set up your target system. If you will always be using the same configuration, this way will save time and reduce errors.

8 Save the configuration commands in a text file, one command per line.

An example configuration command file (for a Motorola MPC860 ADS board) is on page 193.

Green Hills also provides an example initialization sequence in the file MBX800.rc in the “hpdemo” directory.

9 To run the script, enter the following command in the Debugger command pane:

```
<filename
```

Example

Create a file with the following lines:

```
remote hpserv hplogic1
 target cf reset=soft
 _INIT_SP=0x10000
```

Save the file in the MULTI start-up directory and name it hpserv.rc. To run the script, enter the following command in the Debugger command pane:

```
<<hpserv.rc
```

When run, this script will:

- Connect to the target through the emulation module in a logic analysis system frame called "hplogic1".
- Set the reset level to soft.
- Initialize the stack pointer.

To perform common debugger tasks

- To display registers, click the regs button in the Display window.
- To set a breakpoint, click on the source code line where the breakpoint is to be located.
- To clear a breakpoint, click again on the source line.
- To step through code, click next.
- To run from the current PC, click go.
- To toggle the display between source code and source code interlaced with assembly code, click assem.
- To load program symbols, reset the PC, reset the stack pointer, and run from the start, click restart.

To send commands to the emulation module

MULTI communicates to the emulation module using the emulation module's "terminal interface" commands. MULTI automatically generates and sends the commands required for normal operation. If you want to communicate directly with the emulation module during a debug session, you may do so using "terminal interface" commands through the Target window (which comes up when hpserv is brought up). You can also enter these commands from the Debugger window's command pane by preceding the command with the "target" command.

To view commands sent by MULTI to the emulation module

The communication between MULTI and the emulation module can be viewed by running hpserv in a logging mode:

```
remote hpserv -dc -a  -o <filename> <emulation module name>
```

The options -dc and -da log both asynchronous and console messages and the -o <*filename*> directs these messages to a log file called <*filename*>. When using this option, disconnect from hpserv (to flush out the file) and then you may view <*filename*> to see what commands MULTI sent to the emulation

module.

NOTE: logging commands in this way may result in a VERY large file. Beware of the disk space it may require.

To reinitialize the system

If you suspect that the emulation module is out of sync with the MULTI debugger, you may want to reinitialize it. Perform the steps below to accomplish reinitialization:

10 In the Target window, type:

```
init -c
```

11 Repeat steps 5 through 8 in the "Getting started" section to configure the emulation module.

To disconnect from the emulation module

- In the Debugger window, select Remote→**Disconnect**.

The Debugger command pane indicates that the debugger has disconnected from the emulation module.

Error conditions

"!ERROR 800! Invalid command: bcast" usually means that there is not a target interface module (TIM) connected to the emulation module or the emulation module does not have firmware for the MPC800 family. Verify that the emulation module is connected to the target. Next, go to the system window of the logic analyzer interface and verify that the Emulation Module icon (stop-light) is described as a Motorola MPC800 PowerPC Emulator. If it is not, follow the steps on page 181 to update the firmware in the Emulation module for MPC800 processors.

"command socket connection failed: WSAECONNREFUSED: connection refused" usually means the emulation module is not at port #6470 on the Logic Analysis System.

See Also

Green Hills MULTI Software Development Environment User's Guide.

Using MULTI with the Hewlett-Packard Processor Probe from Green Hills Software, Inc.

The Green Hills web site: **http://www.ghs.com**

The configuration section beginning on page 183 for more information on configuration options and the “cf” command.

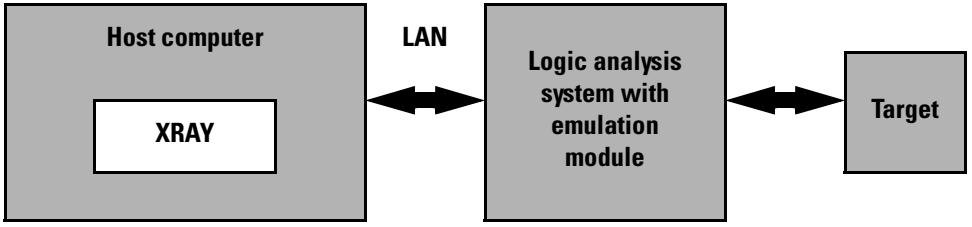
Using the Microtec Research debugger

Compatibility

Versions 4.1 and 4.2 of the XRAY HP Probe debugger from Microtec Research, a Mentor Graphics Company, are two of several debuggers that connect to the HP emulation module.

This section provides information that is specific to using XRAY with the HP emulation module. It is intended to be used in conjunction with the XRAY documentation provided by Microtec Research.

Overview



Getting started

1 Check that your Emulation Module is programmed with firmware for an MPC8xx processor.

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a “Motorola MPC800 PowerPC Emulator”. If it is not, follow the instructions on page 181 to update the firmware.

2 Edit the gtw.brd file.

The file gtw.brd includes example hostnames, port numbers and initialization information for HP emulation modules that might be on the network for XRAY to connect to. The gtw.brd file is in the "etc" directory under the Microtec tools directory.

a Modify gtw.brd to identify the emulation probe.

Modify the file to include the emulation probe that you want XRAY to communicate with.

See page 204 for information on which port number to use for your emulation module.

b Add commands to initialize the target system.

The target system must have various memory locations initialized before it can access RAM and before XRAY can download an application. Normally, code in the target's boot ROM performs this initialization. However, when XRAY resets the target, it immediately places the processor in debug mode. Therefore, any initialization code which may exist on the target board will not have been executed.

XRAY provides a way for target initialization to occur through the gtw.brd file. The initialization sequences (contained in "{}" pairs) included in the gtw.brd commands specify the commands that will be sent to the HP emulation module to initialize it and prepare it for code download.

The example gtw.brd file provided by Microtec Research contains initialization sequences which can be referenced. Please also refer to the example configuration command file for a Motorola PC860 ADS board on page 193 for additional initialization information. If the configuration for your target board is very involved, as is the configuration on page 193, you can use the "gtwinit" definition in gtw.brd to merely reset the processor and break and use an include file to do the many configuration steps. Please refer to the section "Using an INCLUDE file to configure the emulation module and target" on page 218 for more information on using an include file.

If you are unsure of the configuration needed for your emulation module, you can use the Configuration window in the logic analysis system's Emulation Control Interface to explore the configuration options. If you use this interface to actually configure your emulation module while connected to XRAY, exit the Emulation Control Interface before you start

debugging with the XRAY debugger.

NOTE: You must start up XRAY from scratch after gtw.brd is modified for the changes you have made in gtw.brd to be recognized by XRAY.

3 Start XRAY.

After modifying gtw.brd, bring up the XRAY debugger. When XRAY comes up, the Managers dialog will be highlighted. (If the dialog is not present, the Managers dialog can be brought up from the Output Logging Window by selecting **Managers→Connection Manager**).

Using the Managers dialog, set up the connection to your HP emulation module by selecting the Connections tab, clicking on your emulation module name in the lower Available Connections table and click on the **connect** button. You should see your emulation module name appear in the Active Connections table in the top half of the dialog. At this point, you are connected to the emulation module and the initialization commands specified in the gtw.brd file have been sent to your emulation module. If you look in the Output Logging Window, you can verify that the connection and initialization did in fact take place.

4 Download the application code.

In the Managers dialog, select the Debug tab, then select **execution→Load File to Target** or **control→Load File to Target**. This will open the "Load File To Target" dialog. (Alternatively, you may select the Files tab and select **load→Load File to Target**.)

Use the Load File To Target dialog to choose the file you would like to download. When the file you want is listed in the center window, you may double click on it to start the load.

When the load is complete, you will see the file you loaded appear in the Active Files window of the File tab and in the Active Processes window of the Debug tab. You are now ready to debug your application code.

To configure the emulation module and target using an INCLUDE file

You can use an include file to configure the emulation module and set up your target system after bringing up the XRAY debugger. If a complex configuration is needed for your emulation module and target (such as multi-commands sent to the emulation module) this will save time and reduce errors.

5 Save the configuration commands in a text file, one command per line. An example of configuration commands for a Motorola MPC860ADS board is on page 193. Microtec Research provides an example include file in its tools directory under the xhippchp directory in the file “mo8xxads.inc”.

6 To run the include file, select "Include Commands from File" under the Debug menu in the Code window and double click on the include filename you want to execute.

To perform common debugger tasks

- To display registers, select Register under the Windows menu in the Code window.
- To set a breakpoint, double click on the source code line where the breakpoint is to be located.
- To clear a breakpoint, double click on the line where the breakpoint is set.
- To step through code, select one of the step icons at the top of the Code window.
- To run from current PC, click on the first icon in the Code window.
- To toggle the display between source code and source code interlaced with assembly code, click on the Dsm button at the bottom of the code display window.
- To load program symbols, reset the PC, reset the stack pointer, and run from start, click restart.

gtwinit and gtwreset command sequences

The gtwinit command sequence defined in the gtw.brd file is sent to the HP emulation module when XRAY is establishing connection with the module.

The gtwreset command sequence is sent to the emulation module when the XRAY “Reset” command is invoked.

To send commands to the emulation module

"Terminal interface" commands may be sent directly to the emulation module from XRAY. There are two ways to do this:

- Using an include file (as explained in the "Using an INCLUDE file to configure the emulation module and target" section)

OR

- Using the XRAY “cf” command.

  This command takes a string as a parameter and sends it to the emulation module. For example, if you want to send the emulation module command cf rst=soft, you can type

  ```
  cf "rst=soft"
  ```

  in the XRAY Debugger command line.

  Note that the command must be surrounded by double quotes.

To view commands sent by XRAY

XRAY communicates with the emulation module using the emulation module’s "terminal interface" commands. XRAY automatically generates and sends the commands required for normal operation. The communication between XRAY and the emulation module can be logged to a file after a connection has been established between XRAY and the emulation module and viewed later. To enable logging, enter the command:

```
PROBEMESSAGE ON,msgfile
```

This will create the “msgfile” and log a summary of the messages that occur between XRAY and the emulation module to it. The logging can be turned off with the following command:

```
PROBEMESSAGE OFF
```

To disconnect from the emulation module and target

In the Managers window, select the Connect tab. Click on the emulation module name that you want to disconnect. Under the Control menu, select “Disconnect from Board” (or you can “Reconnect to Board” if you have lost connection to the emulation module).

Error conditions

"!ERROR 800! Invalid command: bcast" usually means that there is not a target interface module (TIM) connected to the emulation module or the emulation module does not have firmware for the MPC800 family. Verify that the emulation module is connected to the target. Next, go to the system window of the logic analyzer interface and verify that the Emulation Module icon (stoplight) is described as a Motorola MPC800 PowerPC Emulator. If it is not, follow the steps on page 181 to update the firmware in the Emulation module for MPC800 processors.

"command socket connection failed: WSAECONNREFUSED: connection refused" usually means the emulation module is not at port #6470 on the Logic Analysis System.

See Also

The Microtec Research web site: **http://www.mentorg.com/microtec**

The *XRAY Debugger Reference Manual* by Microtec Research.

The configuration section beginning on page 183 for more information on configuration options and the “cf” command.

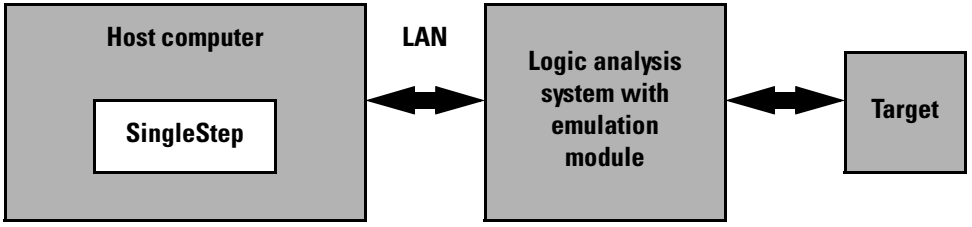
Using the Software Development Systems debugger

Compatibility

Versions 7.2, 7.3, and 7.4 of the SingleStep debugger from Software Development Systems, Inc. are some of the debuggers that connect to the HP emulation module.

This section provides information that is specific to using SingleStep with the HP emulation module. It is intended to be used in conjunction with the SingleStep documentation provided by SDS.

Overview



Startup behavior

The following actions are performed at the start of a session and when you select **File→Debug:**

- If the target reset option is selected, the target is reset and programmed with the register values previously entered in the SingleStep configuration dialog, or by the register initialization commands provided by a configuration script (<*filename*>.cfg). Refer to the SingleStep product release notes and SingleStep documentation for further information.
- Hardware breakpoints are disabled. For further information on the use of hardware breakpoints, please refer to “On-chip breakpoints and debugging ROM code” on page 227.
- Software breakpoints are enabled.

- All breakpoints are cleared.
- main() _exit breakpoints are set, if that option is selected.

Getting started

1 Check that your Emulation Module is programmed with firmware for an MPC8xx processor:

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a "Motorola MPC800 PowerPC Emulator". If it is not, follow the instructions on page 181 to update the firmware.

2 Connect to the emulation module:

a Start SingleStep running on your PC or workstation.

b When the small Debug dialog box appears in the middle of the screen, click the Connection tab and then enter the IP address of the HP logic analysis system which contains the emulation module.

If the Debug dialog box is not visible, select **File→Debug.**

Note: SingleStep is hard-coded to connect to the emulation module at port 6470 of the logic analysis system frame. See page 205 for more information on port numbers.

3 Configure the emulation module with the processor clock speed.

In the Debug dialog box, click the Connection tab and then enter a Processor Clock speed which is less than or equal to the speed at which the processor will run out of reset.

The emulation module must know the target clock speed before it can communicate with the target. This value depends on the oscillator or crystal used on your target system and the multipliers applicable at reset. The communications speed can be changed (see the Download Performance section on page 227) but will be reset to this value each time SingleStep resets the processor.

4 Initialize the target system.

The target system must have various registers and memory locations initialized before it can access RAM and before SingleStep can download an application. Normally, code in the target's boot ROM performs this initialization. However, when SingleStep resets the target, it immediately

places the processor in debug mode. Any initialization code which may exist on the target board has not been run.

SingleStep provides a way for target initialization to occur without running application code through the use of the "_config" alias. _config is used to define a list of commands that will be used to initialize the target after a reset. The _config alias should be defined in the sstep.ini file (in the "cmd" directory) and will point to a file of type .cfg which contains the actual initialization commands. An example of a command which defines _config can be seen in the sample file 821ads.ini.

SDS provides initialization files for some common targets. The "cmd" directory contains the following files for initializing the MPC821/860ADS board UPM memory controller and other configuration registers:

SDS initialization files

File	Contents
821ads.ini	commands which should be added to the sstep.ini file which initialize the _config alias so it points to 821ads.cfg
821ads.cfg	configuration script of initialization commands for most of the configuration registers
adsupm.dbg	configuration script of initialization commands for the UPM memory controller (called by 821ads.cfg)

To automatically configure SingleStep for use with the MPC821/860ADS board, append the contents of the 821ads.ini file to the end of your sstep.ini file. For target systems other than the MPC821/860ADS board you should create your own .cfg and .dbg files, using the SDS files as examples. Additional configuration files are also available on the SDS FTP site at ftp.sdsi.com in the directory download/board_support.

