
User's Guide

Publication number E2455-97011
September 1999

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

© Copyright Hewlett-Packard Company 1994-1999
All Rights Reserved

Solutions for the PowerPC 603

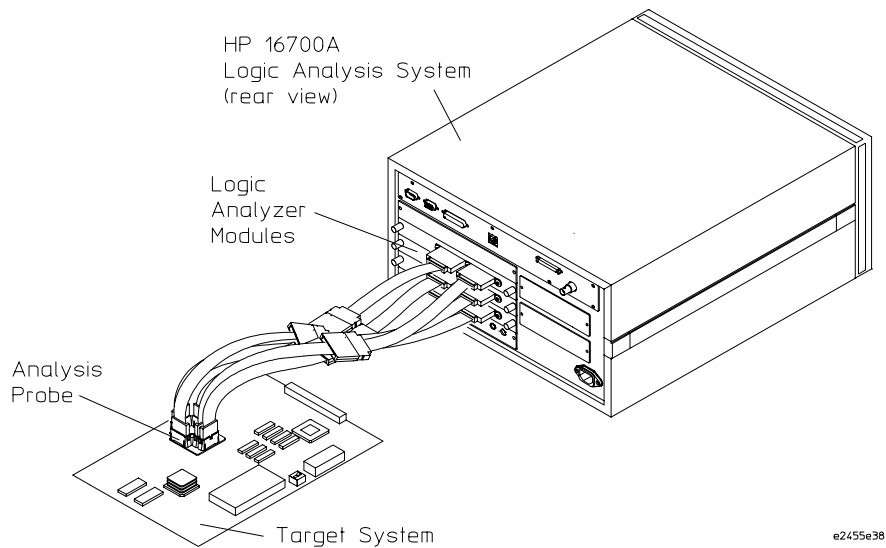
HP Solutions for the PowerPC 603—At a Glance

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.

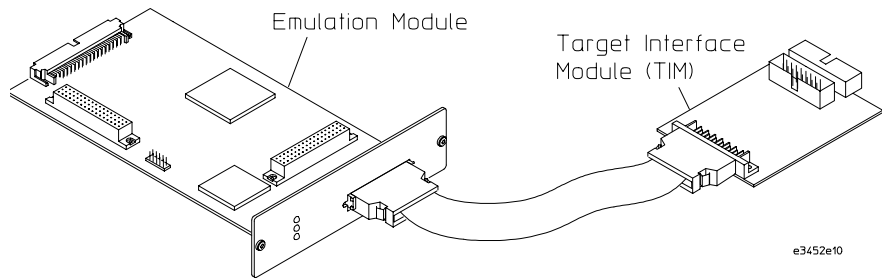
The analysis probe can be purchased alone, or as part of an emulation solution.



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.

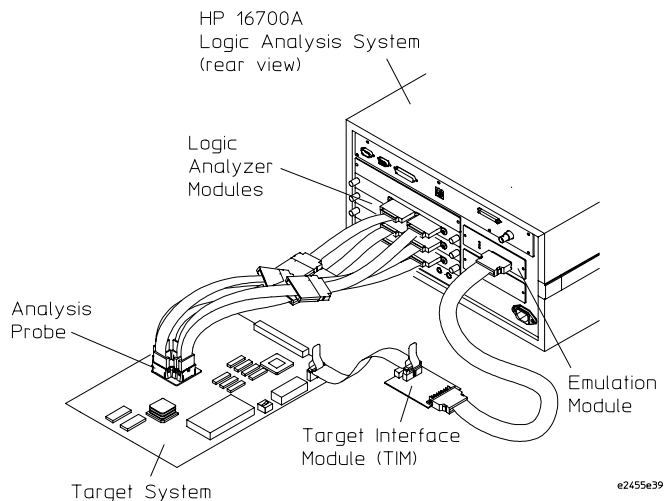
Emulation Module and Target Interface Module

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 use the target interface module (TIM) to connect the emulation module to a debug port on the target system.



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
PowerPC 603, PowerPC 603e, PowerPC 603ev, and PowerPC 603ei Up to 66 MHz 240-pin PQFP	HP E9587A Option #002	HP E2455B PQFP analysis probe and inverse assembler
PowerPC 603, PowerPC 603e, PowerPC 603ev, and PowerPC 603ei any package, custom probing	HP E9587A Option #001	Inverse assembler

Emulation Solution

Processors supported	Product ordered	Includes
PowerPC 603, PowerPC 603e, PowerPC 603ev, and PowerPC 603ei Up to 66 MHz 240-pin PQFP	HP E9487A Option #002	HP E2455B PQFP analysis probe, inverse assembler, emulation module, target interface module (TIM), HP B4620B Source Correlation Tool Set
PowerPC 603, PowerPC 603e, PowerPC 603ev, and PowerPC 603ei any package, custom probing	HP E9487A Option #001	Inverse assembler, emulation module, target interface module (TIM), HP B4620B Source Correlation Tool Set

Aliases

PowerPC 603e aliases: 603P, 603R	When the emulation solution is configured for the PowerPC 603e, it can also be used with the PowerPC 603P and PowerPC 603R.
----------------------------------	-----------------------------------------------------------------------------------------------------------------------------

Contents

HP Solutions for the PowerPC 603—At a Glance

Analysis Probe	2
Emulation Module and Target Interface Module	3
Emulation Solution	3

In This Book

1 Overview 19

Setup Checklist	21
Setup Assistant	23
Analysis Probe	24
Equipment supplied	24
Minimum equipment required	26
Additional equipment supported	26
Logic analyzers supported	27
Logic analyzer software version requirements	29
Emulation Module	30
Equipment supplied	30
Minimum equipment required	31
Emulation Solution	32
Additional Information Sources	33

2 Installing Software 35

- Installing and loading 36
- What needs to be installed 37
- To install the software from CD-ROM (HP 16600A/700A) 38
- To list software packages which are installed (HP 16600A/700A) 39
- To install software on other logic analyzers 39

3 Connecting and Configuring the Analysis Probe 41

- Target System Requirements 43
 - Keep-out area on the target board 43
 - Clearance above the target board 43
 - Analysis probe — circuit board dimensions 44
- Power-on/Power-off Sequence 45
 - To power on HP 16600A and HP 16700A-series logic analysis systems 45
 - To power on all other logic analyzers 46
 - To power off 46
- Connecting the Analysis Probe to the Target System 47
 - To connect to a QFP target system 48
 - To disconnect the analysis probe from the target system 49
 - To separate the analysis probe upper assembly from the probe head 50
 - To reconnect analysis probe and probe head 50
- Connecting the Analysis Probe to the Logic Analyzer 51
 - 64-bit data 51
 - 32-bit data 51
 - No data 52
 - To connect the high-density termination cables to the analysis probe 53
 - To connect to the HP 16710/11/12A or HP 16550A analyzer (two-card) for 64-bit data analysis 54

Contents

To connect to the HP 16710/11/12A or HP 16550A analyzer (one or two-card) for 32-bit data analysis	56
To connect to the HP 16710/11/12A or HP 16550A analyzer (one or two-card) for no-data analysis	57
To connect to the HP 16715/16/17A (two-card) for 64-bit data analysis	58
To connect to the HP 16715/16/17A (three-card) for 64-bit data analysis	59
To connect to the HP 16715/16/17A (two or three-card) for 32-bit analysis	61
To connect to the HP 16715/16/17A (one-card) for no-data analysis	62
To connect to the HP 16600A logic analyzer for 64-bit data analysis	63
To connect to the HP 16600A logic analyzer for 32-bit data analysis	64
To connect to the HP 16600A logic analyzer for no-data analysis	65
To connect to the HP 16601A logic analyzer for 64-bit data analysis	66
To connect to the HP 16601A logic analyzer for 32-bit data analysis	67
To connect to the HP 16601A logic analyzer for no-data analysis	68
To connect to the HP 16602A logic analyzer for 32-bit data analysis	69
To connect to the HP 16602A logic analyzer for no-data analysis	70
To connect to the HP 16603A logic analyzer for no-data analysis	71
To connect to the HP 16554/55/56/57 (two-card) for 64-bit data analysis	72
To connect to the HP 16554/55/56/57 (three-card) for 64-bit data analysis	73
To connect to the HP 16554/55/56/57 (two or three-card) for 32-bit analysis	75

Contents

To connect to the HP 16554/55/56/57 (one-card) for no-data analysis	76
To connect to the HP 1660A/AS/C/CS/CP/E/ES/EP logic analyzers for 64-bit data analysis	77
To connect to the HP 1670A/D/E logic analyzers for 64-bit data analysis	78
Configuring the Logic Analysis System	79
To load configuration and inverse assembler files into HP 16600A/700A logic analysis systems from the system hard disk	80
To load configuration and inverse assembler files into HP 16600A/700A logic analysis systems from the floppy disk	81
To load configuration and inverse assembler files—other logic analyzers	82

4 Designing Connectors for Custom Probing for the Inverse Assembler 85

Direct Probing with GP Probes	87
Designing and Using Built-in Connectors	88
AMP Mictor 38 connectors	89
2x20 headers	94
2x10 headers	95
Connecting the Logic Analyzer to the Target System Connectors	96

5 Analyzing the PPC603 with an HP 16600A/16700A-series Logic Analyzer 97

Modes of Operation	99
State-per-ack mode	99
State-per-clock mode	99
Timing mode	100
Modes of Analysis	101
Inverse assembly analysis	101
Cache-on trace reconstruction	101
Logic Analyzer Configuration	102
Format menu	102
Trigger dialog	107
Using the Inverse Assembler	109
Using Cache-On Trace Reconstruction	109
Enabling branch exception disassembly	112
Inverse Assembler Modes of Operation	113
To use the Invasm menu	114
Loading the Inverse Assembler	114
Setting the Inverse Assembler Preferences	114
To set the memory map preferences	115
To set the processor options preferences	116
To set the decoding options preferences	118
To set the opcode source preferences	123
Display Filtering	124
To enable/disable the instruction cache on the PPC603	126
To display captured state data	128
Inverse assembler output format	129

6 Analyzing the PPC603 with an HP 1660A/1670A/ 16500B/C-series Logic Analyzer 133

Modes of Operation 135

State-per-ack mode 135

State-per-clock mode 135

Timing mode 136

Logic Analyzer Configuration 137

Format menu 137

Trigger dialog 142

Using the Inverse Assembler 144

To disable the instruction cache on the PPC603 144

To display captured state data 146

Inverse assembler output format 148

To use the Invasm menu (HP 1660, HP 1670, and HP 16500B/C
mainframes) 151

7 Symbols and Source Code in the Analyzer 155

Symbols 157

Object file symbols 157

Predefined PPC603 symbols 157

User-defined symbols 158

Symbol use requirements 159

To use object file symbols in the HP 16600A/700A 160

Compilers for PPC603 163

Source Code 166

Inverse assembler generated PC (software address) label 168

Access to source code files 169

Triggering on Symbols and Source Code	170
Using trigger alignment	170
To correlate relocatable code using the address offset	171
Using storage qualification	172

8 Connecting and Configuring the Emulation Module 173

Connecting and Configuring the Emulation Module	174
Overview	174

Using the Emulation Control Interface	175
To start the Emulation Control Interface from the main System window	176
To start the Emulation Control Interface from the Workspace window	177
To start the Emulation Control Interface from the Workspace window for an emulation probe	177

Designing a Target System for the Emulation Module	178
Design requirements for PowerPC 603 and PowerPC 603e	178
Operating considerations for the PowerPC 603	179
Operating considerations for the PowerPC 603e	180
Operating considerations for the PowerPC 603ev	182
PowerPC JTAG interface connections and resistors	183

Installing the Emulation Module	185
To install the emulation module in an HP 16700A-series logic analysis system or an HP 16701A expansion frame	186
To install the emulation module in an HP 16600A-series logic analysis system	188
To test the emulation module	189

Contents

Connecting the Emulation Module to the Target System	190
To connect to a target system using a JTAG port	191
To Update Firmware	192
To display current firmware version information	193
To verify communication between the emulator and target system	193
Configuring the Emulation Module	194
What can be configured	195
To configure using the Emulation Control Interface	196
To configure using the built-in commands	197
To configure using a debugger	199
To configure restriction to real-time runs	199
To configure the JTAG clock speed (communication speed)	200
To configure reset operation	201
To set memory read delays	201
To set memory write delays	202
To generate parity bits on memory operations	202
To set data bus size	203
To enable data retry mode	204
To configure the memory read operation	205
To configure data memory write operations	206
To configure instruction memory write operations	207
Testing the emulator and target system	209
To test memory accesses	209
To test with a running program	210

9 Using the Emulation Module with a Debugger

211

Benefits of using a debugger	212
Compatibility with other logic analysis system tools	212

Contents

Minimum requirements	214
Is your debugger compatible with the emulation module?	214
LAN connection	214
Compatibility with the Emulation Control Interface	214
Setting up Debugger Software	215
To connect the logic analysis system to the LAN	216
To change the port number of an emulation module	217
To verify communication with the emulation module	217
To operate the logic analysis system using a web browser	218
To export the logic analysis system's display to a workstation	218
To export the logic analysis system's display to a PC	219
To enable or disable processor caches	220
Using the Green Hills debugger	222
Compatibility	222
Overview	222
Getting started	223
To configure the emulation module and target using an initialization script	226
To perform common debugger tasks	227
To send commands to the emulation module	227
To view commands sent by MULTI to the emulation module	227
To reinitialize the system	228
To disconnect from the emulation module	228
Error conditions	228
Using the Microtec Research debugger	230
Compatibility	230
Overview	230
Getting started	230
To configure the emulation module and target using an INCLUDE file	232
To perform common debugger tasks	233
gtwinit and gtwreset command sequences	233
To send commands to the emulation module	234
To view commands sent by XRAY	234

Contents

To disconnect from the emulation module and target 234
Error conditions 235

Using the Software Development Systems debugger 236
Compatibility 236
Overview 236
Startup behavior 236
Getting started 237
To send commands to the emulation module 239
Error conditions 240