An alternate way of creating the _config alias is to use the Target Configuration tab in the "Debug" dialog box. The "Debug" dialog method and the sstep.ini method are mutually exclusive. Use one or the other, but not both.

Note: The "Debug" dialog method can not be used to set up the UPM memory controller. If the UPM is being used, you must use the sstep.ini method to define the _config alias. _config should invoke a .cfg file which in turn invokes a .dbg file which sets up the UPM.

Initialization of the target (that is, execution of the _config alias) will not actually occur until the "Debug" dialog is successfully exited.

5 Set up the download and execution options in the Options tab of the Debug dialog.

6 Download the application and run:

Select the File tab and enter the application file name. Exit the “Debug” dialog box by clicking OK.

Emulation module initialization and target initialization occur every time the “Debug” dialog is terminated via the OK button. A summary of the actions taken by SingleStep is given here:

- Initialize the emulation module with the communication speed specified in the "Debug" dialog.
- If “reset target” was selected then execute the commands specified by the _reset alias. The _reset alias should be used to specify commands that are specific to initializing the processor. It is executed each time the processor is reset. The value of the _reset alias can be viewed by issuing a “alias _reset” from the command window.
- Execute the commands specified by the _config alias. The _config alias should be used to specify commands that are specific to initializing (configuring) the target system. It is executed each time the processor is reset and each time the debug dialog is exited. The value of the _config alias can be viewed by issuing an “alias _config” from the command window.
- If "load image" was selected then download the application and set the PC based on object module file contents.
- If "execute until main" was selected then set a breakpoint at main() and run.

To send commands to the emulation module

To view commands sent by SingleStep

SingleStep communicates to the emulation module using the emulation module's "terminal interface" commands. SingleStep automatically generates and sends the commands required for normal operation. This communication between SingleStep and the emulation module can be observed by entering the following command in the SingleStep command window:

```
control -ms
```

To send commands

"Terminal interface" commands may be sent directly to the emulation module from the SingleStep command window or included in SingleStep's .cfg or .dbg command files.

Commands should be enclosed in double quotes and given the prefix: control -c.

Examples

To see the speed that the emulation module is using to communicate with the target system you would issue the following command in the SingleStep command window:

```
control -c "cf procck"
```

To change the speed to match a 25MHz processor clock you would issue the following command in the command window:

```
control -c "cf procck=25"
```

For more information about "terminal interface" commands see page 183.

Download performance

Downloads are fastest when the emulation module speed is set to match that of the target processor. The initial speed that the emulation module uses to communicate with the target processor is set by the Processor clock item in the connection tab of the “Debug” dialog. The user is responsible for specifying this speed to be less than or equal to the initial, reset, speed of the processor. Usually a command in the _config alias will raise the speed of the processor above its initial, reset value. For maximum download performance the command to increase the target processor speed should be followed by a command to increase the speed of the emulation module communication.

Example

The 821ads.cfg file contains the following command which sets the processor speed to 25MHz (assumes a 5 MHz crystal).

```
write -l 0xff000284 = 0x005000000
```

The following command, which increases the emulation module communication speed, should be placed immediately after the write command shown above.

```
control -c "cf procck=25"
```

On-chip breakpoints and debugging ROM code

The MPC821/860 has a built-in hardware breakpoint capability. When SingleStep steps one source line or sets a user defined breakpoint, it will first try to use a software breakpoint. If the breakpoint does not work because the breakpoint address is located in ROM, SingleStep will automatically attempt to use one of the available hardware breakpoints. For more information, see the SingleStep release notes.

To debug ROM based code, unselect “load application image” in the options tab of the “Debug” dialog.

Error conditions

"!ERROR 800! Invalid command: bcast" usually means that there is not a target interface module (TIM) connected to the emulation module or the emulation module does not have firmware for the MPC800 family. Verify that the emulation module is connected to the target. Next, go to the system window of the logic analyzer interface and verify that the Emulation Module icon (stop-light) is described as a Motorola MPC800 PowerPC Emulator. If it is not, follow

the steps on page 181 to update the firmware in the Emulation module for MPC800 processors.

"command socket connection failed: WSAECONNREFUSED: connection refused" usually means the emulation module is not at port #6470 on the Logic Analysis System. See step 2 of the getting started section above.

"unrecognized hostname" usually means that the debugger is unable to establish communication with the emulator. Verify communication to the emulation module by doing a ping to the logic analyzer. If you are unable to ping the logic analyzer refer to page 310 for more information.

See Also

The SDS web site: **http://www.sdsi.com**

The *SDS SingleStep Users Guide*.

The SingleStep to HP MPC 8xx product release notes.

The configuration section beginning on page 183 for more information on configuration options and the "cf" command.

10

Using the Analysis Probe and Emulation Module Together

This chapter describes how to use an analysis probe, an emulation module, and other features of your HP 16600A or HP 16700A logic analysis system to gain insight into your target system.

What are some of the tools I can use?

You can use a combination of all of the following tools to control and measure the behavior of your target system:

- Your analysis probe, to acquire data from the processor bus while it is running full-speed.
- Your emulation module, to control the execution of your target processor and to examine the state of the processor and of the target system.
- The Emulation Control Interface, to control and configure the emulation module, and to display or change target registers and memory.
- Display tools including the Listing tool, Chart tool, and System Performance Analyzer tool to make sense of the data collected using the analysis probe.
- Your debugger, to control your target system using the emulation module. Do not use the debugger at the same time as the Emulation Control Interface.
- The HP B4620B Source Correlation Tool Set, to relate the analysis trace to your high-level source code.

Which assembly-level listing should I use?

Several windows display assembly language instructions. Be careful to use to the correct window for your purposes:

- The Listing tool shows processor states that were captured during a "Run" of the logic analyzer. Those states are disassembled and displayed in the Listing window.
- The Emulation Control Interface shows the disassembled contents of a section of memory in the Memory Disassembly window.
- Your debugger shows your program as it was actually assembled, and (if it supports the emulation module) shows which line of assembly code corresponds to the value of the program counter on your target system.

Which source-level listing should I use?

Different tools display source code for different uses:

- The Source Viewer window allows you to follow how the processor executed code as the analyzer captured a trace. Use the Source Viewer to set analyzer triggers. The Source Viewer window is available only if you have licensed the HP B4620B Source Correlation Tool Set.
- Your debugger shows which line of code corresponds to the current value of the program counter on your target system. Use your debugger to set breakpoints.

Where can I find practical examples of measurements?

The Measurement Examples section in the online help contains examples of measurements which will save you time throughout the phases of system development: hardware turn-on, firmware development, software development, and system integration.

A few of the many things you can learn from the measurement examples are:

- How to find glitches.
- How to find NULL pointer de-references.
- How to profile system performance.

To find the measurement examples, click on the Help icon in the logic analysis system window, then click on "Measurement Examples."

Triggering the Emulation Module from the Analyzer

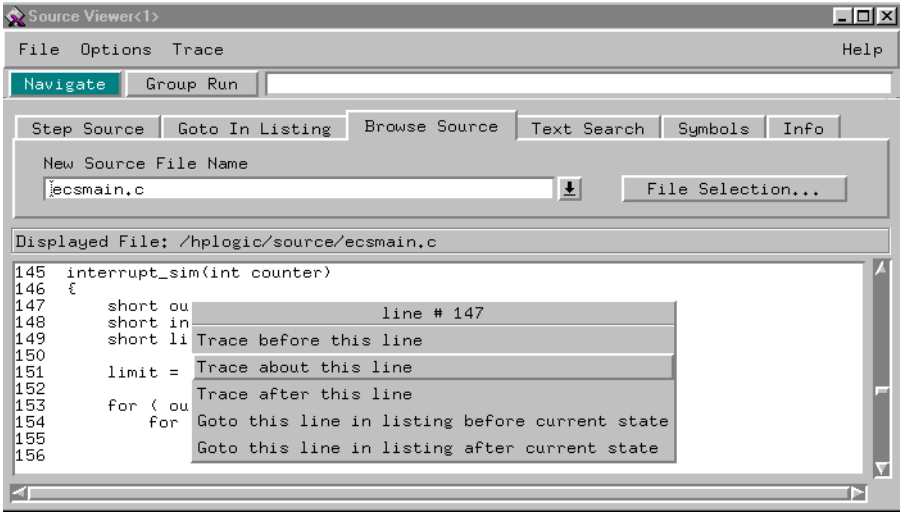
The logic analyzer may be used to signal the emulation module to stop (break) the target processor. This is done from either the Source Viewer window or the Intermodule window. If you are using the HP B4620B Source Correlation Tool Set, using the Source Viewer window is the easiest method.

To stop the processor when the logic analyzer triggers on a line of source code (Source Viewer window)

If you have the HP B4620B Source Correlation Tool Set, you can easily stop the processor when a particular line of code is reached.

1. Click on the logic analyzer module icon in the System window, and choose **Source Viewer...**.
2. In the Source Viewer window, click on the line of source code where you want to set the trigger, then select **Trace about this line**.

   The logic analyzer trigger is now set.



3 Select **Trace→Enable - Break Emulator On Trigger.**

The emulation module is now set to halt the processor after receiving a trigger from the logic analyzer.

To disable the processor stop on trigger, select **Trace→Disable - Break Emulator On Trigger.**

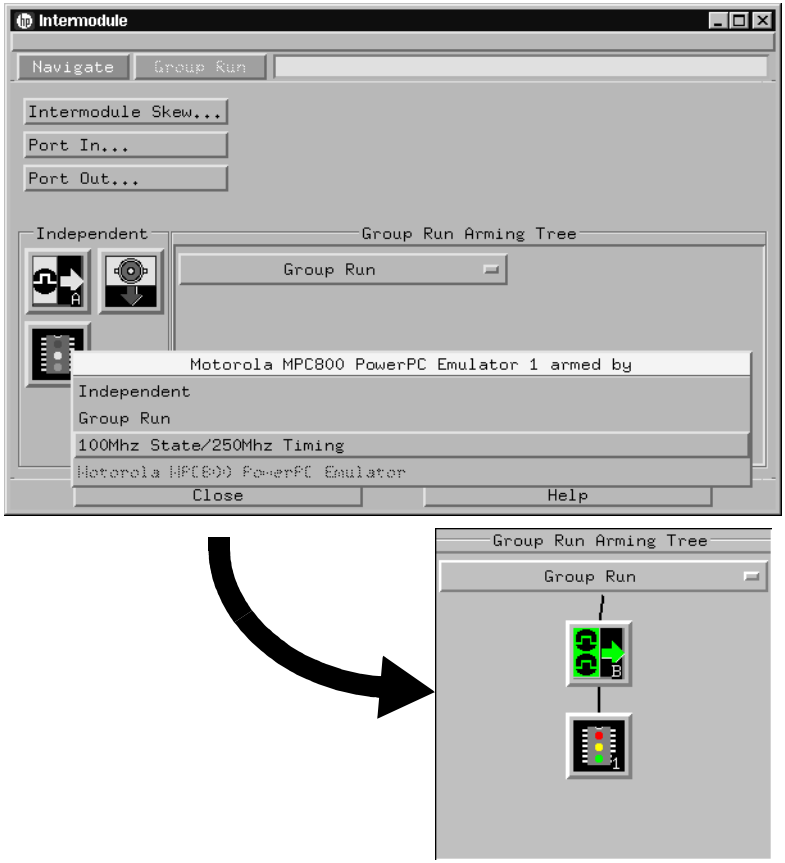
4 Click **Group Run** in the Source window (or other logic analyzer window).

5 If your target system is not already running, click **Run** in the emulation Run Control window to start your target.

To stop the processor when the logic analyzer triggers (Intermodule window)

Use the Intermodule window if you do not have the HP B4620B Source Correlation Tool Set or if you need to use a more sophisticated trigger than is possible in the Source Viewer window.

1. Create a logic analyzer trigger.
2. Click on the **Intermodule** icon in the System window.
3. In the Intermodule window, click the emulation module icon, then select the analyzer which is intended to trigger it.



The emulation module is now set to stop the processor when the logic analyzer triggers.

4 Click **Group Run** in the Source window (or other logic analyzer window).

5 If your target system is not already running, click **Run** in the emulation Run Control window to start your target.

See Also

See the online help for your logic analysis system for more information on setting triggers.

To minimize the "skid" effect

There is a finite amount of time between when the logic analyzer triggers, and when the processor actually stops. During this time, the processor will continue to execute instructions. This latency is referred to as the skid effect.

To minimize the skid effect:

1 In the Emulation Control Interface, open the Configuration window.

2 Set processor clock speed to the maximum value that your target can support.

The amount of skid will depend on the processor's execution speed and whether code is executing from the cache. See page 188 for information on how to configure the clock speed.

To stop the analyzer and view a measurement

- To view an analysis measurement you may have to click **Stop** after the trigger occurs.

NOTE: When the target processor stops it may cause the analyzer qualified clock to stop. Therefore most intermodule measurements will have to be stopped to see the measurement.

Example An intermodule measurement has been set up where the analyzer is triggering the emulation module. The following sequence could occur:

1. The analyzer triggers.
2. The trigger ("Break In") is sent to the emulation module.
3. The emulation module stops the user program which is running on the target processor. The processor enters a background debug monitor.
4. Because the processor has stopped, the analyzer stops receiving a qualified clock signal.
5. If the trigger position is "End", the measurement will be completed.
6. If the trigger position is not "End", the analyzer may continue waiting for more states.
7. The user clicks **Stop** in a logic analyzer window, which tells the logic analyzer to stop waiting, and to display the trace.

Tracing until the processor halts

If you are using a state analyzer, you can begin a trace, run the processor, then manually end the trace when the processor has halted.

To halt the processor, you can set a breakpoint using the Emulation Control Interface or a debugger.

Some possible uses for this measurement are:

- To store and display processor bus activity leading up to a system crash.
- To capture processor activity before a breakpoint.
- To determine why a function is being called. To do this, you could set a breakpoint at the start of the function then use this measurement to see how the function is getting called.

NOTE: This kind of measurement is easier than setting up an intermodule measurement trigger.

If you have already set up an intermodule measurement, you must "undo" it by setting all components in the intermodule window to run independently.

To capture a trace before the processor halts

HP 16600A/16700A

1 Set the sampling to **state mode** and the trigger condition to **Run until user stop**.

Now proceed to step 2 under "All HP logic analysis systems".

HP 1660/70 **HP 16500B/C**

1 In the configuration dialog, set the machine type to **state**, and set the logic analyzer to trigger on **nostate**.

Now proceed to step 2 under "All HP logic analysis systems".

All HP logic analysis systems

2 Set the logic analyzer to trigger on **nostate**.

3 Set the trigger point (position) to **End**.

4 In a logic analyzer window, click **Run**.

5 In the Emulation Control Interface or debugger click **Run**.

6 When the target processor halts, click **Stop** in the logic analyzer window to complete the measurement.

NOTE: This is the recommended method to do state analysis of the processor bus when the processor halts.

If you need to capture the interaction of another bus when the processor halts or you need to make a timing or oscilloscope measurement you will need to trigger the logic analyzer from the emulation module (described in the next section).

Triggering the Logic Analyzer from the Emulation Module

You can create an intermodule measurement which will allow the emulation module to trigger another module such as a timing analyzer or oscilloscope.

If you are only using a state analyzer to capture the processor bus then it will be much simpler to use “Tracing until processor halts” as described on page 238.

Before you trigger a logic analyzer (or another module) from the emulation module, you should understand a few things about the emulation module trigger:

The emulation module trigger signal

The trigger signal coming from the emulation module is an "In Background Debug Monitor" (In Monitor) signal. This may cause confusion because a variety of conditions could cause this signal and falsely trigger your analyzer.

The In Monitor trigger signal can be caused by:

- The most common method to generate the signal is to click **Run** and then click **Break** in the Emulation Control Interface. Going from Run (Running User Program) to Break (In Monitor) generates the trigger signal.
- Another method to generate the In Monitor signal is to click **Reset** and then click **Break**. Going from the reset state of the processor to the In Monitor state will generate the signal.
- In addition, an In Monitor signal is generated any time a debugger or other user interface reads a register, reads memory, sets breakpoints or steps. Care must be taken to not falsely trigger the logic analyzers that are listening to the In Monitor signal.

Group Run

The intermodule bus signals can still be active even without a Group Run.

The following setups can operate independently of Group Run:

- Port In connected to an emulation module
- Emulation modules connected in series
- Emulation module connected to Port Out

Here are some examples:

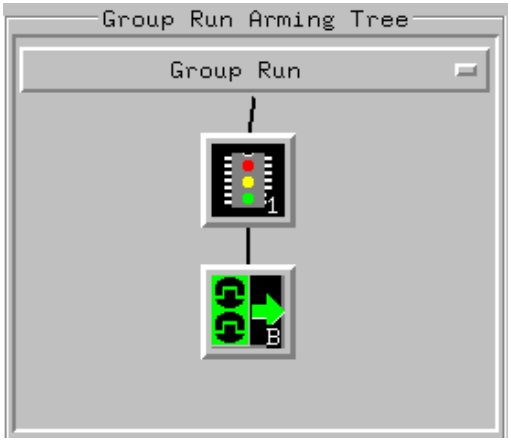
- If Group Run is armed from Port In and an emulation module is connected to Group Run, then any Port In signal will cause the emulation module to go into monitor. The Group Run button does not have to be clicked for this to operate.
- If two emulation modules are connected together so that one triggers another, then the first one going into monitor will cause the second one to go into monitor.
- If an emulation module is connected to Port Out, then the state of the emulation module will be sent out the Port Out without regard to Group Run.

The current emulation module state (Running or In Monitor) should be monitored closely when they are part of a Group Run measurement so that valid measurements are obtained.

Group Run into an emulation module does not mean that the Group Run will Run the emulation module.

The emulation module Run, Break, Step, and Reset are independent of the Group Run of the Analyzers.

For example, suppose you have the following intermodule measurement set up:



Clicking the **Group Run** button (at the very top of the Intermodule window or a logic analyzer window) will start the analyzer running. The analyzer will then wait for an arm signal. Now when the emulation module transitions into Monitor from Running (or from Reset), it will send the arm signal to the analyzer. If the emulation module is In Monitor when you click **Group Run**, you will then have to go to the emulation module or your debugger interface and manually start it running.

Debuggers can cause triggers

Emulation module user interfaces may introduce additional states into your analysis measurement and in some cases falsely trigger your analysis measurement.

When a debugger causes your target to break into monitor it will typically read memory around the program stack and around the current program counter. This will generate additional states that appear in the listing.

You can often distinguish these additional states because the time tags will be in the µs and ms range. You can use the time tag information to determine when the processor went into monitor. Typically the time between states will be in the nanoseconds while the processor is running and will be in the µs and ms range when the debugger has halted the processor and is reading memory.

Note also that some debugger commands may cause the processor to break temporarily to read registers and memory. These states that the debugger introduces will also show up in the trace listing.

If you define a trigger on some state and the debugger happens to read the same state, then you may falsely trigger your analyzer measurement.

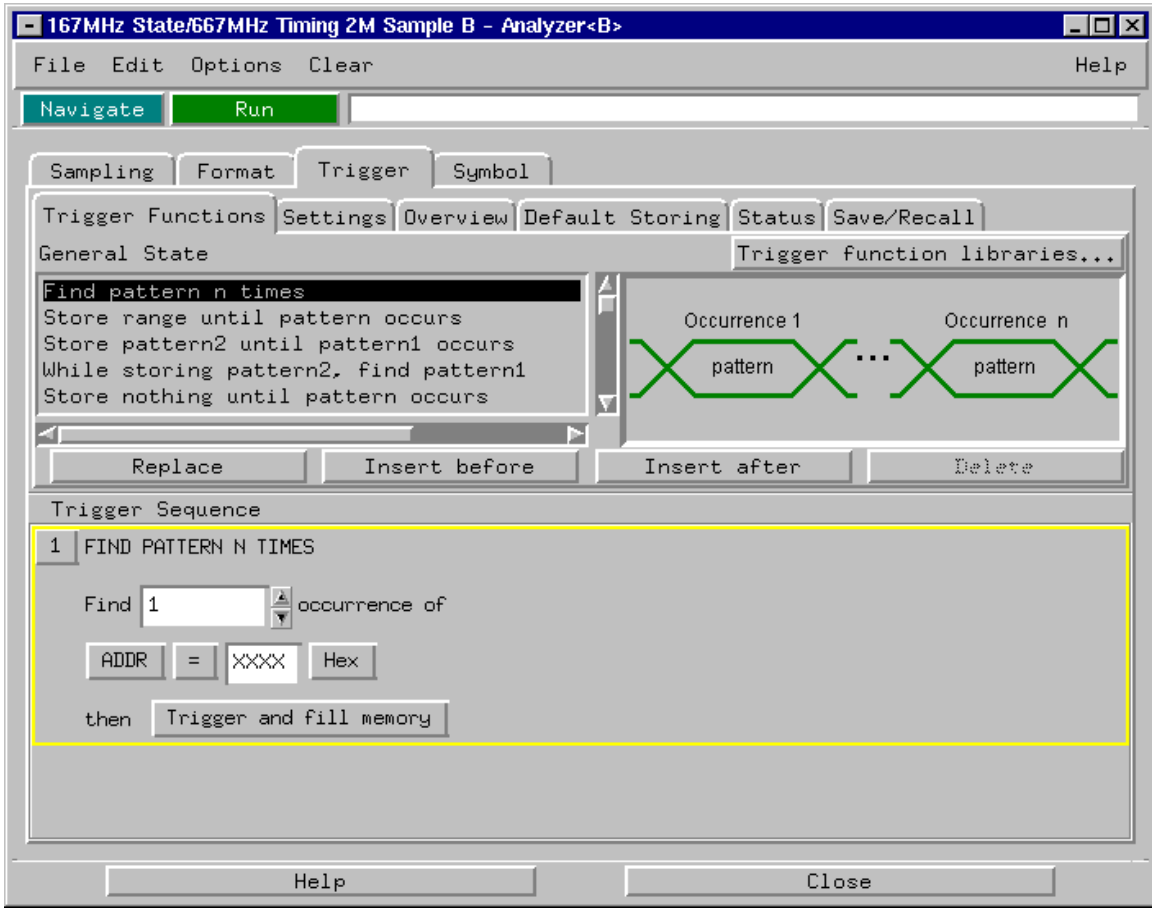
In summary, when you are making an analysis measurement be aware that the debugger could be impacting your measurement.

To trigger the analyzer when the processor halts

NOTE: If you are only using a state analyzer to capture the processor bus then it will be much simpler to use "Tracing until processor halts" as described on page 238.

HP 16600A/16700A with VisiTrigger

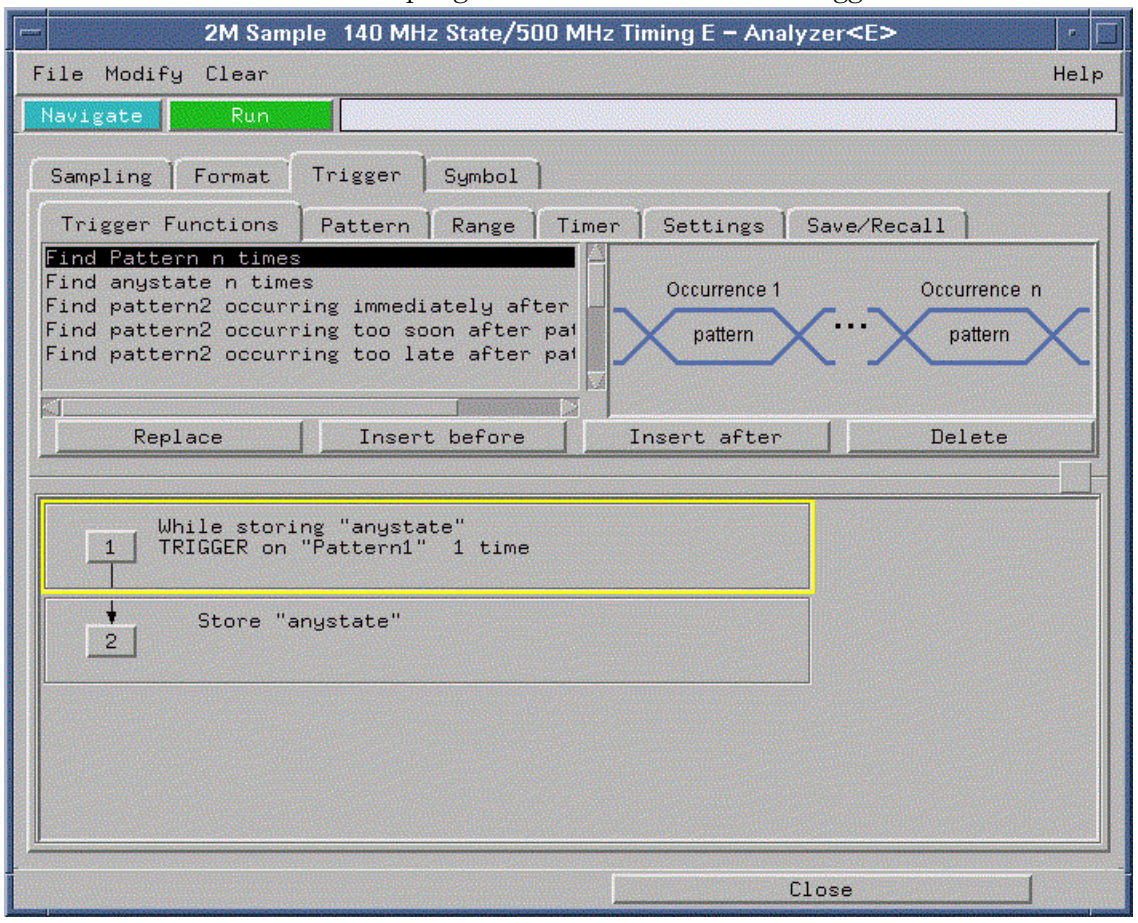
1 Set the sampling mode to state and set the trigger as shown below:



Now proceed to step 2 under "All HP logic analysis systems".

HP 16600A/16700A without VisiTrigger

1 Set the sampling mode to state and set the trigger as shown below:



Now proceed to step 2 under “All HP logic analysis systems”.

HP 1660/70 **HP 16500B/C**

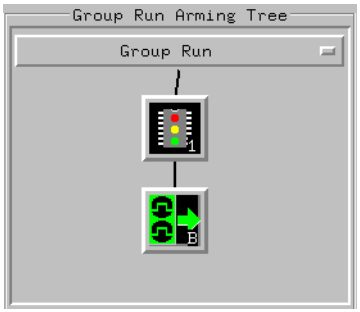
1 In the configuration dialog, set the machine type to **state**, and set the logic analyzer to trigger on **anystate**.

Now proceed to step 2 under "All HP logic analysis systems".

All HP logic analysis systems

2 Set the trigger position to **center** or **end**.

3 In the Intermodule window, click on the logic analyzer you want to trigger and select the emulation module. A picture (similar to the one shown below) will appear in the intermodule window.



The logic analyzer is now set to trigger on a processor halt.

4 Click **Group Run** to start the analyzer(s).

5 Click **Run** in the Emulation Control Interface or use your debugger to start the target processor running.

Clicking **Group Run** will *not* start the emulation module. The emulation module run, break, step, reset are independent of the Group Run of the analyzers.

6 Wait for the Run Control window in the Emulation Control Interface or the status display in your debugger to show that the processor has stopped.

The logic analyzer will store states up until the processor stops, but may continue running.

You may or may not see a "slow clock" error message. In fact, if you are using a

state analyzer on the processor bus the status may never change upon receiving the emulation module trigger (analysis arm). This occurs because the qualified processor clock needed to switch the state analyzer to the next state is stopped. For example, the state analyzer before the arm event may have a status of "**Occurrences Remaining in Level 1: 1**" and after the arm event it may have the same status of
"**Occurrences Remaining in Level 1: 1**"

7 If necessary, in the logic analyzer window, click **Stop** to complete the measurement.

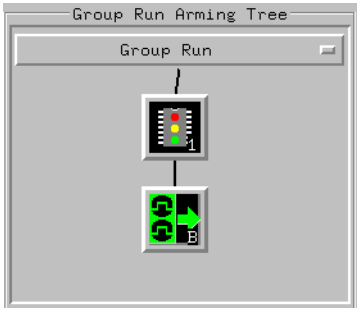
If you are using a timing analyzer or oscilloscope the measurement should complete automatically when the processor halts. If you are using a state analyzer, click **Stop** if needed to complete the measurement.

To trigger the analyzer when the processor reaches a breakpoint

This measurement is exactly like the one on the previous page, but with the one additional complexity of setting breakpoints. Be aware that setting breakpoints may cause a false trigger and that the breakpoints set may not be valid after a reset.

NOTE: If you are only using a state analyzer to capture the processor bus then it will be much simpler to use "Tracing until processor halts" as described on page 238.

1. Set the logic analyzer to trigger on **anystate**.
2. Set the trigger point to **center** or **end**.
3. In the Intermodule window, click on the logic analyzer you want to trigger and select the emulation module.



The logic analyzer is now set to trigger on a processor halt.

4. Set the breakpoint.

   If you are going to run the emulation module from Reset you must do a **Reset** followed by **Break** to properly set the breakpoints. The Reset will clear all on-chip hardware breakpoint registers. The Break command will then reinitialize the breakpoint registers. If you are using software breakpoints that insert an illegal instruction into your program at the breakpoint location you will not need to do the Reset, Break sequence. Instead you must take care to properly insert your software breakpoint in your RAM program location.

5. Click **Group Run** to start the analyzer(s).

6 Click **Run** in the Emulation Control Interface or use your debugger to start the target processor running.

Clicking **Group Run** will *not* start the emulation module. The emulation module run, break, step, reset are independent of the Group Run of the analyzers.

7 Wait for the Run Control window in the Emulation Control Interface or the status display in your debugger to show that the processor has stopped.

The logic analyzer will store states until the processor stops, but may continue running.