10 Using the Analysis Probe and Emulation Module Together 241

What are some of the tools I can use? 242
Which assembly-level listing should I use? 243
Which source-level listing should I use? 243
Where can I find practical examples of measurements? 243

Triggering the Emulation Module from the Analyzer 244
To stop the processor when the logic analyzer triggers on a line of source code (Source Viewer window) 244
To stop the processor when the logic analyzer triggers (Intermodule window) 246
To minimize the "skid" effect 247
To stop the analyzer and view a measurement 248

Tracing until the processor halts 249
To capture a trace before the processor halts 250

Triggering the Logic Analyzer from the Emulation Module 251
The emulation module trigger signal 251
Group Run 252

Debuggers can cause triggers	254
To trigger the analyzer when the processor halts - timing mode	255
To trigger the analyzer when the processor reaches a breakpoint	257

11 Hardware Reference 259

Analysis probe—operating characteristics	261
Theory of operation	262
Analysis probe—signal-to-connector mapping	262
Emulation module—operating characteristics	268
Emulation module—electrical characteristics	269

12 General-Purpose ASCII (GPA) Symbol File Format 271

GPA Record Format Summary	274
SECTIONS	276
FUNCTIONS	277
VARIABLES	277
SOURCE LINES	278
START ADDRESS	279
Comments	279

13 Troubleshooting the Analysis Probe 281

Logic Analyzer Problems	283
Intermittent data errors	283
Unwanted triggers	284
No activity on activity indicators	284

Contents

No trace list display	285
Analyzer won't power up	285
Analysis Probe Problems	286
Target system will not boot up	286
Erratic trace measurements	287
Capacitive loading	287
Inverse Assembler Problems	288
No inverse assembly or incorrect inverse assembly	288
Inverse assembler will not load or run	290
Intermodule Measurement Problems	291
An event wasn't captured by one of the modules	291
Analysis Probe Messages	292
". . . Inverse Assembler Not Found"	292
"Measurement Initialization Error"	293
"No Configuration File Loaded"	294
"Selected File is Incompatible"	294
"Slow or Missing Clock"	294
"Time from Arm Greater Than 41.93 ms"	295
"Waiting for Trigger"	295
Returning Parts to Hewlett-Packard for Service	296
To return a part to Hewlett-Packard	296
To obtain replacement parts	297
Cleaning the Instrument	297

14 Troubleshooting the Emulation Module 299

Emulation Module Troubleshooting Guide	301
Emulation Module Status Lights	302
Emulation Module Built-in Commands	303
To telnet to the emulation module	303
To use the built-in commands	304
Solving Target System Problems	306
What to check first	306
To check the debug port connector signals	308
To interpret the initial prompt	309
If the response is “!ERROR 905! Driver firmware is incompatible with ID of attached device”	309
If the initial prompt is “p>”	309
If the initial prompt is “M>”	309
If the initial prompt is “c>”	309
If the initial prompt is “?>” with “ERROR 171!”	309
If the initial prompt is “U>”	310
If the prompt after rst is “?>” with “ERROR 171!”	310
If the rst command fails	310
If the prompt after rst is “U>”	311
If the prompt after b is “M>” with error messages	311
If the prompt after b is “M>” with no error messages	311
If you can get to the “M>” prompt	311
If you see memory-related problems	313
If running from reset causes problems	315
If you see the “!ASYNC_STAT 173!” error message	315
To test the target system	316
Problems with the LAN Interface	317
If LAN communication does not work	317
If it takes a long time to connect to the network	318

Contents

Problems with the Emulation Module	319
To run the built-in performance verification test using the logic analysis system	319
To run complete performance verification tests using a telnet connection	320
If a performance verification test fails	322
Returning Parts to Hewlett-Packard for Service	323
To return a part to Hewlett-Packard	323
To obtain replacement parts	324
Cleaning the Instrument	325

Glossary 327

Index 331

Overview

Chapter 1: Overview

This chapter describes:

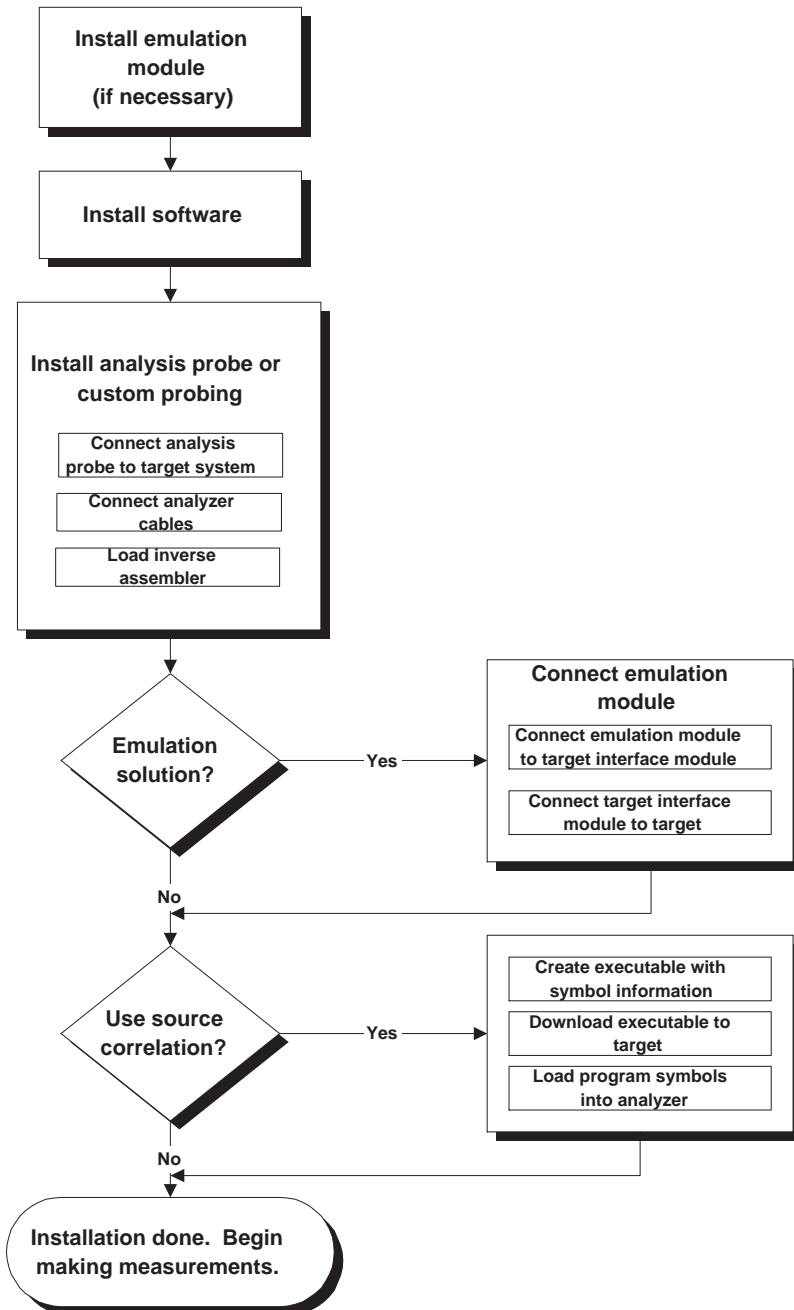
- Setup Checklist
- Setup Assistant
- Equipment used with the analysis probe (including a list of logic analyzers supported)
- Equipment used with the emulation module
- Additional information sources

Setup Checklist

Follow these steps to connect your equipment:

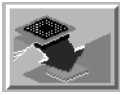
- Check that you received all of the necessary equipment. See page 24 and page 30.
- If you need to install an emulation module in an HP 16600A/700A series logic analysis system, see page 185.
- If you ordered this analysis probe or emulation solution with your HP 16600A/700A-series logic analysis system, 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 the “Installing Software” chapter (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, use the information beginning on page 41 to connect and configure your analysis probe.

Chapter 1: Overview Setup Checklist



E2455F01.VSD

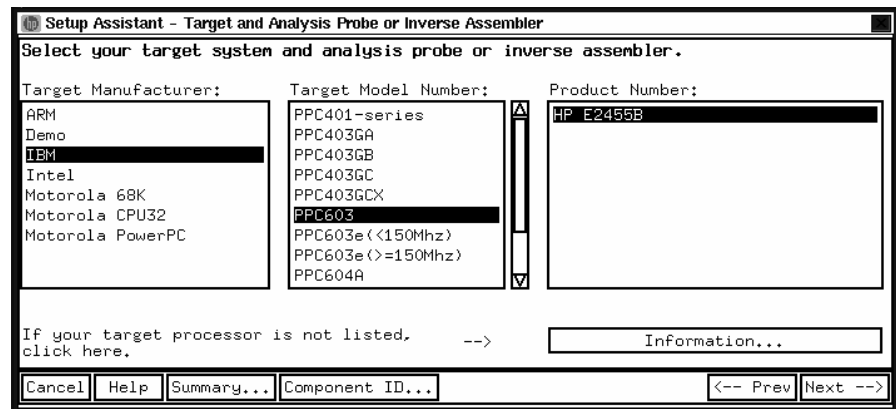
Setup Assistant



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 the “Installing Software” chapter (page 35).

NOTE:

The Setup Assistant may not fully support no_data or 32-bit data setup.

Analysis Probe

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

Equipment supplied

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

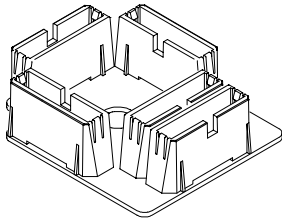
HP E2455B Analysis Probe

- The HP E2455B analysis probe circuit board.
- Five HP E5346A high-density termination cables.
- The HP E5363A elastomeric probe adapter.
- A Pry Tool, for separating the analysis probe from the probe adapter.
- Logic analyzer configuration files and the inverse assembler on a CD-ROM (for HP 16600A/700A series logic analysis systems).
- Logic analyzer configuration files and the inverse assembler on two 3.5-inch disks (for other HP logic analyzers).
- The inverse assembler and configuration files on a 3.5-inch disk (for the HP 16505A prototype analyzer).
- This User's Guide.

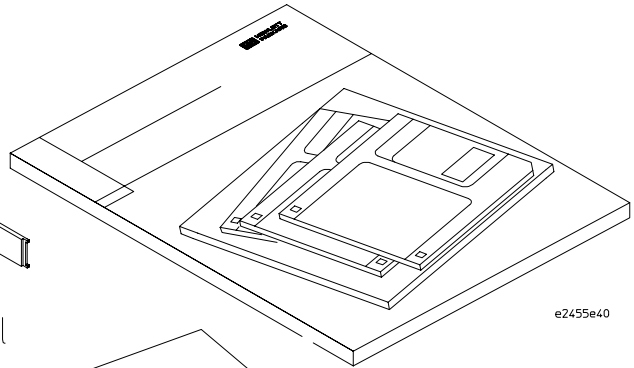
Inverse Assembler Software

If you ordered the HP E9587A #001 or HP E9487A #001, only the inverse assembler software and manuals are supplied.

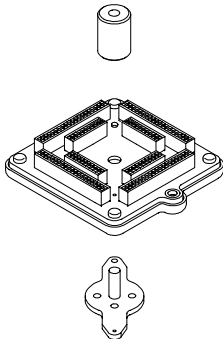
Analysis Probe



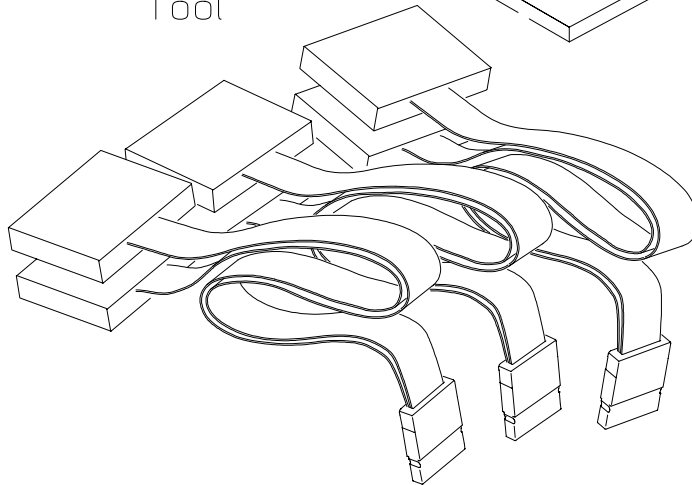
Software and Manual



Elastomeric
Probe Adapter



Pry
Tool



High-Density Termination Cables

Equipment Supplied with the HP E2455B Analysis Probe

Minimum equipment required

For state and timing analysis of a PowerPC 603 target system, you need all of the following items.

- The HP E2455B analysis probe, including the QFP probe adapter and flexible cables. If you are not using the HP E2455B analysis probe, you will need the inverse assembler software and the appropriate connectors on the target system. See page 85 information on designing the appropriate connectors into the target system.
- One of the logic analyzers listed on page 27. The logic analyzer software version requirements are listed on page 29.

Additional equipment supported

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 E2455B analysis probe and the inverse assembler. Logic analyzer software version requirements are shown on the following page.

The HP E2455B and the inverse assembler require eight logic analyzer pods (136 channels) for inverse assembly or four pods for program flow analysis. The analysis probe contains two additional pods that you can monitor.

Chapter 1: Overview
Analysis Probe

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
16603A**	68	100 MHz	125 MHz	64 k states
16550A (1** or 2 cards)	102/card	100 MHz	250 MHz	4 k states
16554A (1**, 2 or 3 cards)	68/card	70 MHz	125 MHz	512 k states
16555A (1**, 2 or 3 cards)	68/card	110 MHz	250 MHz	1 M states
16555D (1**, 2 or 3 cards)	68/card	110 MHz	250 MHz	2 M states
16556A (1**, 2 or 3 cards)	68/card	100 MHz	200 MHz	1 M states
16556D (1**, 2 or 3 cards)	68/card	100 MHz	200 MHz	2 M states
16557D (1**, 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
1670A	136	70 MHz	125 MHz	64 k or .5 M states
1670D	136	100 MHz	125 MHz	64 k or 1 M states
1670E	136	100 MHz	125 MHz	1 M states

* For the HP 16557D, the state and timing speeds decrease for four-card or five-card configurations.