You may or may not see a "slow clock" error message. In fact, if you are using a state analyzer on the processor bus the status may never change upon receiving the emulation module trigger (analysis arm). This occurs because the qualified processor clock needed to switch the state analyzer to the next state is stopped. For example, the state analyzer before the arm event may have a status of "**Occurrences Remaining in Level 1: 1**" and after the arm event it may have the same status of "**Occurrences Remaining in Level 1: 1**"

8 If necessary, in the logic analyzer window, click **Stop** to complete the measurement.

If you are using a timing analyzer or oscilloscope the measurement should complete automatically when the processor halts. If you are using a state analyzer, click **Stop** if needed to complete the measurement.

11

Hardware Reference

This chapter contains additional reference information including the specifications and characteristics for the analysis probe and the emulation probe, as well as signal mapping for the HP E2476A analysis probe and HP E2477A software. It consists of the following information:

- Analysis probe reference
- Emulation module reference

Analysis probe—operating characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the HP E2476A MPC860/821 analysis probe.

Operating Characteristics	
Microprocessor Compatibility	Motorola MPC860 and MPC821, versions DC, DE, DH, EN, MH, and SAR.
Package Supported	357-pin BGA
Microprocessor Clock Speed	50 MHz maximum
Logic Analyzers Supported	HP 1660A/AS/C/CS/CP/E/ES/EP, HP 1661A/AS/C/CS/CP/E/ES/EP, HP 1670A/D/E, HP 1671A/D/E, HP 16550A (one or two cards), HP 16554A/55A/56A (two or three cards), HP 16555D/56D/57D (two or three cards), HP 16600/01/02A, HP 16710/11/12A (one or two cards), HP 16715/16/17A (2 or 3 cards).
Accessories Required	For state and timing analysis, the HP E5355A Probing Kit and the HP E5346A High-density Cables are required (included with the HP E2476A).
Optional Accessories	An emulation module can be connected to the analysis probe.
Pods Required	Six 16-channel pods are required for disassembly of MPC821/860. Connectors for six additional 16-channel pods are available.
Electrical Characteristics	
Power Requirements	100 mA @ 5V, supplied by the logic analyzer. CAT I, Pollution degree 2. Approximately 0.1 µF decoupling on VDDH, VDD, VDDSYN, and KAPWR. Maximum draw of 2 mA from target system VDD @ 3.3 V.
Signal Line Loading	Approximately 25 pF on $\overline{\text{SRESET}}$, $\overline{\text{HRESET}}$, DSDI, DSDO, and DSCK. Approximately 15 pF on TMS and $\overline{\text{TRST}}$. Approximately 10 pF on all other signals.
Propagation Delays	Approximately 1.5 ns on DSDI, DSDO, and DSCK

Analysis probe—environmental characteristics

Environmental Characteristics	
Temperature	Operating: 0 to +50 degrees C
Altitude	Operating: 4,600 m
Humidity	Up to 75% non-condensing. Avoid sudden, extreme temperature changes which could cause condensation on the circuit board. For indoor use only.

Analysis probe—signal-to-connector mapping

The following tables show the electrical signal-to-connector mapping required by the HP E2476A MPC821/860 Analysis Probe and the HP E2477A Inverse Assembler/Execution Tracker Software.

MPC860/821 Logic Analyzer Interface Signal List - Pod J1odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J1odd	6	clk1	C9	TSIZ1	STAT
J1odd	8	15	C13	A16	ADDR
J1odd	10	14	B13	A17	ADDR
J1odd	12	13	D9	A18	ADDR
J1odd	14	12	D11	A19	ADDR
J1odd	16	11	C12	A20	ADDR
J1odd	18	10	B12	A21	ADDR
J1odd	20	9	B10	A22	ADDR
J1odd	22	8	B11	A23	ADDR
J1odd	24	7	C11	A24	ADDR
J1odd	26	6	D10	A25	ADDR
J1odd	28	5	C10	A26	ADDR
J1odd	30	4	A13	A27	ADDR
J1odd	32	3	A10	A28	ADDR
J1odd	34	2	A12	A29	ADDR
J1odd	36	1	A11	A30	ADDR
J1odd	38	0	A9	A31	ADDR

MPC860/821 Logic Analyzer Interface Signal List - Pod J1even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J1even	5	clk1	B9	TSIZ0/$\overline{\text{REG}}$	STAT
J1even	7	15	B19	A0 (MSB)	ADDR
J1even	9	14	B18	A1	ADDR
J1even	11	13	A18	A2	ADDR
J1even	13	12	C16	A3	ADDR
J1even	15	11	B17	A4	ADDR
J1even	17	10	A17	A5	ADDR
J1even	19	9	B16	A6	ADDR
J1even	21	8	A16	A7	ADDR
J1even	23	7	D15	A8	ADDR
J1even	25	6	C15	A9	ADDR
J1even	27	5	B15	A10	ADDR
J1even	29	4	A15	A11	ADDR
J1even	31	3	C14	A12	ADDR
J1even	33	2	B14	A13	ADDR
J1even	35	1	A14	A14	ADDR
J1even	37	0	D12	A15	ADDR

Active-low signals, such as $\overline{\text{SRESET}}$, are shown with an overscore.

MPC860/821 Logic Analyzer Interface Signal List - Pod J2odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J2odd	6	clk1	L1	$\overline{STS}$	STAT
J2odd	8	15	U10	D16	DATA
J2odd	10	14	T12	D17	DATA
J2odd	12	13	V9	D18	DATA
J2odd	14	12	U9	D19	DATA
J2odd	16	11	V8	D20	DATA
J2odd	18	10	U8	D21	DATA
J2odd	20	9	T9	D22	DATA
J2odd	22	8	U12	D23	DATA
J2odd	24	7	V7	D24	DATA
J2odd	26	6	T8	D25	DATA
J2odd	28	5	U7	D26	DATA
J2odd	30	4	V12	D27	DATA
J2odd	32	3	V6	D28	DATA
J2odd	34	2	W5	D29	DATA
J2odd	36	1	U6	D30	DATA
J2odd	38	0	T7	D31 (LSB)	DATA

MPC860/821 Logic Analyzer Interface Signal List - Pod J2even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J2even	5	clk1	W3	CLKOUT	STAT
J2even	7	15	W14	D0 (MSB)	DATA
J2even	9	14	W12	D1	DATA
J2even	11	13	W11	D2	DATA
J2even	13	12	W10	D3	DATA
J2even	15	11	W13	D4	DATA
J2even	17	10	W9	D5	DATA
J2even	19	9	W7	D6	DATA
J2even	21	8	W6	D7	DATA
J2even	23	7	U13	D8	DATA
J2even	25	6	T11	D9	DATA
J2even	27	5	V11	D10	DATA
J2even	29	4	U11	D11	DATA
J2even	31	3	T13	D12	DATA
J2even	33	2	V13	D13	DATA
J2even	35	1	V10	D14	DATA
J2even	37	0	T10	D15	DATA

MPC860/821 Logic Analyzer Interface Signal List - Pod J3odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J3odd	6	clk1	D1	$\overline{\text{TEA}}$	STAT
J3odd	8	15	C3	$\overline{\text{CS0}}$	
J3odd	10	14	A2	$\overline{\text{CS1}}$	
J3odd	12	13	D4	$\overline{\text{CS2}}$	
J3odd	14	12	E4	$\overline{\text{CS3}}$	
J3odd	16	11	A4	$\overline{\text{CS4}}$	
J3odd	18	10	B4	$\overline{\text{CS5}}$	
J3odd	20	9	D5	$\overline{\text{CS6}}$/$\overline{\text{CE1\_B}}$	
J3odd	22	8	C4	$\overline{\text{CS7}}$/$\overline{\text{CE2\_B}}$	
J3odd	24	7	D8	$\overline{\text{BS\_A0}}$	
J3odd	26	6	C8	$\overline{\text{BS\_A1}}$	
J3odd	28	5	A7	$\overline{\text{BS\_A2}}$	
J3odd	30	4	B8	$\overline{\text{BS\_A3}}$	
J3odd	32	3	C7	$\overline{\text{WE0}}$/$\overline{\text{BS\_B0}}$/$\overline{\text{IORD}}$	
J3odd	34	2	A6	$\overline{\text{WE1}}$/$\overline{\text{BS\_B1}}$/$\overline{\text{IOWR}}$	
J3odd	36	1	B6	$\overline{\text{WE2}}$/$\overline{\text{BS\_B2}}$/$\overline{\text{PCOE}}$	
J3odd	38	0	A5	$\overline{\text{WE3}}$/$\overline{\text{BS\_B3}}$/$\overline{\text{PCWE}}$	

MPC860/821 Logic Analyzer Interface Signal List - Pod J3even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J3even	5	clk1	C2	LTA	STAT
J3even	7	15	H2	VFLS0/IP_B0/IWP0	STAT
J3even	9	14	J3	VFLS1/IP_B1/IWP1	STAT
J3even	11	13	J2	IPP_B2/$\overline{\text{IOIS16\_B}}$/AT2	STAT
J3even	13	12	G1	IPP_B3/IWP2/VF2	STAT
J3even	15	11	G2	IP_B4/LWP0/VF0	STAT
J3even	17	10	J4	IP_B5/LWP1/VF1	STAT
J3even	19	9	K3	IP_B6/DSDI/AT0	STAT
J3even	21	8	H1	IP_B7//PTR/AT3	STAT
J3even	23	7	E2	$\overline{\text{BG}}$	STAT
J3even	25	6	E1	$\overline{\text{BB}}$	STAT
J3even	27	5	G4	$\overline{\text{BR}}$	STAT
J3even	29	4	E3	$\overline{\text{BI}}$	STAT
J3even	31	3	D2	$\overline{\text{BDIP}}$/$\overline{\text{GPLB5}}$	STAT
J3even	33	2	F1	$\overline{\text{BURST}}$	STAT
J3even	35	1	B2	RD/$\overline{\text{WR}}$	STAT
J3even	37	0	F3	$\overline{\text{TS}}$	STAT

MPC860/821 Logic Analyzer Interface Signal List - Pod J4odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J4odd	6	clk1	K2	ALEA	
J4odd	8	15	H3	$\overline{\text{RSV}}$/$\overline{\text{IRQ2}}$	
J4odd	10	14	F2	$\overline{\text{CR}}$/$\overline{\text{IRQ3}}$	
J4odd	12	13	V3	DP0/$\overline{\text{IRQ3}}$	
J4odd	14	12	V5	DP1/$\overline{\text{IRQ4}}$	
J4odd	16	11	W4	DP2/$\overline{\text{IRQ5}}$	
J4odd	18	10	V4	DP3/$\overline{\text{IRQ6}}$	
J4odd	20	9	G3	FRZ/$\overline{\text{IRQ6}}$	
J4odd	22	8	K1	$\overline{\text{KR}}$/$\overline{\text{IRQ4}}$/SPKROUT	
J4odd	24	7	T5	IP_A0	
J4odd	26	6	T4	IP_A1	
J4odd	28	5	U3	IP_A2/$\overline{\text{IOIS16\_A}}$	
J4odd	30	4	W2	IP_A3	
J4odd	32	3	U4	IP_A4	
J4odd	34	2	U5	IP_A5	
J4odd	36	1	T6	IP_A6	
J4odd	38	0	T3	IP_A7	

MPC860/821 Logic Analyzer Interface Signal List - Pod J4even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J4even	5	clk1	N2	EXTCLK	
J4even	7	15			
J4even	9	14	J1	ALEB/DSCK/AT1	
J4even	11	13	N3	TEXP	
J4even	13	12	L4	OP0	
J4even	15	11	L2	OP1	
J4even	17	10	L1	OP2/MODCK1/$\overline{\text{STS}}$	
J4even	19	9	M4	OP3/MODCK2/DSDO	
J4even	21	8	R2	$\overline{\text{PORESET}}$	
J4even	23	7	P3	$\overline{\text{RSTCONF}}$	
J4even	25	6	N4	$\overline{\text{HRESET}}$	
J4even	27	5	P2	$\overline{\text{SRESET}}$	
J4even	29	4	R4	$\overline{\text{WAIT\_B}}$	
J4even	31	3	R3	$\overline{\text{WAIT\_A}}$	
J4even	33	2	C1	$\overline{\text{UPWAITA}}$/$\overline{\text{GPLA4}}$	
J4even	35	1	B1	$\overline{\text{UPPWAITB}}$/$\overline{\text{GPLB4}}$	
J4even	37	0	B3	$\overline{\text{CE1\_A}}$	

MPC860/821 Logic Analyzer Interface Signal List - Pod J5odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J5odd	6	clk1	A3	$\overline{\text{CE2\_A}}$	
J5odd	8	15	U19	PA0/CLK8/$\overline{\text{TOUT4}}$/L1TCLKB	
J5odd	10	14	T19	PA1/CLK7/TIN4/BRGOUT4	
J5odd	12	13	R18	PA2/CLK6/$\overline{\text{TOUT3}}$/L1RCLKB/ BRGCLK2	
J5odd	14	12	P17	PA3/CLK5/TIN3/BRGOUT3	
J5odd	16	11	P19	PA4/CLK4/$\overline{\text{TOUT2}}$	
J5odd	18	10	N18	PA5/CLK3/TIN2/L1TCLKA/ BRGOUT2	
J5odd	20	9	M17	PA6/CLK2/$\overline{\text{TOUT1}}$/BRGCLK1	
J5odd	22	8	M19	PA7/CLK1/TIN1/L1RCLKA/ BRGOYT1	
J5odd	24	7	L17	PA8/L1RXDA	
J5odd	26	6	K18	PA9/L1TXDA	
J5odd	28	5	J17	PA10/L1RXDB	
J5odd	30	4	G16	PA11/L1TXDB	
J5odd	32	3	F17	PA12/TXD2	
J5odd	34	2	E17	PA13/RXD2	
J5odd	36	1	D17	PA14/TXD1	
J5odd	38	0	C18	PA15/RXD1	

MPC860/821 Logic Analyzer Interface Signal List - Pod J5even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J5even	5	clk1	D7	$\overline{\text{GPLA0}}$/$\overline{\text{GPLB00}}$	
J5even	7	15	D3	$\overline{\text{GPLA5}}$	
J5even	9	14	C5	$\overline{\text{GPLA3}}$/$\overline{\text{GPLB3}}$/$\overline{\text{CS3}}$	
J5even	11	13	B5	$\overline{\text{GPLA2}}$/$\overline{\text{GPLB2}}$/$\overline{\text{CS2}}$	
J5even	13	12	C6	$\overline{\text{OE}}$/$\overline{\text{GPLA1}}$/$\overline{\text{GPLB1}}$	
J5even	15	11	T17	PC4/L1RSYNCA/$\overline{\text{CD4}}$	
J5even	17	10	T18	PC5/L1TSYNCA/$\overline{\text{SDACK1}}$/$\overline{\text{CTS4}}$	
J5even	19	9	R19	PC6/L1RSYNCB/$\overline{\text{CD3}}$	
J5even	21	8	M16	PC7/L1TSYNCB/$\overline{\text{SDACK2}}$/$\overline{\text{CTS3}}$	
J5even	23	7	M18	PC8/$\overline{\text{CD2}}$/$\overline{\text{TGATE2}}$	
J5even	25	6	L18	PC9/$\overline{\text{CTS2}}$	
J5even	27	5	K19	PC10/$\overline{\text{CD1}}$/$\overline{\text{TGATE1}}$	
J5even	29	4	J19	PC11/$\overline{\text{CTS1}}$	
J5even	31	3	F18	PC12//L1RQA/LIST4	
J5even	33	2	E18	PC13/L1RQB/LIST3	
J5even	35	1	D18	PC14/$\overline{\text{DREQ2}}$/$\overline{\text{RTS2}}$/LIST2	
J5even	37	0	D16	PC15/$\overline{\text{DREQ1}}$/RTS1/LIST1	