** Program flow analysis only.

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 E2455B and inverse assembler. You can obtain the latest software at the following web site:

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

If your software version is older than those listed, load new system software with the higher version numbers before loading the HP E2455B or inverse assembler software.

Logic Analyzer Software Version Requirements

Logic Analyzer	Minimum Logic Analyzer Software Version for use with HP E2455B and Inverse Assembler Software
HP 16600 Series	The latest HP 16600 logic analyzer software version is on the CD-ROM shipped with this product.
HP 1660A/AS Series	3.01
HP 1660C/CS/CP/E/ES/EP Series	A.02.01
HP 1670A/D/E Series	A.02.02
Mainframes*	
HP 16700 Series	The latest HP 16700 logic analyzer software version is on the CD-ROM shipped with this product.
HP 16500C Mainframe	1.07
HP 16500B Mainframe	3.14

* The mainframes are used with the HP 16550 and HP 16554/55/56 logic analyzers.

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.

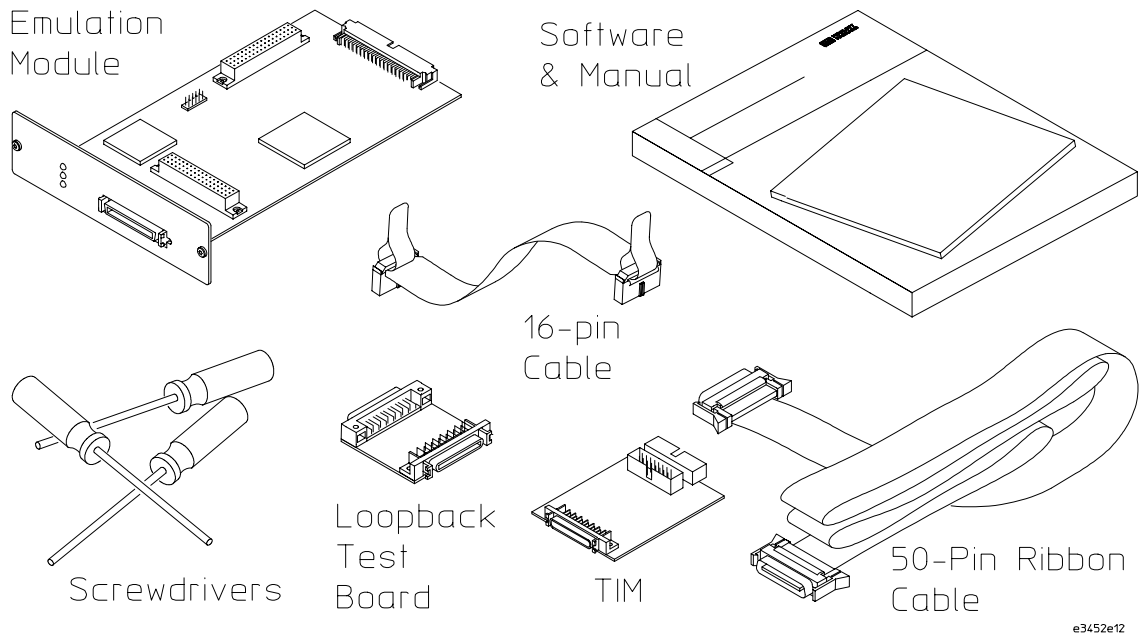
Emulation Module

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.
- A 16-pin ribbon cable for connecting the target interface module to the target system.
- One Torx T-8, one Torx T-10, and one Torx T-15 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. You can design a debug port connector on the target system. Chapter 7 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.

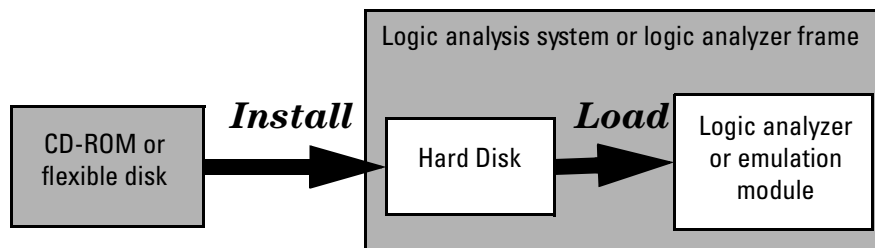
If you cannot easily find the information you need, send email to documentation@col.hp.com. Your comments will help HP improve future manuals. (This address is for comments only; contact your local HP representative if you need technical support.)

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. A password may be required to enable the tool set. Follow the instructions on the entitlement certificate.

The following files can be installed from the floppy disk supplied with your analysis probe.

- Logic analysis system configuration file
- Inverse assembler (automatically loaded with the configuration file)

Other HP logic analyzers

The following files can be installed from a floppy disk:

- Logic analyzer configuration files, which automatically load the inverse assembler. The inverse assembler must be on the same floppy disk as the configuration file.

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 “POWERPC6XX” 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 `/hpllogic/configs/hp/ppc6xx`.

The inverse assemblers are stored in `/hpllogic/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 list software packages which 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.

Connecting and Configuring the Analysis Probe

Chapter 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 (page 43)
- Read the power on/power off sequence (page 45)
- Connect the analysis probe to the target system (page 47)
- Connect the analysis probe to the logic analyzer (page 51)
- Configure the logic analyzer (page 78)

Target System Requirements

The PQFP probe adapter connects to the microprocessor on the target system, and the analysis probe connects to PQFP probe adapter. There are no additional target system requirements.

Keep-out area on the target board

The elastomeric probe requires a 44.20 mm by 44.20 mm keep-out area where it overhangs the QFP microprocessor. The maximum height of components under the elastomeric probe in this area cannot exceed 19.2 mm.

Clearance above the target board

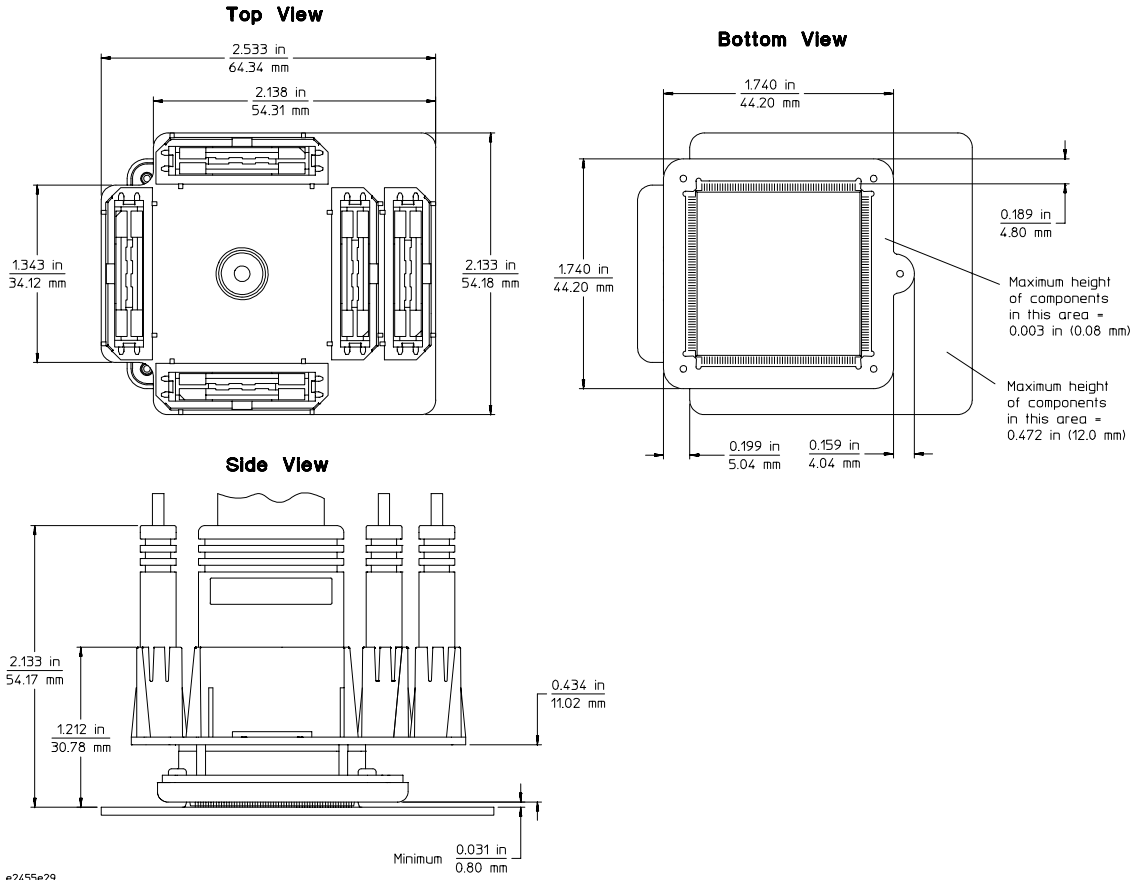
See the diagram on the next page for the dimensions of the analysis probe.

See Also

The *Emulation and Analysis Solutions for Motorola/IBM PowerPC 6XX Microprocessors* data sheet, available from your HP representative, has more detailed information.

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 E2455B 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

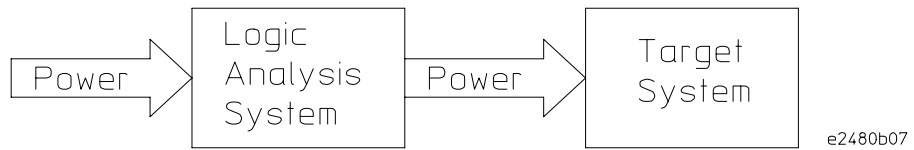
Ensure the target system is powered off.

- 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:

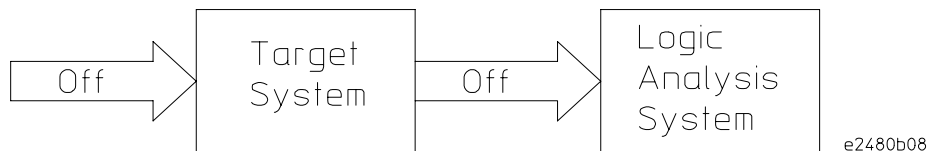
- 1 Logic analysis system.
- 2 Your target system.



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.



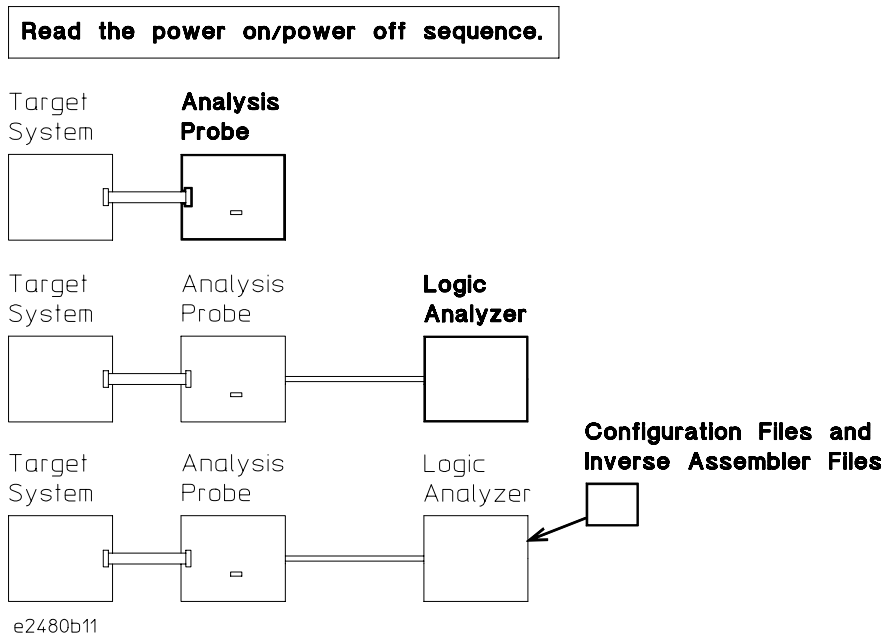
Connecting the Analysis Probe to the Target System

This section explains how to connect the HP E2455B 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).
- Attach the QFP probe adapter to the target system.
- Attach the analysis probe to the probe adapter.
- Connect the high-density termination cables to the probe adapter.

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

The following illustration shows the sequence for connecting the analysis probe.