MPC860/821 Logic Analyzer Interface Signal List - Pod J6odd

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J6odd	6	clk1	W15	$\overline{\text{IRQ7}}$	
J6odd	8	15	N16	PB16/L1RQA/LIST4	
J6odd	10	14	P18	PB17/L1RQB/LIST3	
J6odd	12	13	N17	PPB18/$\overline{\text{RST2}}$/LIST2	
J6odd	14	12	N19	PB19/$\overline{\text{RST1}}$/LIST1	
J6odd	16	11	L16	PB20/SMRXD2/L1CLKOA	
J6odd	18	10	K16	PB21/SMTXD2/L1CLKOB	
J6odd	20	9	L19	PB22/$\overline{\text{SMSYN2}}$/$\overline{\text{SDACK2}}$	
J6odd	22	8	K17	PB23/$\overline{\text{SMSYN1}}$/$\overline{\text{SDACK1}}$	
J6odd	24	7	J18	PB24/SMRXD1	
J6odd	26	6	J16	PB25/SMTXD1	
J6odd	28	5	F19	PB26/I2CSCL/BRGOUT2	
J6odd	30	4	E19	PPB27/I2CSDA/BRGOUT1	
J6odd	32	3	D19	PB28/SPIMISO/BRGOUT4	
J6odd	34	2	E16	PB29/SPIMOSI	
J6odd	36	1	C19	PB30/SPICLK	
J6odd	38	0	C17	PB31/$\overline{\text{SPISEL}}$/$\overline{\text{RRJECT1}}$	

MPC860/821 Logic Analyzer Interface Signal List - Pod J6even

Cable Connector	2x19 pin	LA bit	860/821 Pin	860/821 Signal	Label
J6even	5	clk1	U14	$\overline{\text{IRQ1}}$	
J6even	7	15	V14	$\overline{\text{IRQ0}}$	
J6even	9	14	W16	PD3/SHIFT_CLK/$\overline{\text{RRJECT4}}$	
J6even	11	13	U16	PD4/LOAD_HSYNC/$\overline{\text{RRJECT3}}$	
J6even	13	12	U15	PD5/FRAME_VSYNC/$\overline{\text{RRJECT2}}$	
J6even	15	11	V16	PD6/LCD_AC/LOE/$\overline{\text{RTS4}}$	
J6even	17	10	T15	PD7/LD0/$\overline{\text{RTS4}}$	
J6even	19	9	W17	PD8/LD1/TXD4	
J6even	21	8	V17	PD9/LD2/RXD4	
J6even	23	7	W18	PD10/LD3/TXD3	
J6even	25	6	T16	PD11/LD4/RXDD3	
J6even	27	5	R16	PD12/LD5/L1RSYNCB	
J6even	29	4	V18	PD13/LD6/L1TSYNCB	
J6even	31	3	V19	PD14/LD7/L1RSYNCA	
J6even	33	2	U17	PD15/LD8/L1TSYNCA	
J6even	35	1	U18	PB14/$\overline{\text{RSTRT1}}$	
J6even	37	0	R17	PB15/BRGOUT3	

Emulation module—operating characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the HP 16610A emulation module and MPC821/860 target interface module.

Operating Characteristics	
Microprocessor Compatibility	Motorola MPC860 (and derivatives), MPC821, MPC801, and MPC850 Embedded PowerPC microprocessors.
Environmental Characteristics (Temperature, Altitude, Humidity)	The HP 16610A emulation module meets the environmental characteristics of the logic analysis system in which it is installed. For indoor use only.

Emulation module—electrical characteristics

Maximum Ratings

Characteristics for the MPC800 Embedded PowerPC emulation module	Symbol	Min	Max	Unit
Input voltage range	V_{in}	-0.5	5.5	V
Input voltage range (Vtt)		1.3	1. 7	V
Input High Voltage	V_{ih}	2 / 3 V_{tt} + 0.2		V
Input Low Voltage	V_{il}		2 / 3 V_{tt} - 0.2	V
Input High Current	I_{ih}		-15	µA
Input Low Current	I_{il}		100	µA
Output High Voltage	V_{oh}	2.4	3.3	V
Output Low Voltage	V_{ol}		0.5	V
Output High Current	I_{oh}	8		mA
Output Low Current	I_{ol}	-16		mA

12

General-Purpose ASCII (GPA) Symbol File Format

General-purpose ASCII (GPA) format files are loaded into a logic analyzer just like other object files, but they are usually created differently.

If your compiler is not one of those listed on page 153, if your compiler does not include symbol information in the output, or if you want to define a symbol not in the object file, you can create an ASCII format symbol file.

Typically, ASCII format symbol files are created using text processing tools to convert compiler or linker map file output that has symbolic information into the proper format.

You can typically get symbol table information from a linker map file to create a General-Purpose ASCII (GPA) symbol file.

Various kinds of symbols are defined in different records in the GPA file. Record headers are enclosed in square brackets; for example, [VARIABLES]. For a summary of GPA file records and associated symbol definition syntax, refer to the "GPA Record Format Summary" that follows.

Each entry in the symbol file must consist of a symbol name followed by an address or address range.

While symbol names can be very long, the logic analyzer only uses the first 16 characters.

The address or address range corresponding to a given symbol appears as a hexadecimal number. The address or address range must immediately follow the symbol name, appear on the same line, and be separated from the symbol name by one or more blank spaces or tabs. Ensure that address ranges are in the following format:

```
beginning address..ending address
```

Example

```
main     00001000..00001009
 test     00001010..0000101F
 var1     00001E22     #this is a variable
```

This example defines two symbols that correspond to address ranges and one point symbol that corresponds to a single address.

For more detailed descriptions of GPA file records and associated symbol definition syntax, refer to these topics that follow:

- SECTIONS
- FUNCTIONS
- VARIABLES
- SOURCE LINES
- START ADDRESS
- Comments

GPA Record Format Summary

Format

```
[SECTIONS]
 section_name  start..end  attribute

 [FUNCTIONS]
 func_name  start..end

 [VARIABLES]
 var_name   start [size]
 var_name   start..end

 [SOURCE LINES]
 File: file_name
 line#  address

 [START ADDRESS]
 address

 #Comments
```

If no record header is specified, [VARIABLES] is assumed. Lines without a preceding header are assumed to be symbol definitions in one of the VARIABLES formats.

Example

This is an example GPA file that contains several different kinds of records:

```
[SECTIONS]
 prog     00001000..0000101F
 data     40002000..40009FFF
 common   FFFF0000..FFFF1000

 [FUNCTIONS]
 main     00001000..00001009
 test     00001010..0000101F

 [VARIABLES]
 total    40002000  4
 value    40008000  4

[SOURCE LINES]
 File: main.c
 10       00001000
 11       00001002
 14       0000100A
 22       0000101E

 File: test.c
  5       00001010
  7       00001012
 11       0000101A
```

SECTIONS

Format

```
[SECTIONS]
 section_name  start..end  attribute
```

Use SECTIONS to define symbols for regions of memory, such as sections, segments, or classes.

`section_name` A symbol representing the name of the section.

`start` The first address of the section, in hexadecimal.

`end` The last address of the section, in hexadecimal.

`attribute` This is optional, and may be one of the following:

NORMAL (default)—The section is a normal, relocatable section, such as code or data.

NONRELOC—The section contains variables or code that cannot be relocated; this is an absolute segment.

> **Enable Section Relocation**
>
> To enable section relocation, section definitions must appear before any other definitions in the file.

Example

```
[SECTIONS]
 prog           00001000..00001FFF
 data           00002000..00003FFF
 display_io     00008000..0000801F  NONRELOC
```

If you use section definitions in a GPA symbol file, any subsequent function or variable definitions must be within the address ranges of one of the defined sections. Functions and variables that are not within the range are ignored.

FUNCTIONS

Format

```
[FUNCTIONS]
 func_name  start..end
```

Use FUNCTIONS to define symbols for program functions, procedures, or subroutines.

`func_name` A symbol representing the function name.

`start` The first address of the function, in hexadecimal.

`end` The last address of the function, in hexadecimal.

Example

```
[FUNCTIONS]
 main     00001000..00001009
 test     00001010..0000101F
```

VARIABLES

Format

```
[VARIABLES]
 var_name   start [size]
 var_name   start..end
```

You can specify symbols for variables either by using the address of the variable, the address and the size of the variable, or a range of addresses occupied by the variable. If you specify only the address of a variable, the size is assumed to be one byte.

`var_name` A symbol representing the variable name.

`start` The first address of the variable, in hexadecimal.

`end` The last address of the variable, in hexadecimal.

`size` This is optional, and indicates the size of the variable, in bytes, in decimal.

Example

```
[VARIABLES]
 subtotal     40002000   4
 total        40002004   4
 data_array   40003000..4000302F
 status_char  40002345
```

SOURCE LINES

Format

```
[SOURCE LINES]
 File: file_name
 line#  address
```

Use SOURCE LINES to associate addresses with lines in your source files.

`file_name` The name of a file.

`line#` The number of a line in the file, in decimal.

`address` The address of the source line, in hexadecimal.

Example

```
[SOURCE LINES]
 File: main.c
 10       00001000
 11       00001002
 14       0000100A
 22       0000101E
```

START ADDRESS

Format

```
[START ADDRESS]
 address
```

`address` The address of the program entry point, in hexadecimal.

Example

```
[START ADDRESS]
 00001000
```

Comments

Format

```
#comment text
```

Use the # character to include comments in a file. Any text following the # character is ignored. You can put comments on a line alone or on the same line following a symbol entry.

Example

```
 #This is a comment.
```

13

Troubleshooting the Analysis Probe

If you encounter difficulties while making measurements, use this chapter to guide you through some possible solutions. Each heading lists a problem you may encounter, along with some possible solutions.

If you still have difficulty using the analyzer after trying the suggestions in this chapter, please contact your local Hewlett-Packard service center.

CAUTION: When you are working with the analyzer, be sure to power down both the analyzer and the target system before disconnecting or connecting cables, probes, and analysis probes. Otherwise, you may damage circuitry in the analyzer, analysis probe, or target system.

Logic Analyzer Problems

This section lists general problems that you might encounter while using the logic analyzer.

Intermittent data errors

This problem is usually caused by poor connections, incorrect signal levels, or marginal timing.

- ❏ Remove and re-seat all cables and probes, ensuring that there are no bent pins on the analysis probe interface or poor probe connections.
- ❏ Adjust the threshold level of the data pod to match the logic levels in the system under test.
- ❏ Use an oscilloscope to check the signal integrity of the data lines.

  Clock signals for the state analyzer must meet particular pulse shape and timing requirements. Data inputs for the analyzer must meet pulse shape and setup and hold time requirements.

See Also

See “Capacitive Loading” in this chapter for information on other sources of intermittent data errors.

Unwanted triggers

Unwanted triggers can be caused by instructions that were fetched but not executed.

❏ Add the prefetch queue or pipeline depth to the trigger address to avoid this problem.

The logic analyzer captures prefetches, even if they are not executed. When you are specifying a trigger condition or a storage qualification that follows an instruction that may cause branching, an unused prefetch may generate an unwanted trigger.

No activity on activity indicators

❏ Check for loose cables, board connections, and analysis probe interface connections.

❏ Check for bent or damaged pins on the analysis probe.

No trace list display

If there is no trace list display, it may be that your trigger specification is not correct for the data you want to capture, or that the trace memory is only partially filled.

- ❏ Check your trigger sequence to ensure that it will capture the events of interest.
- ❏ Try stopping the analyzer; if the trace list is partially filled, this should display the contents of trace memory.

Analyzer won't power up

If logic analyzer power is cycled when the logic analyzer is connected to a target system or emulation probe that remains powered up, the logic analyzer may not be able to power up. Some logic analyzers are inhibited from powering up when they are connected to a target system or emulation probe that is already powered up.

- ❏ Remove power from the target system, then disconnect all logic analyzer cabling from the analysis probe. This will allow the logic analyzer to power up. Reconnect logic analyzer cabling after power up.

Analysis Probe Problems

This section lists problems that you might encounter when using an analysis probe. If the solutions suggested here do not correct the problem, you may have a damaged analysis probe. Contact your local Hewlett-Packard Sales Office if you need further assistance.

Target system will not boot up

If the target system will not boot up after connecting the analysis probe interface, the microprocessor (if socketed) or the analysis probe interface may not be installed properly, or they may not be making electrical contact.

❏ Ensure that you are following the correct power-on sequence for the analysis probe and target system.

a Power up the analyzer and analysis probe.

b Power up the target system.

If you power up the target system before you power up the analysis probe, interface circuitry in the analysis probe may latch up and prevent proper target system operation.

❏ Verify that the microprocessor and the analysis probe are properly rotated and aligned, so that the index pin on the microprocessor (pin A1) matches the index pin on the analysis probe interface.

❏ Verify that the microprocessor and the analysis probe interface are securely inserted into their respective sockets.

❏ Verify that the logic analyzer cables are in the proper sockets of the analysis probe interface and are firmly inserted.

Erratic trace measurements

- ❏ Do a full reset of the target system before beginning the measurement.

  Some analysis probe designs require a full reset to ensure correct configuration.

- ❏ Ensure that your target system meets the timing requirements of the processor with the analysis probe installed.

  See “Capacitive loading” in this chapter. While analysis probe loading is slight, pin protectors, extenders, and adapters may increase it to unacceptable levels. If the target system design has close timing margins, such loading may cause incorrect processor functioning and give erratic trace results.

- ❏ Ensure that you have sufficient cooling for the microprocessor.

  Ensure that you have ambient temperature conditions and air flow that meet or exceed the requirements of the microprocessor manufacturer.

Capacitive loading

Excessive capacitive loading can degrade signals, resulting in incorrect capture by the analysis probe interface, or system lockup in the microprocessor. All analysis probe interfaces add additional capacitive loading, as can custom probe fixtures you design for your application.

Careful layout of your target system can minimize loading problems and result in better margins for your design. This is especially important for systems that are running at frequencies greater than 50 MHz.

- ❏ Remove as many pin protectors, extenders, and adapters as possible.
- ❏ If multiple analysis probe interface solutions are available, use one with lower capacitive loading.

Inverse Assembler Problems

This section lists problems that you might encounter while using the inverse assembler.

When you obtain incorrect inverse assembly results, it may be unclear whether the problem is in the analysis probe or in your target system. If you follow the suggestions in this section to ensure that you are using the analysis probe and inverse assembler correctly, you can proceed with confidence in debugging your target system.

No inverse assembly or incorrect inverse assembly

This problem may be due to incorrect synchronization, modified configuration, incorrect connections, or a hardware problem in the target system. A locked status line can cause incorrect or incomplete inverse assembly.

- ❏ Ensure that each logic analyzer pod is connected to the correct analysis probe connector.