Connection Sequence

To connect to a QFP target system

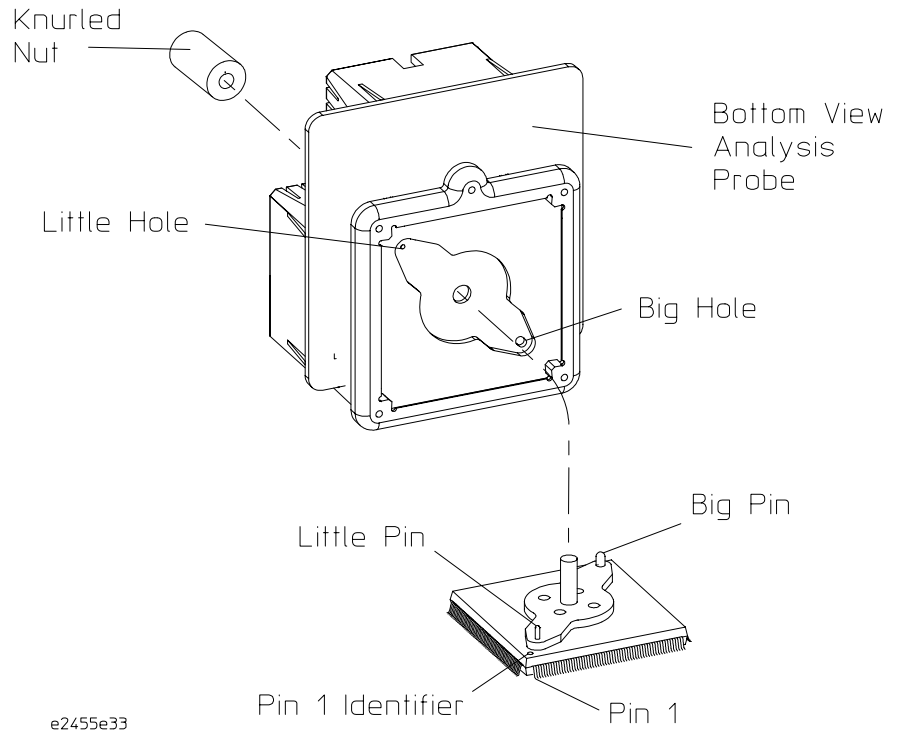
CAUTION:

To prevent equipment damage, remove power from the target system and make sure no logic analyzer cables are connected to the analysis probe.

- 1** Turn off the target system and logic analyzer.
- 2** Using the instructions in the *HP QFP Elastomeric Probing System Installation Guide*:
 - Prepare to attach the Retainer to the QFP microprocessor
 - Test the alignment before adhering the Retainer
 - Adhere the Retainer to the QFP microprocessor
 - Install the HP E2455B analysis probe as described in “Install the Probe Adapter”
- 3** Using the illustration on the next page, note the following indicators:
 - position of Pin 1 on the microprocessor
 - position of little pin on the retainer
 - position of little hole on the probe adapter

CAUTION:

Serious damage to the target system or analysis probe can result from incorrect connection. Note the position of pin 1 on the target system and analysis probe prior to making any connection. Also, take care to align the pins so that all pins are making contact.



Alignment for Analysis Probe and Target System

To disconnect the analysis probe from the target system

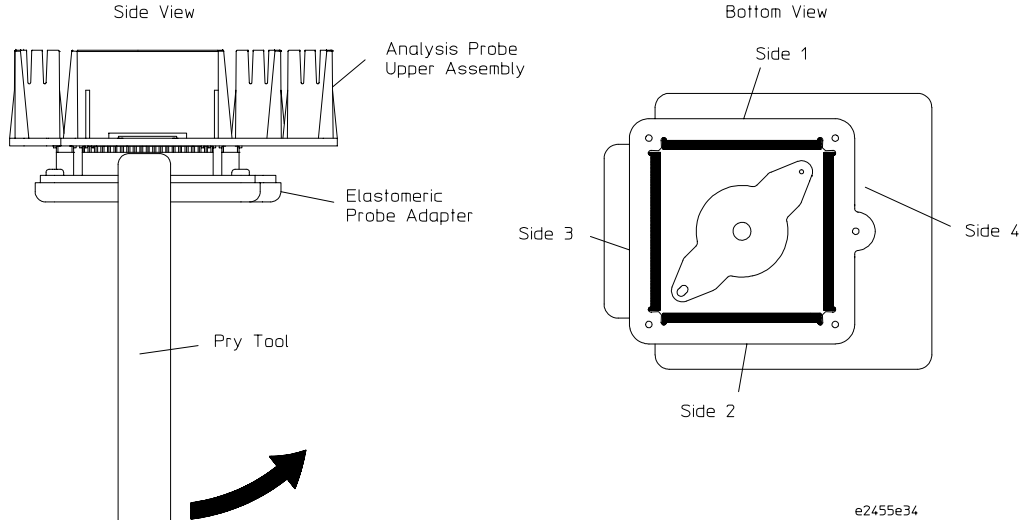
Use the following steps to disconnect the analysis probe from the target system.

- 1 Remove power from the target system.
- 2 Remove power from the logic analyzer.
- 3 Unscrew the knurled nut.

To separate the analysis probe upper assembly from the probe head

Hewlett-Packard does not recommend separating the analysis probe upper assembly from the elastomeric probe head. However, unforeseen circumstances might require you to separate the assembly.

Use the Cam Tool supplied. Insert the tool into the first side as shown in the following illustration, and rotate it until the connectors begin to separate. Repeat this process for the other three sides in consecutive order until the analysis probe upper assembly and the elastomeric probe head are separated.



To reconnect analysis probe and probe head

Place the elastomeric probe head in its protective cover. Orient the elastomeric probe head and the analysis probe upper assembly as shown in the illustration on the previous page. As you begin to insert the pins of the analysis probe upper assembly into the sockets on the elastomeric probe head, ensure that all of the pins are engaging. Look closely at both ends of all four sockets to ensure all pins are properly mated. Gently apply pressure until the connectors are fully mated.

Connecting the Analysis Probe to the Logic Analyzer

This section shows the connections between the logic analyzer pod cables and the cables on the analysis probe. The illustrations on the following pages show the analysis probe pod locations. There are three types of analysis available:

64-bit data

This type of analysis captures all of the data signals through the data bus. Use the appropriate page, listed below, for your logic analyzer. The configuration file names are included with the connection diagrams. Connectors J1 through J4 are required for inverse assembly. Connector J5 contains additional signals you might want to monitor.

- HP 16710/11/12A and HP 16550A logic analyzers (two cards) (page 54 and 55)
- HP 16715/16/17A logic analyzers (two or three cards) (page 58, 59, and 60)
- HP 16600A logic analysis system (page 63)
- HP 16601A logic analysis system (page 66)
- HP 16554/55/56/57 logic analyzers (two or three cards) (page 72, 73, and 74)
- HP 1660A/AS/C/CS/CP/E/ES/EP logic analyzers (page 77)
- HP 1670A/D/E logic analyzers (page 78)

32-bit data

This type of analysis will allow you to trace the 32 bits of DATA only, not DATA_B. Use the appropriate page, listed below, for your logic analyzer. The configuration file names are included with the connection diagrams. Connectors J1 and J2 are required for tracing program flow. Connectors J3 and J4 allow you to trace DATA

- HP 16710/11/12A and HP 16550A logic analyzers (one or two cards) (page 56)
- HP 16715/16/17A logic analyzers (two or three cards) (page 61)
- HP 16600A logic analysis system (page 64)

Chapter 3: Connecting and Configuring the Analysis Probe

Connecting the Analysis Probe to the Logic Analyzer

- HP 16601A logic analysis system (page 67)
- HP 16602A logic analysis system (page 69)
- HP 16554/55/56/57 logic analyzers (two or three cards) (page 75)

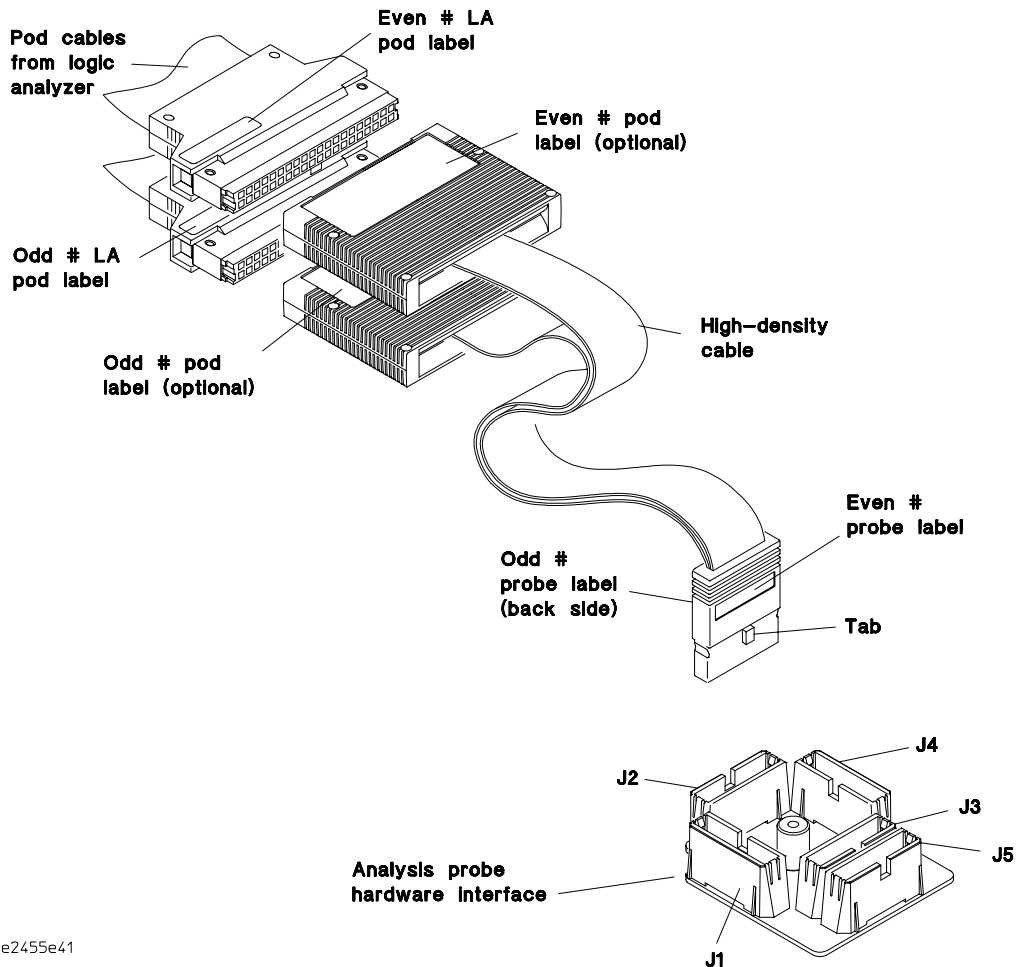
No data

This type of analysis allows you to trace program flow only using J1 and J2. Data values in the inverse assembly listing will be taken from the S-Record file, not from the data bus. Use the appropriate page listed below for your logic analyzer. The configuration file names are included with the connection diagrams.

- HP 16710/11/12A and HP 16550A logic analyzers (one or two cards) (page 57)
- HP 16715/16/17A logic analyzers (page 62)
- HP 16600A logic analysis system (page 65)
- HP 16601A logic analysis system (page 68)
- HP 16602A logic analysis system (page 70)
- HP 16603A logic analysis system (page 71)
- HP 16554/55/56/57 logic analyzers (one, two, or three cards) (page 76)

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

Five HP E5346A high-density termination cables, and labels to identify them, are included with the HP E2455B. 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.