  There is not always a one-to-one correspondence between analyzer pod numbers and analysis probe cable numbers. Analysis Probes must supply address (ADDR), data (DATA), and status (STAT) information to the analyzer in a predefined order. The cable connections for each analysis probe are often altered to support that need. Thus, one analysis probe might require that you connect cable 2 to analyzer pod 2, while another will require you to connect cable 5 to analyzer pod 2. See Chapter 3 for connection information.

- ❏ Check the activity indicators for status lines locked in a high or low state.

- ❏ Verify that the STAT, DATA, and ADDR format labels have not been modified from their default values.

  These labels must remain as they are configured by the configuration file. Do not change the names of these labels or the bit assignments within the labels. Some analysis probes also require other data labels. See Chapter 3 for more

information.

- ❏ Verify that all microprocessor caches and memory managers have been disabled.

  In most cases, if the microprocessor caches and memory managers remain enabled you should still get inverse assembly. It may be incorrect because a portion of the execution trace was not visible to the logic analyzer.

  To determine if a cache is on or off, examine the most significant bit of the ICCST register (for the instruction cache) or the DCCST register (for data cache). If this bit is 1, the cache is on; if the bit is 0, the cache is off.

  For instructions on how to disable the cache, see page 128.

- ❏ Verify that storage qualification has not excluded storage of all the needed opcodes and operands.

- ❏ Verify that the data format is big-endian.

  Unlike most processors, the MPC821/860 can run in either little-endian or big-endian mode. The inverse assembler can only decode data in big-endian format. To verify the format of the data, the MSR or Machine State Register must be examined. The least significant bit (bit 0 according to Motorola convention, or bit 31 according to IBM convention) indicates the mode of the processor. A value of 1 indicates little-endian mode. A value of 0 indicates big-endian mode.

Inverse assembler will not load or run

You need to ensure that you have the correct system software loaded on your analyzer.

❑ Ensure that the inverse assembler is on the same disk as the configuration files you are loading.

Configuration files for the state analyzer contain a pointer to the name of the corresponding inverse assembler. If you delete the inverse assembler or rename it, the configuration process will fail to load the disassembler.

See Chapter 2 for details.

If the inverse assembler cannot determine cycle sizes

HP 16600A and HP 16700A logic analysis systems only

The processor sometimes fails to put correct information on the bus.

- In the Listing window, select the Preferences menu.

Enter each memory bank address into the appropriate field.

Intermodule Measurement Problems

Some problems occur only when you are trying to make a measurement involving multiple modules.

An event wasn't captured by one of the modules

If you are trying to capture an event that occurs very shortly after the event that arms one of the measurement modules, it may be missed due to internal analyzer delays. For example, suppose you set an oscilloscope module to trigger upon receiving a trigger signal from the logic analyzer because you are trying to capture a pulse that occurs right after the analyzer's trigger state. If the pulse occurs too soon after the analyzer's trigger state, the oscilloscope will miss the pulse.

- ❏ Adjust the skew in the Intermodule menu.

  You may be able to specify a skew value that enables the event to be captured.

- ❏ Change the trigger specification for modules upstream of the one with the problem.

  If you are using a logic analyzer to trigger an oscilloscope module, try specifying a trigger state one state before the one you are using. This may be more difficult than working with the skew because the prior state may occur more often and not always be related to the event you are trying to capture with the oscilloscope.

Analysis Probe Messages

This section lists some of the messages that the analyzer displays when it encounters a problem.

". . . Inverse Assembler Not Found"

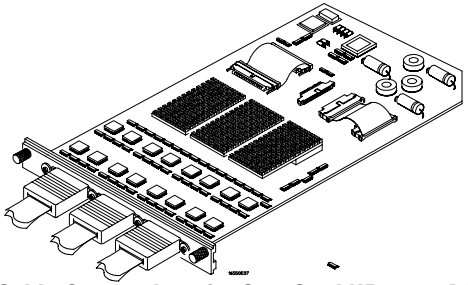
This error occurs if you rename or delete the inverse assembler file that is attached to the configuration file.

Ensure that the inverse assembler file is not renamed or deleted, and that it is located in the correct directory:

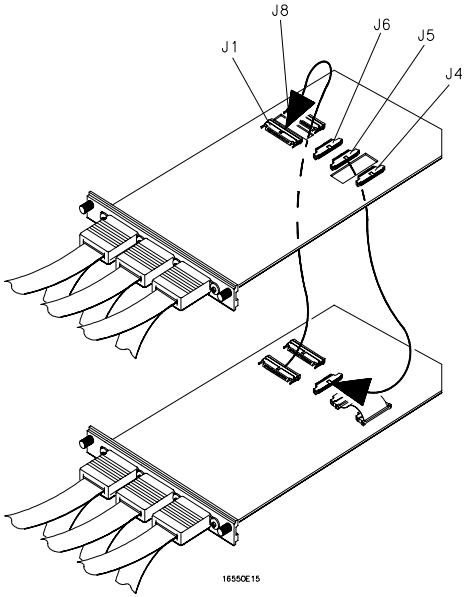
- For HP 16600A/700A-series logic analysis systems it should be in /hplogic/ia.
- For other logic analyzers it should be in the same directory as the configuration file.

"Measurement Initialization Error"

This error occurs when you have installed the cables incorrectly for one or two HP 16550A logic analysis cards. The following diagrams show the correct cable connections for one-card and two-card installations. Ensure that your cable connections match the silk screening on the card, and that they are fully seated in the connectors. Then, repeat the measurement.



Cable Connections for One-Card HP 16550A Installations



Cable Connections for Two-Card HP 16550A Installations

See Also

The *HP 16550A 100-MHz State/500-MHz Timing Logic Analyzer Service Guide*.

"No Configuration File Loaded"

This is usually caused by trying to load a configuration file for one type of module/system into a different type of module/system.

- ❏ Verify that the appropriate module has been selected when you load the configuration file. Selecting Load {All} will cause incorrect operation when loading most analysis probe interface configuration files.

See Also Chapter 3 describes how to load configuration files.

"Selected File is Incompatible"

This occurs when you try to load a configuration file for the wrong module. Ensure that you are loading the appropriate configuration file for your logic analyzer.

"Slow or Missing Clock"

- ❏ This error message might occur if the logic analyzer cards are not firmly seated in the logic analysis system frame. Ensure that the cards are firmly seated.
- ❏ This error might occur if the target system is not running properly. Ensure that the target system is on and operating properly.
- ❏ If the error message persists, check that the logic analyzer pods are connected to the proper connectors on the analysis probe interface. See Chapter 3 to determine the proper connections.

"Time from Arm Greater Than 41.93 ms"

The HP 16550A state/timing analyzers have a counter to keep track of the time from when an analyzer is armed to when it triggers. The width and clock rate of this counter allow it to count for up to 41.93 ms before it overflows. Once the counter has overflowed, the system does not have the data it needs to calculate the time between module triggers. The system must know this time to be able to display data from multiple modules on a single screen.

"Waiting for Trigger"

If a trigger pattern is specified, this message indicates that the specified trigger pattern has not occurred. Verify that the triggering pattern is correctly set.

- When analyzing microprocessors that fetch only from word-aligned addresses, ensure that the trigger condition is set to look for an opcode fetch at an address corresponding to a word boundary.

Returning Parts to Hewlett-Packard for Service

The repair strategy for this emulation solution is board replacement.

Exchange assemblies are available when a repairable assembly is returned to Hewlett-Packard. These assemblies have been set up on the "Exchange Assembly" program. This lets you exchange a faulty assembly with one that has been repaired, calibrated, and performance verified by the factory. The cost is significantly less than that of a new assembly.

To return a part to Hewlett-Packard

1. Follow the procedures in this chapter to make sure that the problem is caused by a hardware failure, not by configuration or cabling problems.
2. In the U.S., call 1-800-403-0801. Outside the U.S., call your nearest HP sales office. Ask them for the address of the nearest HP service center.
3. Package the part and send it to the HP service center.

   Keep any parts which you know are working. For example, if only the target interface module is broken, keep the emulation module and cables.
4. When the part has been replaced, it will be sent back to you.

   The unit returned to you will have the same serial number as the unit you sent to HP.

   The HP service center can also troubleshoot the hardware and replace the failed part. To do this, send your entire measurement system to the service center, including the logic analysis system, analysis probe, and cables.

   In some parts of the world, on-site repair service is available. Ask the HP sales or service representative for details.

To obtain replacement parts

The following table lists some parts that may be replaced if they are damaged or lost. Contact your nearest Hewlett-Packard Sales Office for further information.

Analysis Probe Replaceable PartsCleaning the Instrument

HP Part Number	Description
E5346A	High-density Cable
E2476-66501	Analysis Probe Circuit Board
E2476-87602	BGA Extender (one pre-installed on the analysis probe board, and one supplied with the product for customer installation. Total 2)
E2476-87606	Double Header, 19x19
E2476-87605	BGA Socket, 19x19
E2476-87607	BGA Carrier
E5355A	BGA Probe Kit

Cleaning the Instrument

If the instrument requires cleaning:

1. Disconnect power from the instrument.
2. Clean the instrument using a soft cloth that has been moistened in a mixture of mild detergent and water.
3. Make sure that the instrument is completely dry before reconnecting it to a power source.

14

Troubleshooting the Emulation Module

If you have problems with the emulation module, your first task is to determine the source of the problem. Problems may originate in any of the following places:

- The connection between the emulation module and your debugger
- The emulation module itself
- The connection between the emulation module and the target interface module
- The connection between the target interface module and the target system
- The target system

You can use several means to determine the source of the problem:

- The troubleshooting guide on the next page
- The status lights on the emulation module
- The emulation module "performance verification" tests
- The emulation module's built-in "terminal interface" commands

Emulation Module Troubleshooting Guide

Common problems and what to do about them

Symptom	What to do	See also
Commands from the Emulation Control Interface have no effect	Check that you are using the correct firmware.	
Commands from debugger have no effect	Use the Emulation Control Interface to try a few built-in commands. If this works, your debugger may not be configured properly. If this does not work, continue with the steps for the next symptom....	page 299
Emulation module built-in commands do not work	**1** Check that the emulation module has been properly configured for your target system.	page 183
	2 Run the emulation module performance verification tests.	page 312
	3 If the performance verification tests pass, then there is an electrical problem with the connection to the target processor OR the target system may not have been designed according to "Designing a Target System."	page 169, page 302
"Slow or missing clock" message after a logic analyzer run	Check that the target system is running user code or is in reset. (This message can appear if the processor is in background mode.)	
"Slow clock" message in the Emulation Control Interface or "c>" prompt in the built-in terminal interface	Check that the clock rate is properly configured.	page 188
Some commands fail	Check the "restrict to real-time runs" configuration	page 189

Emulation Module Status Lights

The emulation module uses status lights to communicate various modes and error conditions.

The following table gives more information about the meaning of the power and target status lights.

❍ = LED is off
● = LED is on
✻ = Not applicable (LED is off or on)

Power/Target Status Lights

Pwr/Target LEDs	Meaning
❍ Reset ❍ Break ❍ Run	No target system power, or emulation module is not connected to the target system
● Reset ❍ Break ❍ Run	Target system is in a reset state
❍ Reset ● Break ❍ Run	The target processor is executing in Debug Mode
❍ Reset ❍ Break ● Run	The target processor is executing user code
❍ Reset ● Break ● Run	Only boot firmware is good (other firmware has been corrupted)

Emulation Module Built-in Commands

The emulation module has some built-in "terminal interface" commands which you can use for troubleshooting.

You can access the terminal interface using:

- A telnet (LAN) connection
- The Command Line window in the Emulation Control Interface
- A "debugger command" window in your debugger

To telnet to the emulation module

You can establish a telnet connection to the emulation module if:

- A host computer and the logic analysis system are both connected to a local-area network (LAN), and
- The host computer has the telnet program (often part of the operating system or an internet software package).

To establish a telnet connection:

1. Find out the port number of the emulation module.

   The default port number of the first emulation module in an HP 16600A/700A series logic analysis system is 6472. The default port of a second module in an HP 16600A-series system is 6476. The default port numbers of a third and fourth module in an expansion frame are 6480 and 6484. These port numbers can be changed, but that is rarely necessary.

2. Find out the LAN address or LAN name of the logic analysis system.

3. Start the telnet program.

   If the LAN name of the logic analysis system is "test2" and you have only one emulation module installed, the command might look like this:

   ```
   telnet test2 6472
   ```

4. If you do not see a prompt, press the <Return> key a few times.

   To exit from this telnet session, type <CTRL>D at the prompt.

To use the built-in commands

Here are a few commonly used built-in commands:

Commonly used built-in commands

b	Break—go into the background monitor state
cf	Configuration—read or write configuration options
help	Help—display online help for built-in commands
init	Initialize—init -c re-initializes everything in the emulation module except for the LAN software; init -p is the equivalent of cycling power (it will break LAN connections)
lan	configure LAN address (emulation probes only)
m	Memory—read or write memory
reg	Register—read or write a register
r	Run—start running user code
rep	Repeat—repeat a command or group of commands
rst	Reset—reset the target processor (the emulation module will wait for you to press the target's RESET button)
s	Step—do a low-level single step
ver	Version—display the product number and firmware version of the emulation module

The prompt indicates the status of the emulation module:

Emulation module prompts

U	Running user program
M	Running in background monitor
p	No target power
R	Emulation reset
r	Target reset
?	Unknown state

Examples

To set register R0, then view R0 to verify that it was set, enter:

```
R>rst -m
M>reg r0=ffff
M>reg r0
   reg R0=0000ffff
```

To break execution then step a single instruction, enter:

```
M>b
M>s
  PC=xxxxxxxx
M>
```

To determine what firmware version is installed in the emulation module, enter:

```
M>ver
```

See Also

Use the `help` command for more information on these and other commands. Note that some of commands listed in the help screens are generic commands for HP emulators and may not be available for your product.

If you are writing your own debugger, contact HP for more information.

Problems with the Target System

This section describes how to determine whether your target system is causing problems with the operation of the emulation module.

What to check first

1 Try some basic built-in commands using the Command Line window or a telnet connection:

```
U>rst
R>
```

This should reset the target and display an "R>" prompt.

```
R>b
M>
```

This should stop the target and display an "M>" prompt.

```
M>reg r1
  reg r1=00000000
M>
```

This should read the value of the r1 register (the value will probably be different on your target system).

```
M>m 0..
   00000000   7c3043a6 7c2802a6 7c3143a6 4bf04111
   00000010   00000000 00000000 00000000 00000000
   00000020   00000000 00000000 00000000 00000000
   00000030   00000000 00000000 00000000 00000000
   00000040   00000000 00000000 00000000 00000000
   00000050   00000000 00000000 00000000 00000000
   00000060   00000000 00000000 00000000 00000000
   00000070   00000000 00000000 00000000 00000000
M>
```

This should display memory values starting at address 0.

```
M>s
```

This should execute one instruction at the current program counter.

If any of these commands don't work, there may be a problem with the design of your target system, a problem with the revision of the processor you are

using, or a problem with the configuration of the emulation module.

2 Check that the emulation module firmware matches your processor. To do this, enter:

```
M>ver
```

See Also Page 299 for information on entering built-in commands.

To interpret the initial prompt

The initial prompt can be used to diagnose several common problems. To get the most information from the prompt, follow this procedure:

1 Connect the emulation module to your target system.

2 Set the default configuration settings. Enter:

```
M>init -c
```

You can enter this command at any prompt. The emulation module will respond with the same information as printed by the "ver" command.

If the response is "!ERROR 905! Driver firmware is incompatible with ID of attached device"

Make sure the target interface module is connected to the cable of the emulation module, then try the "init -c" command again.

If the initial prompt is "p>"

Check pin 9 on header, 3.3V (V_{OD}).

If the initial prompt is "M>"

The processor entered debug mode without the help of the emulation module. Is another debugger connected?

If the initial prompt is "U>"

The emulation module is scanning the instruction register correctly. Now you can do some more tests:

3 Enter the reset command:

```
U>rst
R>
```

The "R>" prompt is a good response that indicates SRESET and HRESET are working.

If you see the “MSR.RI bit not set - Break may not be recoverable” error message

❑ Check that interrupt service routines (ISRs) in the target code meet the requirements listed in the PowerPC documentation.

For proper debugging in ISRs, the PowerPC documentation specifies that the exception handlers must do the following:

- As an epilogue to the ISR:
  - Save the SRR0, SRR1, DAR, DSISR registers
  - Set the RI bit in the MSR (Machine State Register.Recoverable Interrupt Bit)
- As a prologue to the ISR:
  - Restore the SRR0, SRR1, DAR, DSISR registers
  - Issue an RFI (Return from Interrupt) instruction

Upon entering the ISR, the processor clears the MSR.RI bit, and copies the IP (Instruction Pointer)->SRR0 and the MSR->SRR1. The SRR0 and SRR1 are the save and restore registers. These contain the information needed to return to the state prior to the interrupt.

The RI bit will prevent the processor from breaking into debug mode with a maskable debug port breakpoint. A non-maskable breakpoint is required to break the processor when the RI bit is cleared, resulting in a possible non-recoverable state.

Software breakpoints place a “Trap” instruction into the breakpoint address. If the trap instruction is executed within an ISR, a break to background mode will occur. This causes the SRR0 and SRR1 registers to be written over, causing a non-recoverable state. If the exception handler saves these registers, and sets the MSR.RI bit, the software breakpoint will always be recoverable.

If hardware breakpoints have no effect

Hardware breakpoints by default will not break the processor if they are set within an exception handler which has not saved the SRRs and set the MSR.RI bit. However, these can quite easily be reprogrammed to assert a non-

maskable break. Note that the breakpoint will halt the processor, but will cause a non-recoverable state.

To reprogram the hardware breakpoint to assert a non-maskable break:

```
M>bc -e hwbp
M>reg lctrl2
reg lctrl2=02018000
M>reg lctrl2=02018800   # OR in 0x00000800 with previous
                         # value.
```

Hardware breakpoints will now cause a non-maskable break, which will halt the processor regardless of the status of the MSR.RI bit. Again, note that in this case the break will be non-recoverable if the exception handler has not saved the SRRs.

If the target resets itself

The most common plug-in issue is the target resetting itself. If the PC is set to some initial location, and then a short time later, the PC=100 or PC=fff00100, the target is resetting itself. In most cases, the chip is causing the reset, not the target hardware.

There are a number of possible causes of the reset. To determine the cause of reset, read the RSR (Reset Status Register):

```
M>m -a2 -d2 288@reg  # telnet command that reads the RSR
```

The bits in this register show the cause of the reset:

RSR Bit Encoding

Bit		Cause of reset	Explanation
MSB	0	External Hard Reset	The emulation module actually uses an external reset when resetting the target.
	1	External Soft Reset	
	2	Loss of Lock	Caused when the PLL loses the phase lock on the external clock source.
	3	SW Watchdog	Make sure the SYPCR register disables the watchdog timer. R > reg cf_sypcr = ffffff88 or M > m -a4 -d4 4@reg = ffffff88
	4	Checkstop	Occurs when the processor enters a checkstop state.
	5	Debug Port Hard Reset	
	6	Debug Port Soft Reset	
	7	JTAG Reset	

To clear the RSR, execute the following:

```
M>m -a2 -d2 288@reg=ffff
```

If running from reset causes problems

Running from reset may cause some problems once background is entered. To insure proper operation, the DER register must have bits 31,30,29,28 set (0x0000000f), and the SYPCR register must have the “Disable watchdog freeze” bit set (0x00000080).

If you see the "!ASYNC_STAT 173!" error message

If after a break, the following error arises:

```
!ASYNC_STAT  173! MSR.RI bit not set - Break may not be recoverable
```

This indicates that the MSR.RI bit is not set, implying that a non-maskable break was needed, and the interrupt may not be recoverable. If this occurs while breaking out of regular code, then the MSR.RI bit was not set in the boot code. This can be fixed by “OR-ing” in 0x00000002 into the SRR1 register and resuming the run.

If there are problems with the debug port signals

❑ Check for pull down resistors on DSDI and DSCK.

Some target systems may have 220 Ohm pull downs on these two signals. These signals are series terminated by the analysis probe or TIM with a 46 Ohm resistor. A 220 Ohm pull-down would present a 20% drop in signal level when driven high, which could easily cause some malfunctions. There should be a very weak pull down on the target, if any at all. If you want to pull-down DSCK, use a value of 2.2K or greater.

To test the target system

The following program can be placed into memory.

```
start:  addi  r1,1     - 0x38210001
nop             - 0x60000000
nop             - 0x60000000
bra   start     - 0x4bffffff4
```

The opcode 0x4bffffff4 is a branch to a relative offset, so this program can be placed at any start address.

```
M>reg r1=0
M>m -a2 -d2 10000=3821,1,6000,0,6000,0,4bff,fff4
M>r 10000
U>reg r1
reg r1=00034567    # or some number
U>reg r1
reg r1=00102333    # or some number
U>
```

This program will loop forever, incrementing r1. This is a good test program to load once a memory system is up to make sure the microprocessor can run code out of memory.

Problems with the LAN Interface

If LAN communication does not work

If you cannot verify the connection, or if the commands are not accepted by the emulation module:

- ❏ Make sure that you wait for the power-on self test to complete before connecting.
- ❏ Make sure that the LAN cable is connected. Watch the LAN LED's on the back of the logic analysis system to see whether the system is seeing LAN activity. Refer to your LAN documentation for testing connectivity.
- ❏ Check that the host computer or debugger was configured with the correct LAN address. If the logic analysis system is on a different subnet than the host computer, check that the gateway address is correct.
- ❏ Make sure that the logic analysis system's IP address is set up correctly.

If it takes a long time to connect to the network

❏ Check the subnet masks on the other LAN devices connected to your network. All of the devices should be configured to use the same subnet mask.

Subnet mask error messages do not indicate a major problem. You can continue using the emulation module.

The subnet mask is set in the logic analysis system's System Admin window. If it then detects other subnet masks, it will generate error messages.

If there are many subnet masks in use on the local subnet, the logic analysis system may take a very long time to connect to the network after it is turned on.

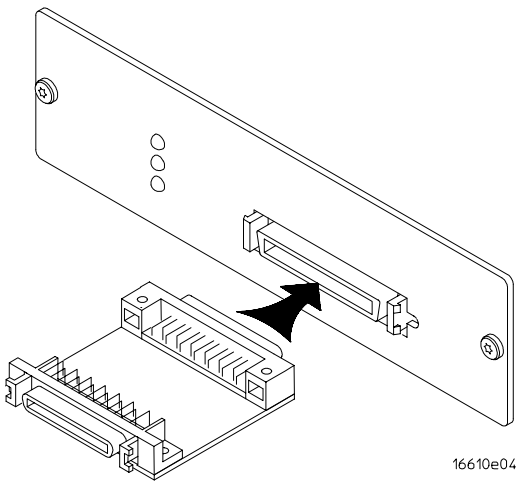
Problems with the Emulation Module

Occasionally you may suspect a hardware problem with the emulation module or target interface module. The procedures in this section describe how to test the hardware, and if a problem is found, how to repair or replace the broken component.

To run the built-in performance verification test using the logic analysis system

1 End any Emulation Control Interface or debugger sessions.

2 Disconnect the 50-pin cable from the emulation module, and plug the loopback test board (HP part number E3496-66502) into the emulation module.



3 In the system window, click the emulation module and select **Performance Verification**.

4 Click **Start PV**.

The results will appear on-screen.

To run complete performance verification tests using a telnet connection

1. Disconnect the 50-pin cable from the emulation module, and plug the loopback test board (HP part number E3496-66502) directly into the emulation module. Do not plug anything into the other end of the loopback test board.

   On a good system, the RESET LED will light and the BKG and USER LEDs will be out.

2. telnet to the emulation module.

3. Enter the **pv 1** command.

See Also

Options available for the "pv" command are explained in the help screen displayed by typing "help pF" or "? pv" at the prompt. Note, however, that some of the options listed may not apply to your emulation module.

Examples:

If you are using a UNIX system, to telnet to a logic analysis system named "mylogic", enter:

```
telnet mylogic 6472
```

Here are some examples of ways to use the **pv** command.

To execute both tests one time:

```
pv 1
```

To execute test 2 with maximum debug output repeatedly until a ^C is entered:

```
pv -t2 -v9 0
```

To execute tests 3, 4, and 5 only for 2 cycles:

```
pv -t3-5 2
```

The results on a good system with the loopback test board connected, are as follows:

```
M>pv 1

   Testing: HPE3499C Series Emulation System
     Test  1: Powerup PV Results                              Passed!
     Test  2: Target Probe Feedback Test                      Passed!
     Test  3: Boundary Scan Master Test                       Passed!
     Test  4: I2C Test                                        Passed!
     Test  5: Data Lines Test                                 Passed!
   PASSED  Number of tests: 1           Number of failures: 0

          Copyright (c) Hewlett-Packard Co. 1987
 All Rights Reserved.  Reproduction, adaptation, or translation without prior
 written permission is prohibited, except as allowed under copyright laws.

   HPE3499C Series Emulation System
     Version:   A.07.51 17Dec97
     Location:  Generics

   HPE3497A Motorola MPC800 Embedded PowerPC Emulator
     Version:   A.01.02 18Dec97
 M>
```

You may get an error like "!ERROR 172! Bad status code (0xff) from the hard reset sequence" just before the prompt. This is because the selftest loopback connector is installed instead of being connected to a real PowerPC target system. You may also get a “?>” prompt for the same reason, and this is normal and expected. Any errors after the “PASSED Number of tests: 1 Number of failures: 0” line can be ignored.

If a performance verification test fails

There are some things you can do if a failure is found on one of these tests. Details of the failure can be obtained through using a -v option ("verbose" level) of 2 or more.

> If the particular failure you see is not listed below, contact HP for assistance.

TEST 5: Target Probe Feedback Test

A verbose output on this test can be extensive. For example, the following is the output of this test if you forget to plug in the loopback test board.

```
p>pv -t5 -v2 1

   Testing: HPE3499A Series Emulation System
     Test # 5: Target Probe Feedback Test                    failed!
       Bad 20 Pin Status Read when unconnected = 0x7fb7
                            Expected Value = 0xffb7
       Bad 20 Pin Status Read when connected= 7fb7
                           Expected Value = 0x7fb7
       Output 19 Low not received on Input 11
       Output 11 Low not received on Input 19
       Output 13 Low not received on Input 1
       Output 12 High not received on Input 6
       Output 12 and Input 6 not pulled high on release
       Output 8 Low not received on Input 10
       Output 7 Low not received on Input 20
       Output 4 Low not received on Input 14
       Output 2 Low not received on Input 18
   FAILED  Number of tests: 1           Number of failures: 1
```

If the you get a verbose output like this, check to make sure that the loopback test board was connected properly.

TEST 6: Boundary Scan Master Test
TEST 7: I2C Test

If these tests are not executed, check that you have connected the loopback test board.

If these tests fail, return the emulation module to HP for replacement.

Returning Parts to Hewlett-Packard for Service

The repair strategy for this emulation solution is board replacement.

Exchange assemblies are available when a repairable assembly is returned to Hewlett-Packard. These assemblies have been set up on the "Exchange Assembly" program. This lets you exchange a faulty assembly with one that has been repaired, calibrated, and performance verified by the factory. The cost is significantly less than that of a new assembly.

To return a part to Hewlett-Packard

1. Follow the procedures in this chapter to make sure that the problem is caused by a hardware failure, not by configuration or cabling problems.
2. In the U.S., call 1-800-403-0801. Outside the U.S., call your nearest HP sales office. Ask them for the address of the nearest HP service center.
3. Package the part and send it to the HP service center.

   Keep any parts which you know are working. For example, if only the target interface module is broken, keep the emulation module and cables.
4. When the part has been replaced, it will be sent back to you.

   The unit returned to you will have the same serial number as the unit you sent to HP.

   The HP service center can also troubleshoot the hardware and replace the failed part. To do this, send your entire measurement system to the service center, including the logic analysis system, target interface module, and cables.

   In some parts of the world, on-site repair service is available. Ask an HP sales or service representative for details.

To obtain replacement parts

The following table lists some parts that may be replaced if they are damaged or lost. Contact your nearest Hewlett-Packard Sales Office for further information.

Part Numbers

Exchange Assemblies

Part Number	Description
16600-69515	Emulation module

Replacement Assemblies

Part number	Description
E3496-61603	10-pin target cable
E3496-61601	50-pin cable
E3496-66502	Loopback test board
E3497-66502	Target Interface Module
16700-61608	Expansion cable

Cleaning the Instrument

If the instrument requires cleaning:

1. Disconnect power from the instrument.
2. Clean the instrument using a soft cloth that has been moistened in a mixture of mild detergent and water.
3. Make sure that the instrument is completely dry before reconnecting it to a power source.

Analysis Probe A probing solution connected to the target microprocessor. It provides an interface between the signals of the target microprocessor and the inputs of the logic analyzer. Formerly called a “preprocessor.”

Background Debug Monitor In Also called Debug Mode, In Background, and In Monitor. The normal processor execution is suspended and the processor waits for commands from the debug port. The debug port commands include the ability to read and write memory, read and write registers, set breakpoints and start the processor running (exit the Background Debug Monitor).

Debug Mode See Background Debug Monitor.

Elastomeric Probe Adapter A connector that is fastened on top of a target microprocessor using a retainer and knurled nut. The conductive elastomer on the bottom of the probe adapter makes contact with pins of the target microprocessor and delivers their signals to connection points on top of the probe adapter.

Emulation Module An emulation module is installed within the mainframe of a logic analyzer. It provides run control within an emulation and analysis test setup. See Emulation Probe.

Emulator An emulation module or an emulation probe

Emulation Probe An emulation probe is a standalone instrument connected via LAN to the mainframe of a logic analyzer or to a host computer. It provides run control within an emulation and analysis test setup. Formerly called a "processor probe" or "software probe." See Emulation Module.

Extender A part whose only function is to provide connections from one location to another. One or more extenders might be stacked to raise a probe above a target microprocessor to avoid mechanical contact with other components installed close to the target microprocessor. Sometimes called a "connector board."

Flexible Adapter Two connection devices coupled with a flexible cable. Used for connecting probing hardware on the target microprocessor to the analysis probe.