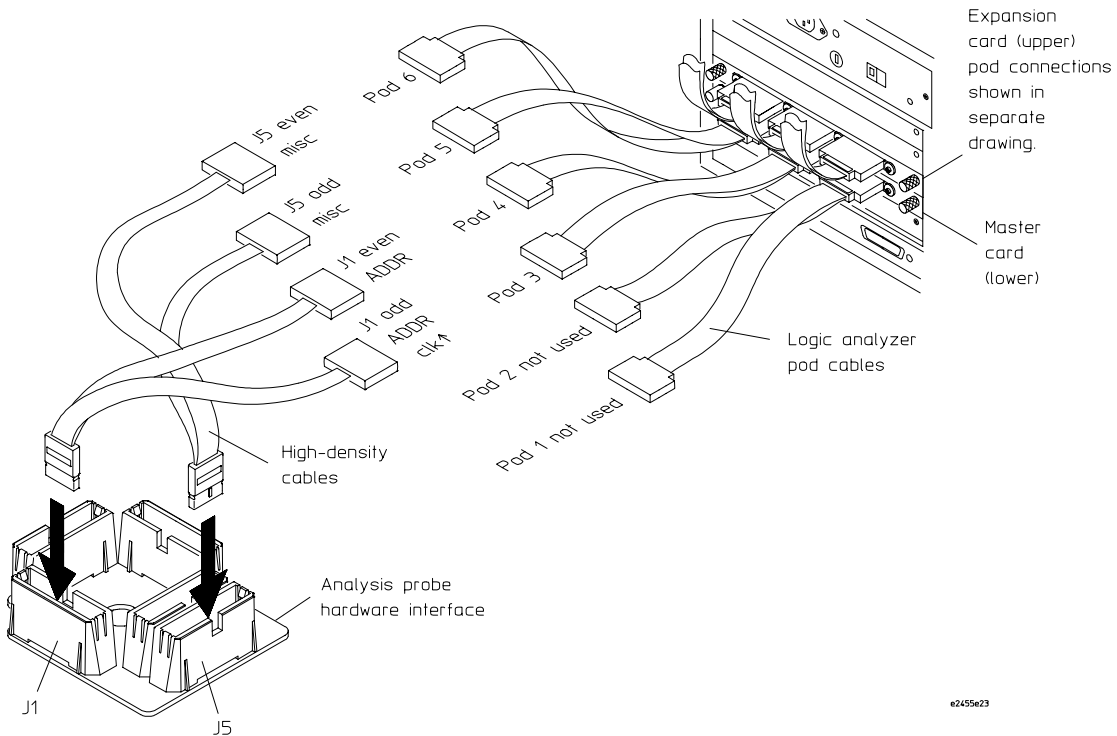


e2455e41

Connecting High-Density Cables to the Analysis Probe

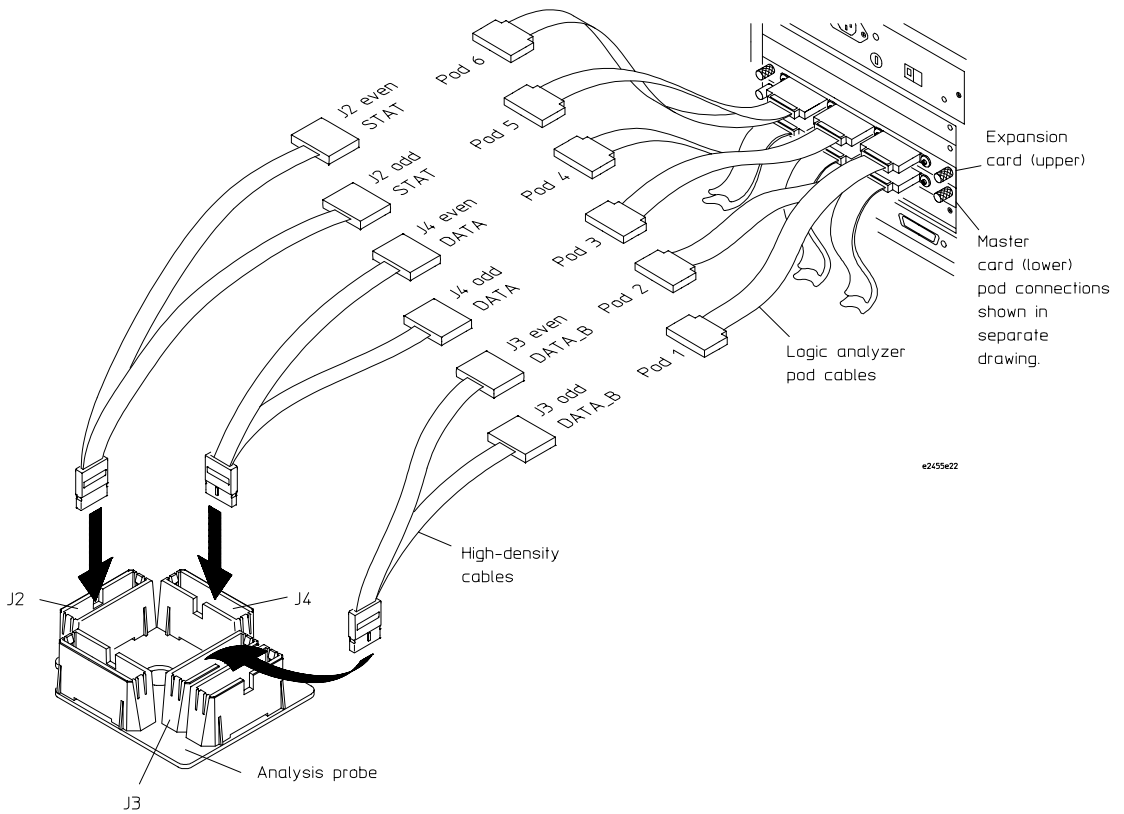
To connect to the HP 16710/11/12A or HP 16550A analyzer (two-card) for 64-bit data analysis

Use the figure below and on the following page to connect the analysis probe to the two-card HP 16710/11/12A or HP 16550A logic analyzer for 64-bit data analysis.



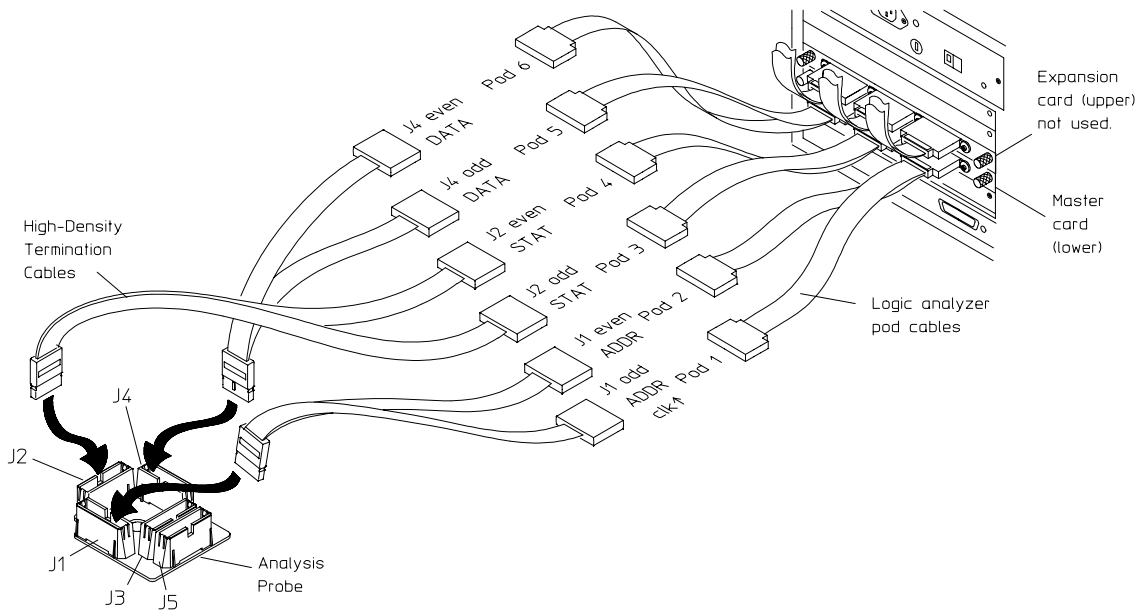
e2459e23

Two-card 64-bit HP 16710/11/12A or HP 16550A connections (continued)



To connect to the HP 16710/11/12A or HP 16550A analyzer (one or two-card) for 32-bit data analysis

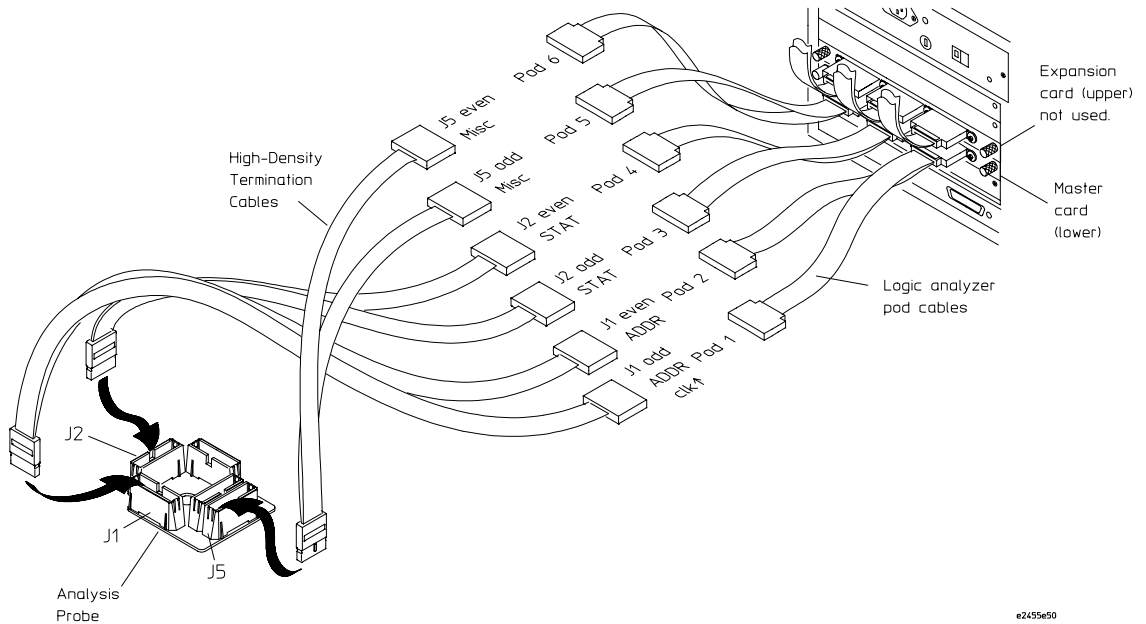
Use the figure below to connect the analysis probe to the one or two-card HP 16710/11/12A or HP 16550A logic analyzer for 32-bit data analysis.



e2455e49

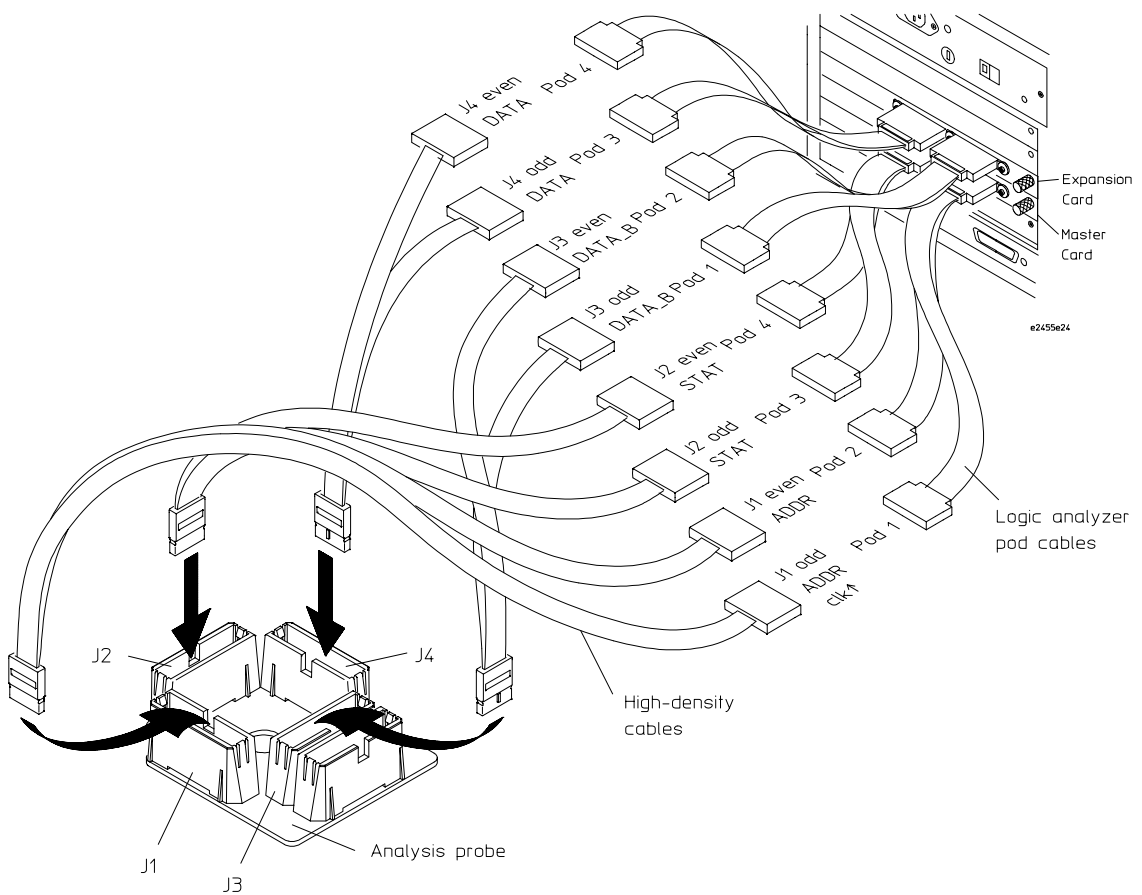
To connect to the HP 16710/11/12A or HP 16550A analyzer (one or two-card) for no-data analysis

Use the figure below to connect the analysis probe to the one-card HP 16710/11/12A or HP 16550A logic analyzer for no-data analysis.



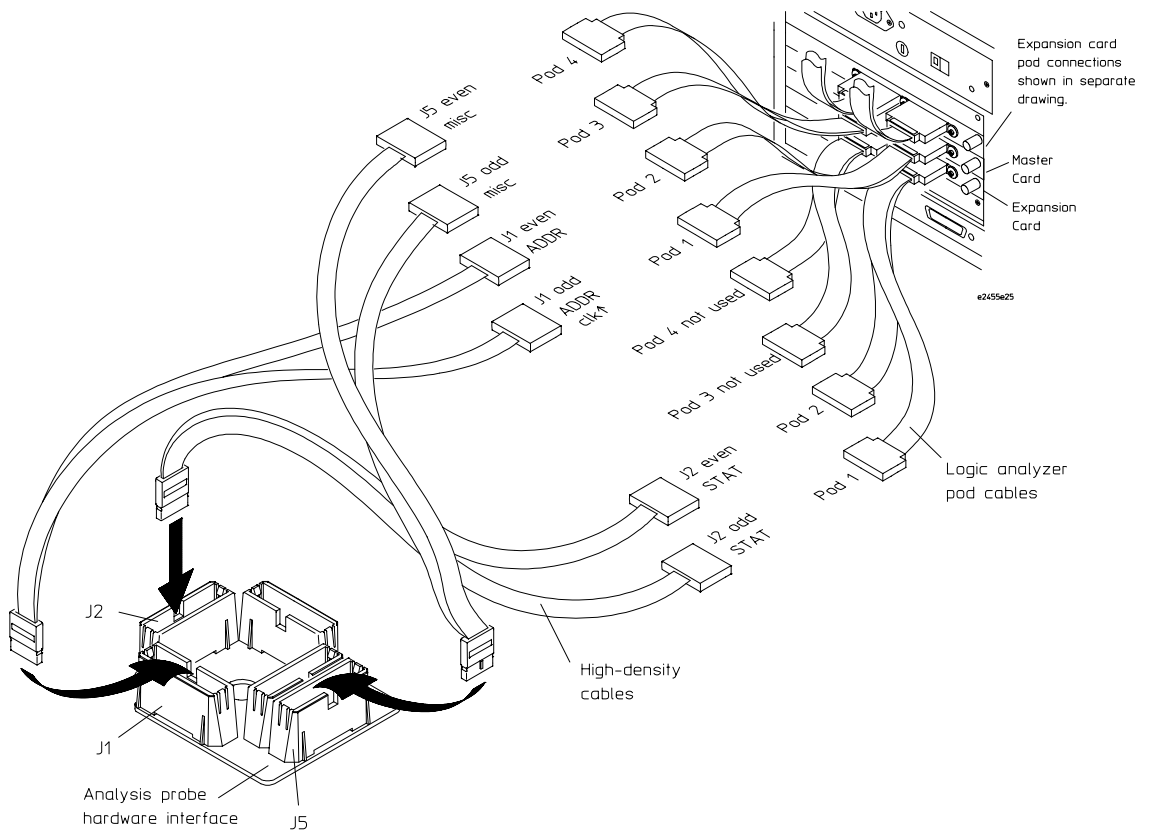
To connect to the HP 16715/16/17A (two-card) for 64-bit data analysis

Use the figure below to connect the analysis probe to the two-card HP 16715/16/17A logic analyzers.



To connect to the HP 16715/16/17A (three-card) for 64-bit data analysis

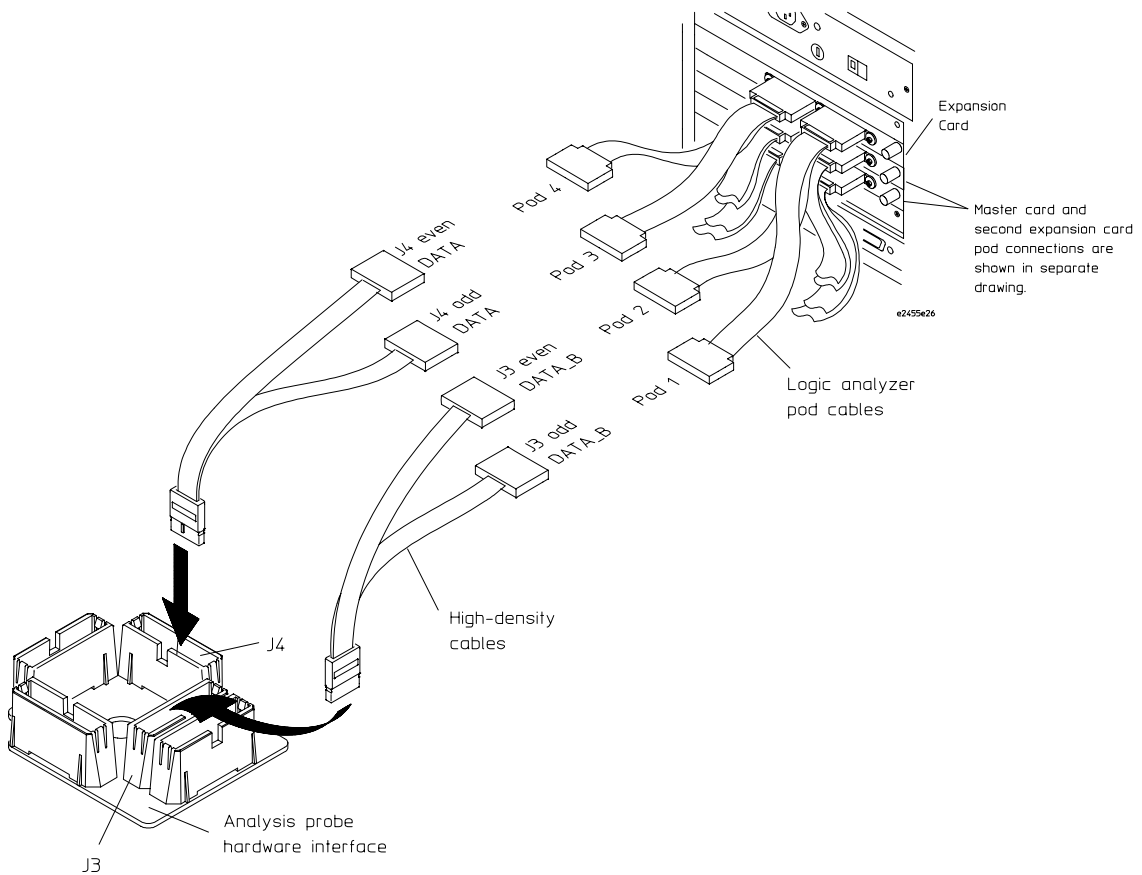
Use the figures below and on the following page to connect the analysis probe to the three-card HP 16715/16/17A logic analyzers for 64-bit data analysis.



Continued on next page

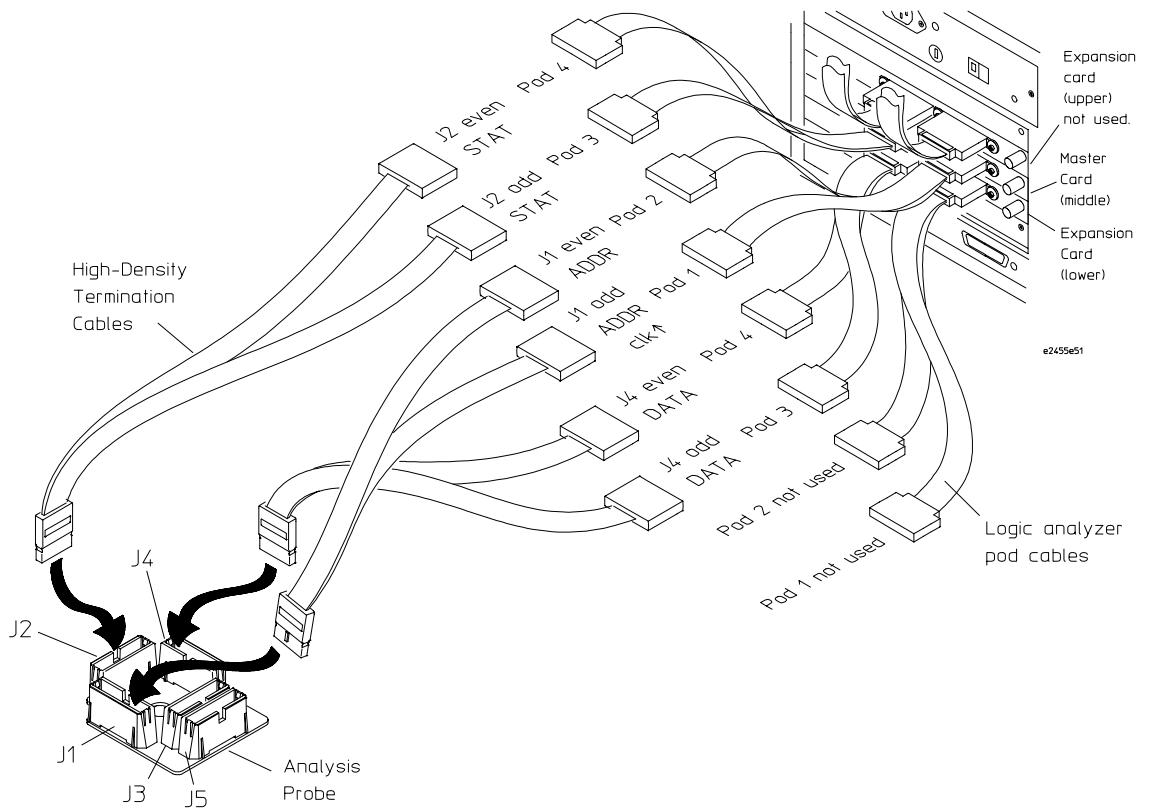
Chapter 3: Connecting and Configuring the Analysis Probe
Connecting the Analysis Probe to the Logic Analyzer

Three-card 64-bit HP 16715/16/17 connections (continued)



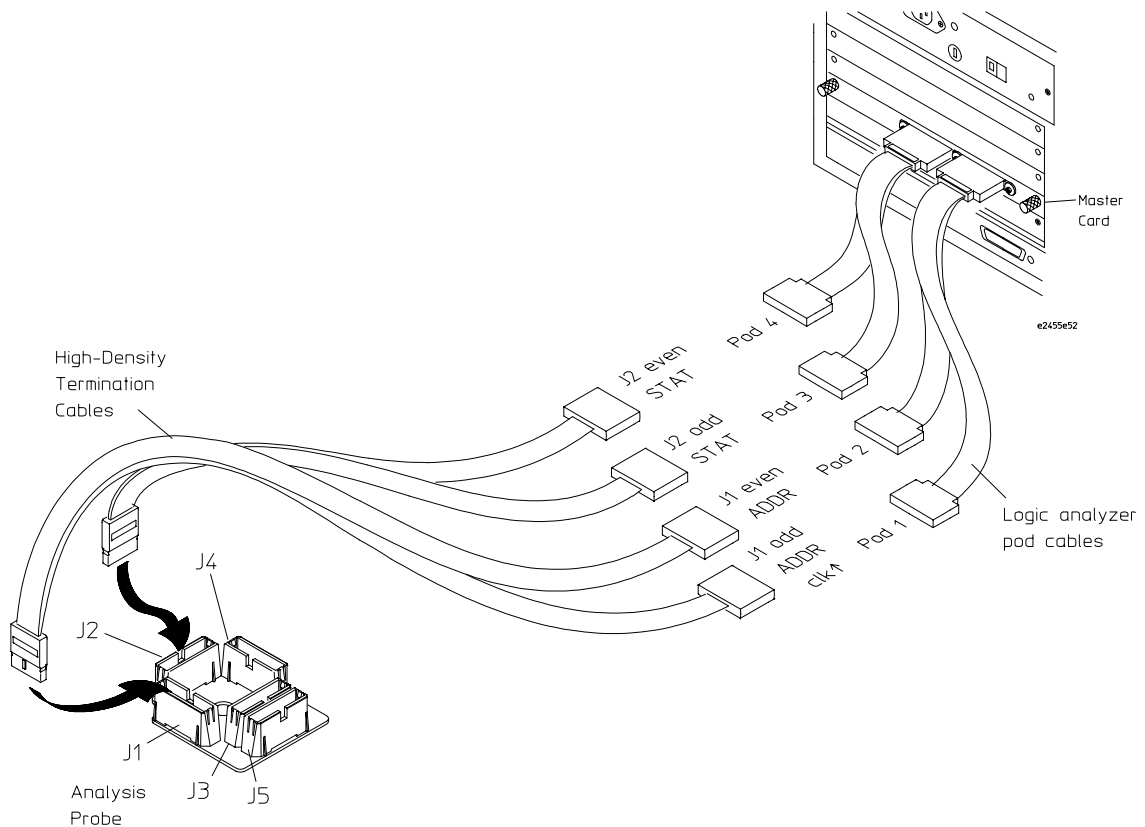
To connect to the HP 16715/16/17A (two or three-card) for 32-bit analysis

Use the figure below to connect the analysis probe to the two or three-card HP 16715/16/17A logic analyzers for 32-bit data analysis.



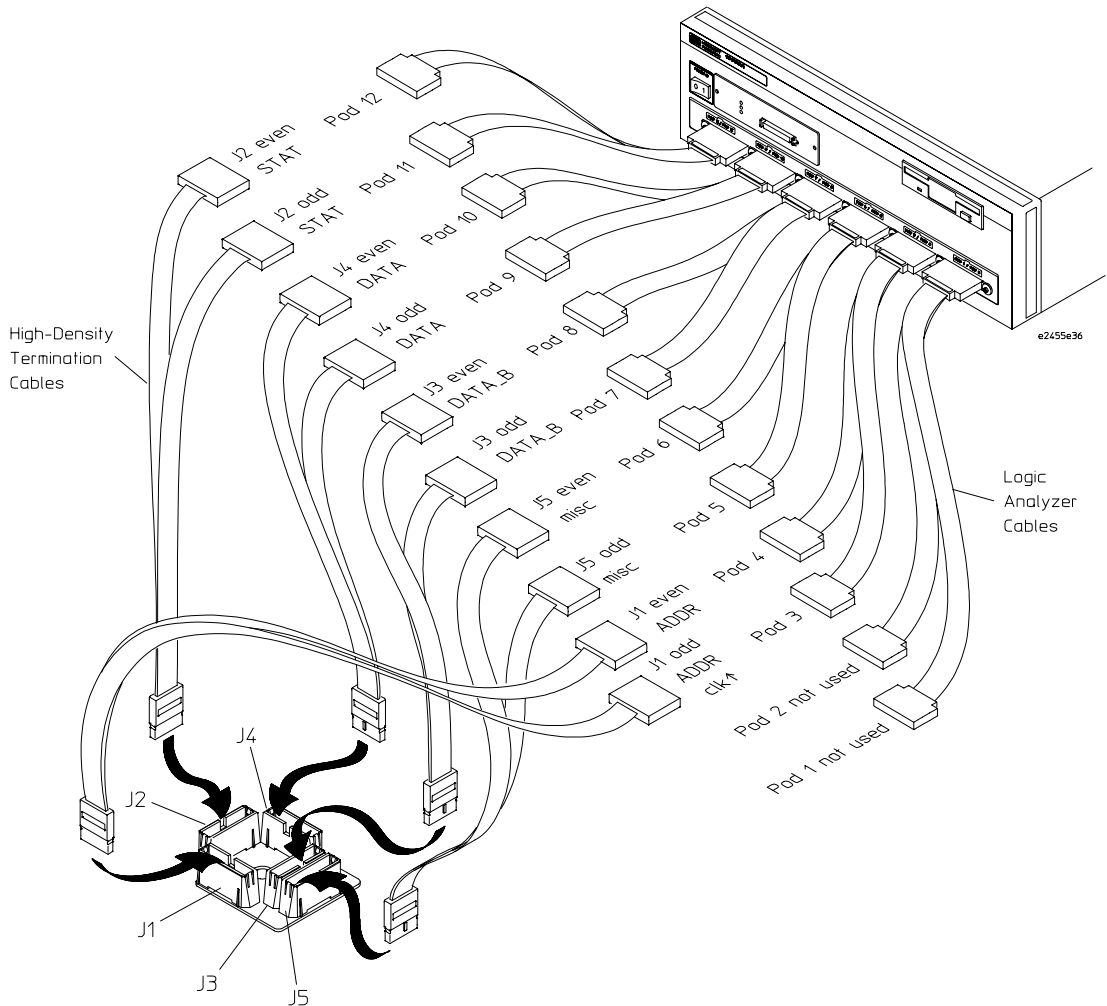
To connect to the HP 16715/16/17A (one-card) for no-data analysis

Use the figure below to connect the analysis probe to the one-card HP 16715/16/17A logic analyzers for no-data analysis.