General-Purpose Flexible Adapter A cable assembly that

connects the signals from an elastomeric probe adapter to an analysis probe. Normally, a male-to-male header or transition board makes the connections from the general-purpose flexible adapter to the analysis probe.

High-Density Adapter Cable A cable assembly that delivers signals from an analysis probe hardware interface to the logic analyzer pod cables. A high-density adapter cable has a single Mictor connector that is installed into the analysis probe, and two cables that are connected to corresponding odd and even logic analyzer pod cables.

High-Density Termination Adapter Cable Same as a High-Density Adapter Cable, except it has a termination in the Mictor connector.

In Background, In Monitor See Background Debug Monitor.

Inverse Assembler Software that displays captured bus activity as assembly language mnemonics. In addition, inverse assemblers may show execution history or decode control busses.

Jumper Moveable direct electrical connection between two points.

Mainframe Logic Analyzer A logic analyzer that resides on one or more board assemblies installed in an HP 16500, HP 1660-series, or HP 16600A/700A-series mainframe.

Male-to-male Header A board assembly that makes point-to-point connections between the female pins of a flexible adapter or transition board and the female pins of an analysis probe.

Monitor, In See Background Debug Monitor.

Preprocessor See Analysis Probe.

Preprocessor Interface See Analysis Probe.

Probe adapter See Elastomeric Probe Adapter.

Processor Probe See Emulation Probe.

Prototype Analyzer The HP 16505A prototype analyzer acts as an analysis and display processor for the HP 16500B/C logic analysis system. It provides a windowed interface and powerful analysis capabilities. Replaced by HP 16600A/700A-series logic analysis systems.

Run Control Probe See Emulation

Probe and Emulation Module.

Setup Assistant A software program that guides a user through the process of connecting and configuring a logic analyzer to make measurements on a specific microprocessor.

Shunt Connector. See Jumper.

Software Probe See Emulation Probe.

Solution HP's term for a set of tools for debugging your target system. A solution includes probing, inverse assembly, the HP B4620B Source Correlation Tool Set, and an emulation module.

Stand-alone Logic Analyzer A standalone logic analyzer has a predefined set of hardware components which provide a specific set of capabilities. It is designed to perform logic analysis. A standalone logic analyzer differs from a mainframe logic analyzer in that it does not offer card slots for installation of additional capabilities, and its specifications are not modified based upon selection from a set of optional hardware boards that might be installed within its frame.

Target Control Port An 8-bit, TTL port on a logic analysis system that you can use to send signals to your target system. It does not function like a pattern generator or emulation module, but more like a remote control for the target's switches.

Target Interface Module A small circuit board which connects the 50-pin cable from an emulation module or emulation probe to signals from the debug port on a target system.

TIM See Target Interface Module.

Trigger Specification A set of conditions that must be true before the instrument triggers. See the printed or online documentation of your logic analyzer for details.

Transition Board A board assembly that obtains signals connected to one side and rearranges them in a different order for delivery at the other side of the board.

1/4-Flexible Adapter An adapter that obtains one-quarter of the signals from an elastomeric probe adapter (one side of a target microprocessor) and makes them available for probing.

Index

DECLARATION OF CONFORMITY

according to ISO/IEC Guide 22 and EN 45014

Manufacturer's Name:	Hewlett-Packard Company
Manufacturer's Address:	Colorado Springs Division 1900 Garden of the Gods Road Colorado Springs, CO 80907 USA

declares, that the product

Product Name:	Processor Probe
Model Number(s):	HP E3497A
Product Option(s):	All

conforms to the following Product Specifications:

Safety:	IEC 1010-1:1990+A1 / EN 61010-1:1993 UL3111 CSA-C22.2 No. 1010.1:1993	
EMC:	CISPR 11:1990 / EN 55011:1991	Group 1 Class A
	IEC 555-2:1982 + A1:1985 / EN 60555-2:1987	
	IEC 555-3:1982 + A1:1990 / EN 60555-3:1987 + A1:1991	
	IEC 801-2:1991 / EN 50082-1:1992	4 kV CD, 8 kV AD
	IEC 801-3:1984 / EN 50082-1:1992	3 V/m, {1kHz 80% AM, 27-1000 MHz}
	IEC 801-4:1988 / EN 50082-1:1992	0.5 kV Sig. Lines, 1 kV Power Lines

Supplementary Information:

The product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC and carries the CE marking accordingly.

This product was tested in a typical configuration with Hewlett-Packard test systems.

Colorado Springs, 10/30/96

John Strathman, Quality Manager

European Contact: Your local Hewlett-Packard Sales and Service Office or Hewlett-Packard GmbH, Department ZQ / Standards Europe, Herrenberger Strasse 130, D-71034 Böblingen Germany (FAX: +49-7031-14-3143)

Product Regulations

Safety

IEC 1010-1:1990+A1 / EN 61010-1:1993
UL3111
CSA-C22.2 No. 1010.1:1993

EMC

CE
ISM 1-A

This Product meets the requirement of the European Communities (EC) EMC Directive 89/336/EEC.

Emissions EN55011/CISPR 11 (ISM, Group 1, Class A equipment)

N279

Immunity

EN50082-1	Code[1]	Notes[2]
IEC 801-2 (ESD) 4kV CD, 8kV AD	1	1
IEC 801-3 (Rad.) 3V/m	1	
IEC 801-4 (EFT) 0.5kV, 1kV	1	

[1]Performance Codes:
1 PASS - Normal operation, no effect.
2 PASS - Temporary degradation, self recoverable.
3 PASS - Temporary degradation, operator intervention required.
4 FAIL - Not recoverable, component damage.

[2]Notes:
1 The target cable assembly is sensitive to ESD events. Use standard ESD preventative practices to avoid component damage.

Sound Pressure Level N/A

DECLARATION OF CONFORMITY

according to ISO/IEC Guide 22 and EN 45014

Manufacturer's Name:	Hewlett-Packard Company
Manufacturer's Address:	Colorado Springs Division 1900 Garden of the Gods Road Colorado Springs, CO 80907 USA

declares, that the product

Product Name:	Logic Analyzer
Model Number(s):	HP 16600A, HP 16601A, HP 16602A, HP 16603A
Product Option(s):	All

conforms to the following Product Specifications:

Safety:	IEC 1010-1:1990+A1 / EN 61010-1:1993 UL3111 CSA-C22.2 No. 1010.1:1993	
EMC:	CISPR 11:1990 / EN 55011:1991	Group 1 Class A
	IEC 555-2:1982 + A1:1985 / EN 60555-2:1987	
	IEC 555-3:1982 + A1:1990 / EN 60555-3:1987 + A1:1991	
	IEC 801-2:1991 / EN 50082-1:1992	4 kV CD, 8 kV AD
	IEC 801-3:1984 / EN 50082-1:1992	3 V/m, {1kHz 80% AM, 27-1000 MHz}
	IEC 801-4:1988 / EN 50082-1:1992	0.5 kV Sig. Lines, 1 kV Power Lines

Supplementary Information:

The product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC and carries the CE marking accordingly.

This product was tested in a typical configuration with Hewlett-Packard test systems.

Colorado Springs, 08/18/97

John Strathman, Quality Manager

European Contact: Your local Hewlett-Packard Sales and Service Office or Hewlett-Packard GmbH, Department ZQ / Standards Europe, Herrenberger Strasse 130, D-71034 Böblingen Germany (FAX: +49-7031-14-3143)

Product Regulations

Safety

IEC 1010-1:1990+A1 / EN 61010-1:1993
UL3111
CSA-C22.2 No. 1010.1:1993

EMC

CE
ISM 1-A

This Product meets the requirement of the European Communities (EC) EMC Directive 89/336/EEC.

Emissions

EN55011/CISPR 11 (ISM, Group 1, Class A equipment), IEC 555-2 and IEC 555-3

N279

Immunity

EN50082-1	Code[1]	Notes[2]
IEC 801-2 (ESD) 4kV CD, 8kV AD	3	
IEC 801-3 (Rad.) 3V/m	1	
IEC 801-4 (EFT) 0.5kV, 1kV	3	

[1]Performance Codes:
1 PASS - Normal operation, no effect.
2 PASS - Temporary degradation, self recoverable.
3 PASS - Temporary degradation, operator intervention required.
4 FAIL - Not recoverable, component damage.

[2]Notes: (none)

Sound Pressure Level

<60 dBA

DECLARATION OF CONFORMITY

according to ISO/IEC Guide 22 and EN 45014

Manufacturer's Name:	Hewlett-Packard Company
Manufacturer's Address:	Colorado Springs Division 1900 Garden of the Gods Road Colorado Springs, CO 80907 USA

declares, that the product

Product Name:	Logic Analyzer
Model Number(s):	HP 16700A
Product Option(s):	All

conforms to the following Product Specifications:

Safety:	IEC 1010-1:1990+A1 / EN 61010-1:1993 UL3111 CSA-C22.2 No. 1010.1:1993	
EMC:	CISPR 11:1990 / EN 55011:1991	Group 1 Class A
	IEC 555-2:1982 + A1:1985 / EN 60555-2:1987	
	IEC 555-3:1982 + A1:1990 / EN 60555-3:1987 + A1:1991	
	IEC 801-2:1991 / EN 50082-1:1992	4 kV CD, 8 kV AD
	IEC 801-3:1984 / EN 50082-1:1992	3 V/m, {1kHz 80% AM, 27-1000 MHz}
	IEC 801-4:1988 / EN 50082-1:1992	0.5 kV Sig. Lines, 1 kV Power Lines

Supplementary Information:

The product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC and carries the CE marking accordingly.

This product was tested in a typical configuration with Hewlett-Packard test systems.

Colorado Springs, 09/22/97

John Strathman, Quality Manager

European Contact: Your local Hewlett-Packard Sales and Service Office or Hewlett-Packard GmbH, Department ZQ / Standards Europe, Herrenberger Strasse 130, D-71034 Böblingen Germany (FAX: +49-7031-14-3143)

Product Regulations

Safety

IEC 1010-1:1990+A1 / EN 61010-1:1993
UL3111
CSA-C22.2 No. 1010.1:1993

EMC

This Product meets the requirement of the European Communities (EC) EMC Directive 89/336/EEC.

CE
ISM 1-A

Emissions

EN55011/CISPR 11 (ISM, Group 1, Class A equipment), IEC 555-2 and IEC 555-3

N279

Immunity

EN50082-1	Code[1]	Notes[2]
IEC 801-2 (ESD) 4kV CD, 8kV AD	3	
IEC 801-3 (Rad.) 3V/m	1	
IEC 801-4 (EFT) 0.5kV, 1kV	1	

[1]Performance Codes:
1 PASS - Normal operation, no effect.
2 PASS - Temporary degradation, self recoverable.
3 PASS - Temporary degradation, operator intervention required.
4 FAIL - Not recoverable, component damage.

[2]Notes: (none)

Sound Pressure Level

Less than 60 dBA

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in subparagraph (C) (1) (ii) of the Rights in Technical Data and Computer Software Clause in DFARS 252.227-7013. Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, CA 94304 U.S.A. Rights for non-DOD U.S. Government Departments and Agencies are set forth in FAR 52.227-19 (c) (1,2).

Document Warranty

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

Hewlett-Packard shall not be liable for errors contained herein or for damages in connection with the furnishing, performance, or use of this material.

Safety

This apparatus has been designed and tested in accordance with IEC Publication 1010, Safety Requirements for Measuring Apparatus, and has been supplied in a safe condition. This is a Safety Class I instrument (provided with terminal for protective earthing). Before applying power, verify that the correct safety precautions are taken (see the following warnings). In addition, note the external markings on the instrument that are described under "Safety Symbols."

Warning

- Before turning on the instrument, you must connect the protective earth terminal of the instrument to the protective conductor of the (mains) power cord. The mains plug shall only be inserted in a socket outlet provided with a protective earth contact. You must not negate the protective action by using an extension cord (power cable) without a protective conductor (grounding). Grounding one conductor of a two-conductor outlet is not sufficient protection.
- Only fuses with the required rated current, voltage, and specified type (normal blow, time delay, etc.) should be used. Do not use repaired fuses or short-circuited fuseholders. To do so could cause a shock of fire hazard.
- Service instructions are for trained service personnel. To avoid dangerous electric shock, do not perform any service unless qualified to do so. Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.
- If you energize this instrument by an auto transformer (for voltage reduction), make sure the common terminal is connected to the earth terminal of the power source.
- Whenever it is likely that the ground protection is impaired, you must make the instrument inoperative and secure it against any unintended operation.
- Do not operate the instrument in the presence of flammable gasses or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.
- Do not install substitute parts or perform any unauthorized modification to the instrument.
- Capacitors inside the instrument may retain a charge even if the instrument is disconnected from its source of supply.

Safety Symbols

Instruction manual symbol: the product is marked with this symbol when it is necessary for you to refer to the instruction manual in order to protect against damage to the product.

Hazardous voltage symbol.

Earth terminal symbol: Used to indicate a circuit common connected to grounded chassis.

WARNING

The Warning sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to, could result in personal injury. Do not proceed beyond a Warning sign until the indicated conditions are fully understood and met.

CAUTION

The Caution sign denotes a hazard. It calls attention to an operating procedure, practice, or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product. Do not proceed beyond a Caution symbol until the indicated conditions are fully understood or met.

Hewlett-Packard
P.O. Box 2197
1900 Garden of the Gods Road
Colorado Springs, CO 80901-2197, U.S.A.

Product Warranty

This Hewlett-Packard product has a warranty against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Hewlett- Packard Company will, at its option, either repair or replace products that prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Hewlett- Packard.

For products returned to Hewlett-Packard for warranty service, the Buyer shall prepay shipping charges to Hewlett- Packard and Hewlett-Packard shall pay shipping charges to return the product to the Buyer. However, the Buyer shall pay all shipping charges, duties, and taxes for products returned to Hewlett-Packard from another country.

Hewlett-Packard warrants that its software and firmware designated by Hewlett-Packard for use with an instrument will execute its programming instructions when properly installed on that instrument. Hewlett-Packard does not warrant that the operation of the instrument software, or firmware will be uninterrupted or error free.

Limitation of Warranty

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by the Buyer, Buyer- supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

No other warranty is expressed or implied. Hewlett-Packard specifically disclaims the implied warranties of merchantability or fitness for a particular purpose.

Exclusive Remedies

The remedies provided herein are the buyer's sole and exclusive remedies. Hewlett-Packard shall not be liable for any direct, indirect, special, incidental, or consequential damages, whether based on contract, tort, or any other legal theory.

Assistance

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products. For any assistance, contact your nearest Hewlett-Packard Sales Office.

Certification

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology, to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

About this edition

This is the *Solutions for the Motorola Embedded PowerPC MPC860/821 User's Guide*.

Publication number
E2476-97005, July, 1999
Printed in USA.

The information in this manual previously appeared in:
E2476-97004 January 1999
E2476-97003 March 1998
E2476-97002 June 1995
E2766-97002 August 1996
E3497-97001 March 1997

New editions are complete revisions of the manual. Many product updates do not require manual changes, and manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.

Comments welcome!

Send your comments or suggestions regarding this manual to:
documentation@col.hp.com