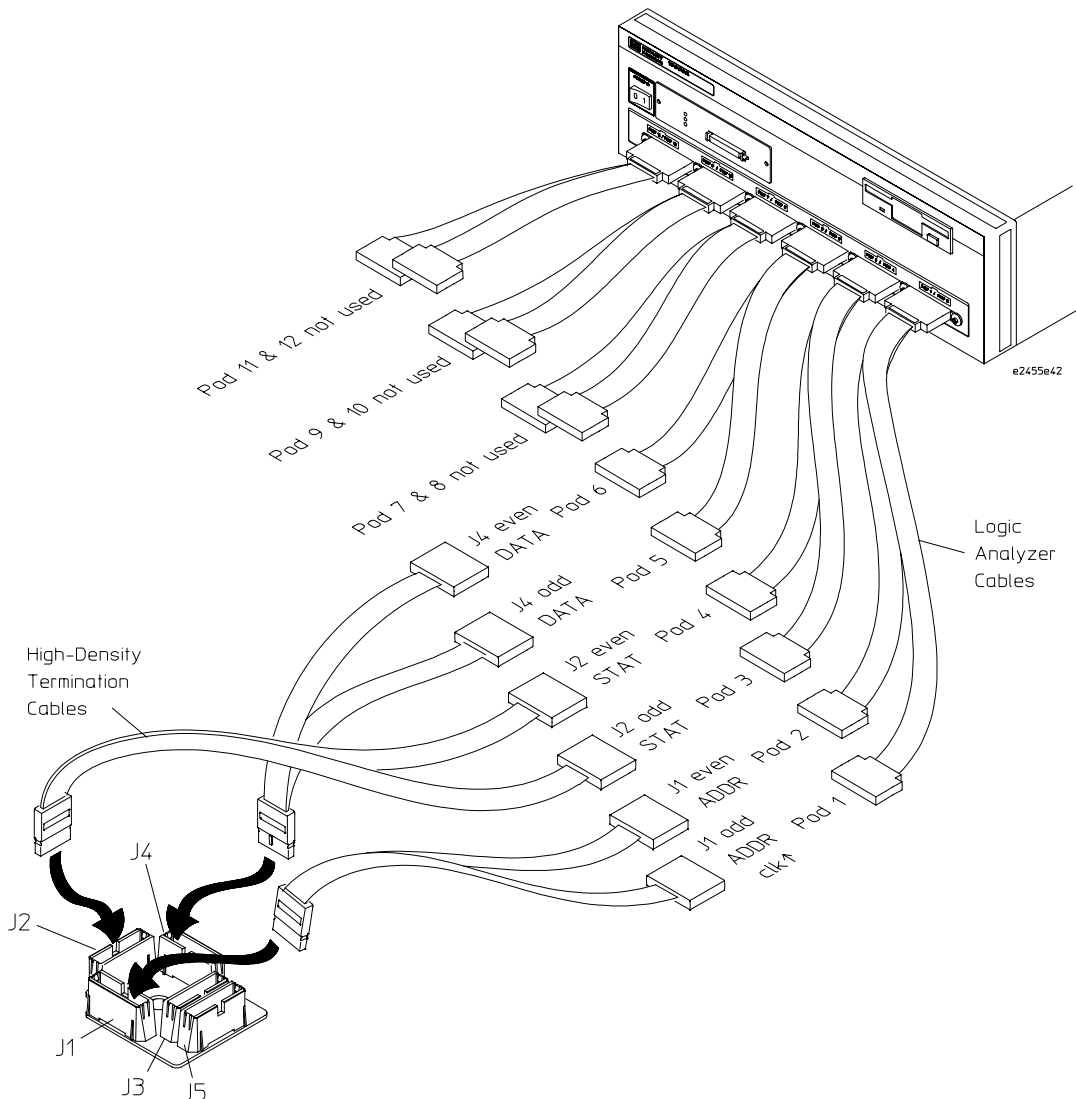
To connect to the HP 16600A logic analyzer for 64-bit data analysis

Use the figure below to connect the analysis probe to the HP 16600A logic analysis system.



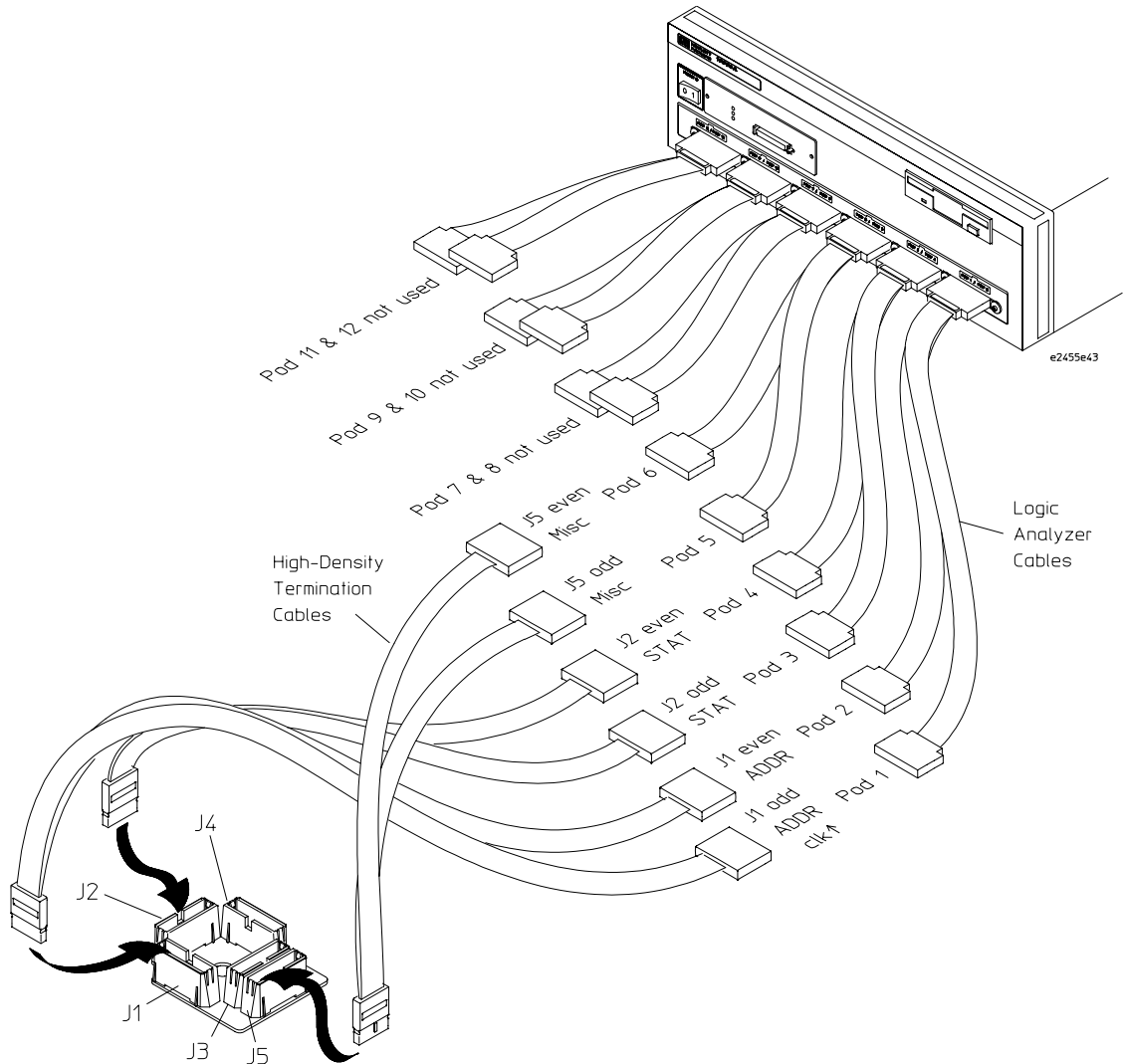
To connect to the HP 16600A logic analyzer for 32-bit data analysis

Use the figure below to connect the analysis probe to the HP 16600A logic analysis system for 32-bit data analysis.



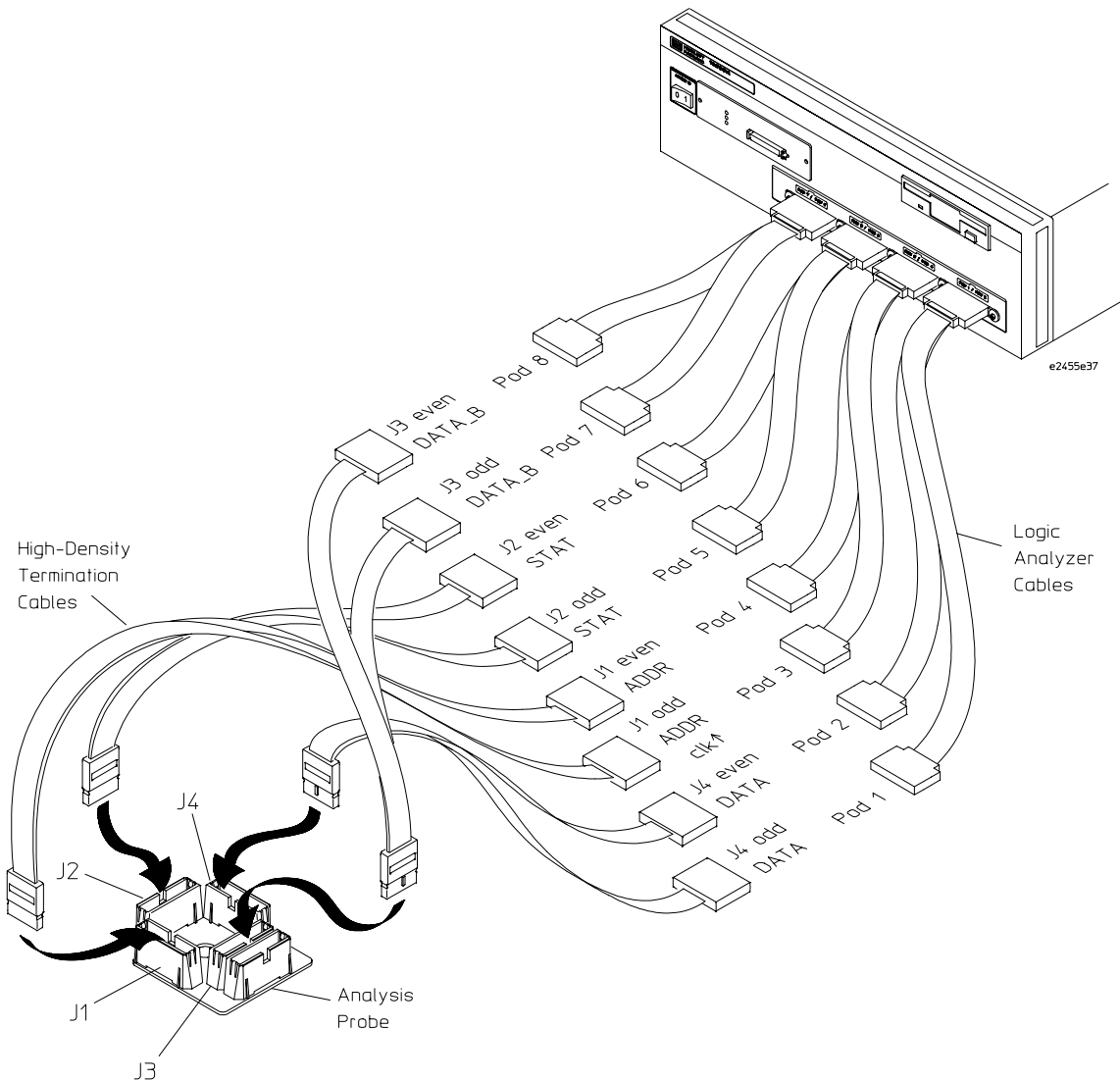
To connect to the HP 16600A logic analyzer for no-data analysis

Use the figure below to connect the analysis probe to the HP 16600A logic analysis system for no-data analysis.



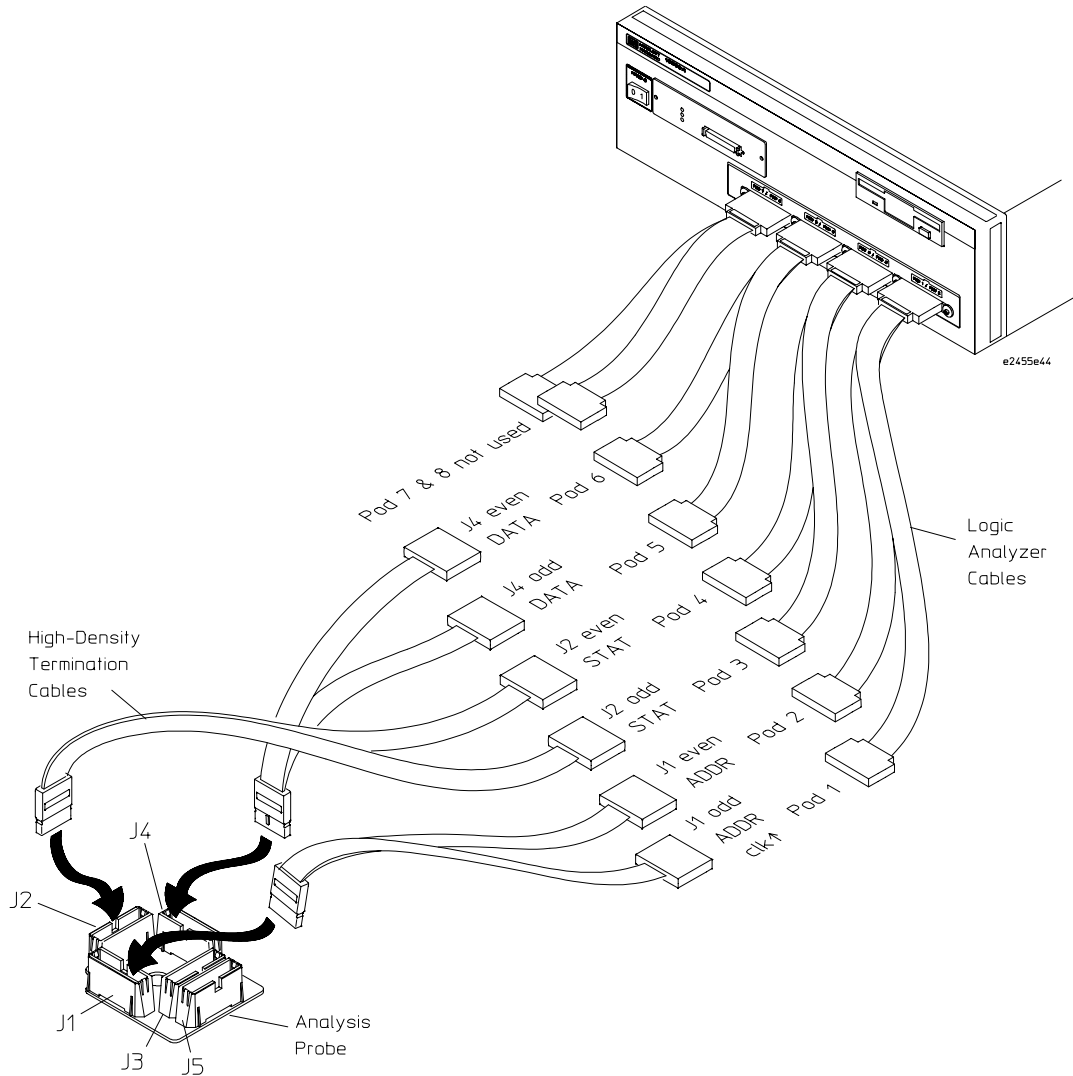
To connect to the HP 16601A logic analyzer for 64-bit data analysis

Use the figure below to connect the analysis probe to the HP 16601A logic analysis system for 64-bit data analysis.



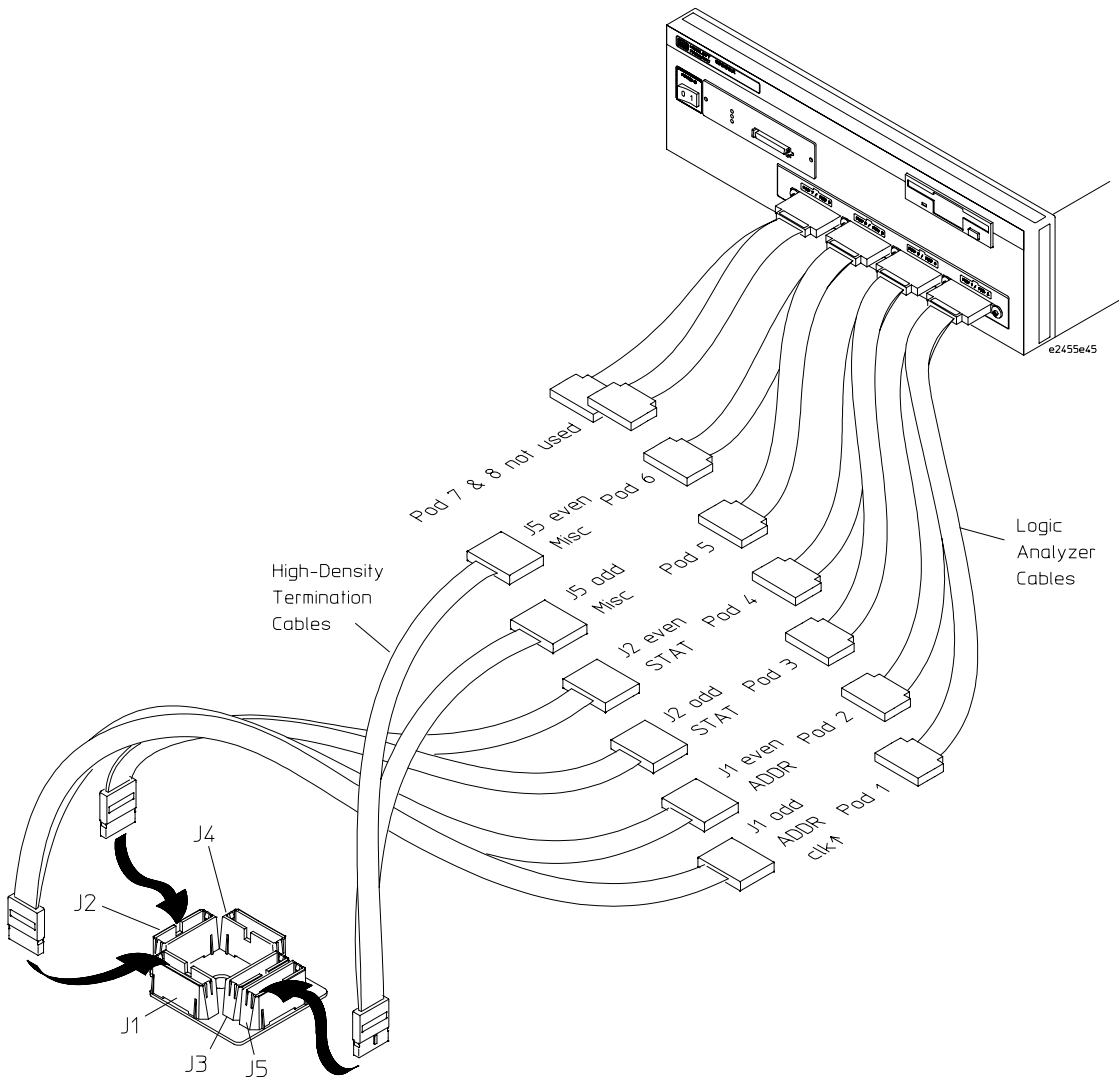
To connect to the HP 16601A logic analyzer for 32-bit data analysis

Use the figure below to connect the analysis probe to the HP 16601A logic analysis system for 32-bit data analysis.



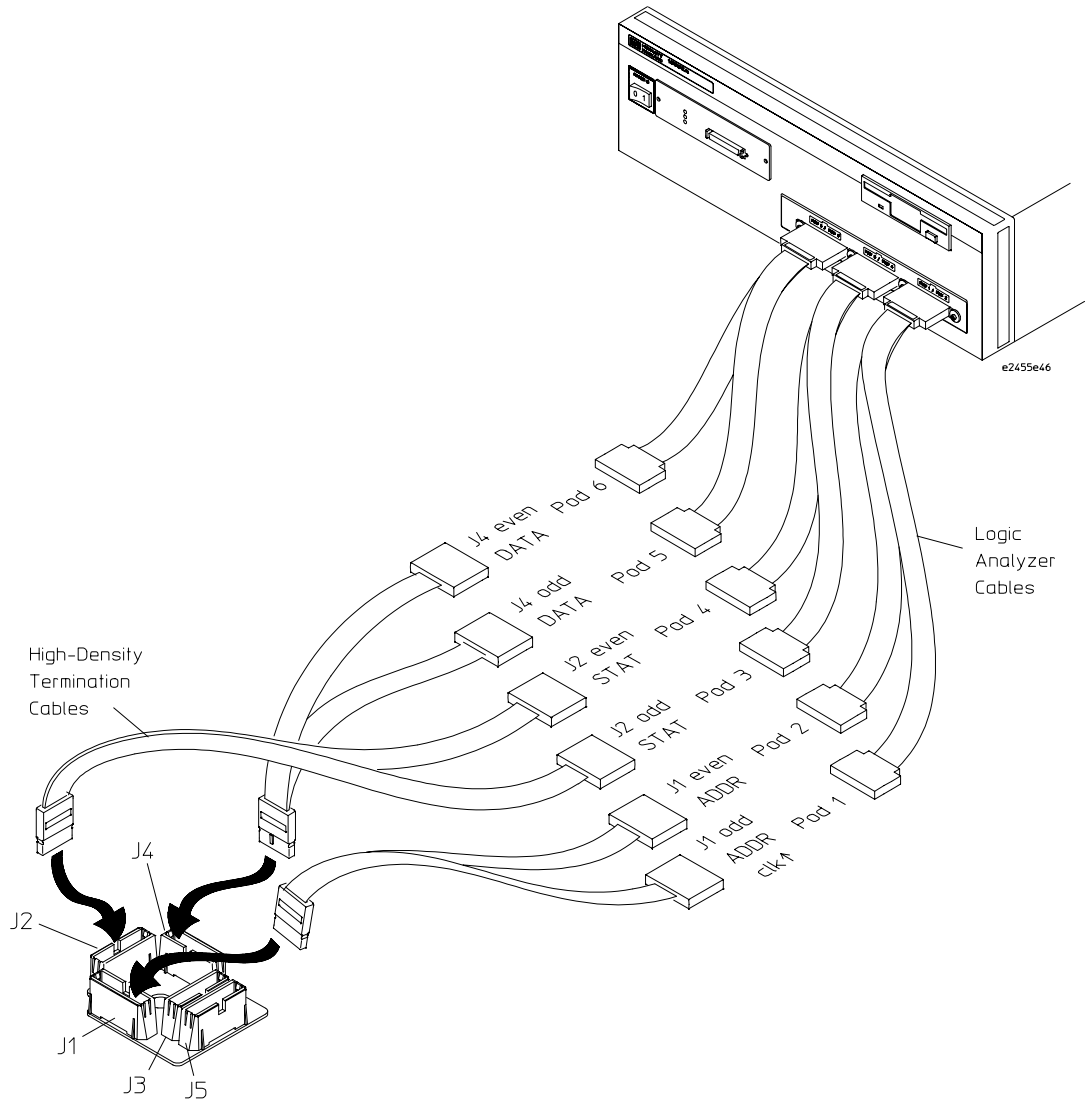
To connect to the HP 16601A logic analyzer for no-data analysis

Use the figure below to connect the analysis probe to the HP 16601A logic analysis system for no-data analysis.



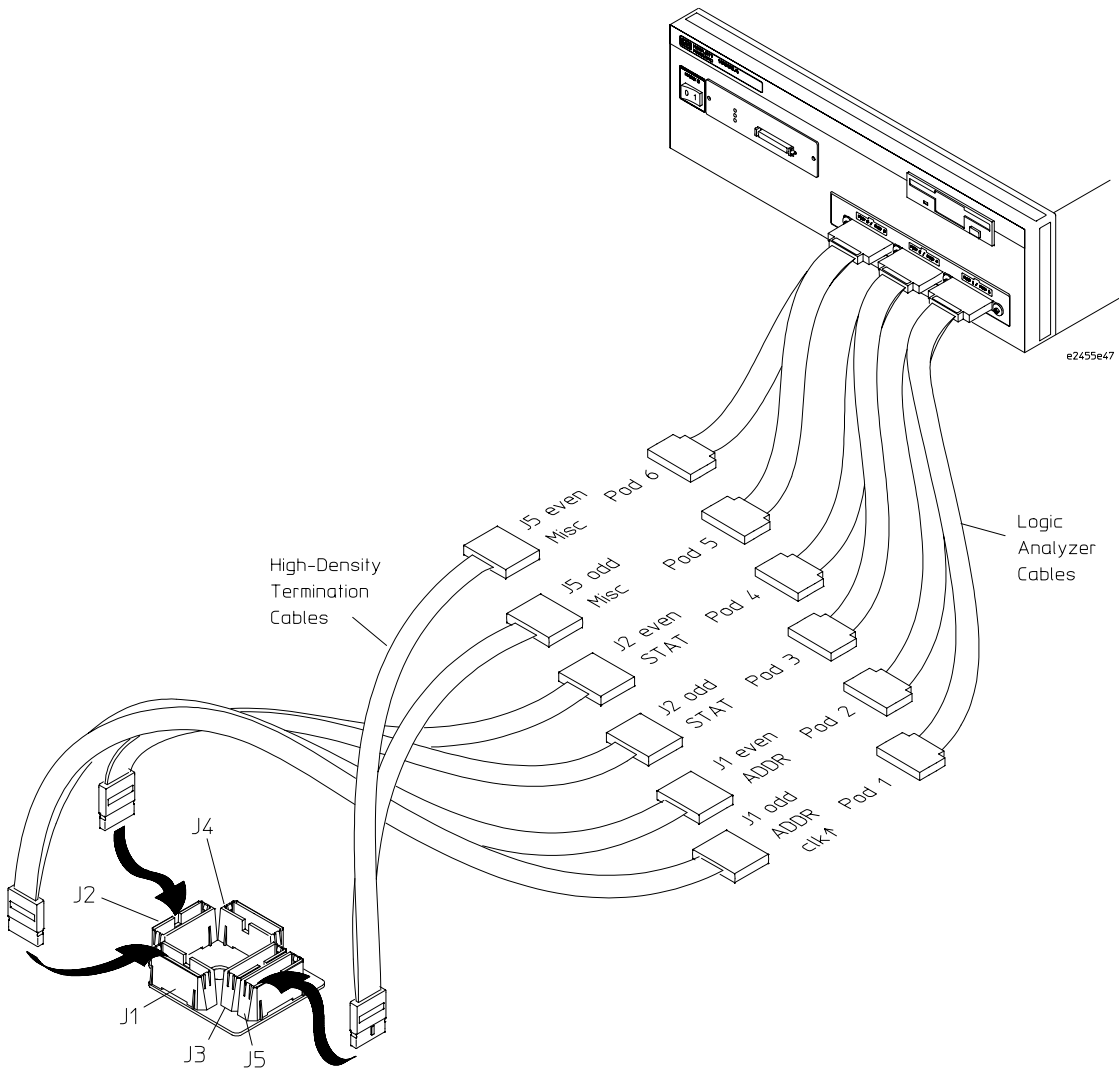
To connect to the HP 16602A logic analyzer for 32-bit data analysis

Use the figure below to connect the analysis probe to the HP 16602A logic analysis system for 32-bit data analysis.



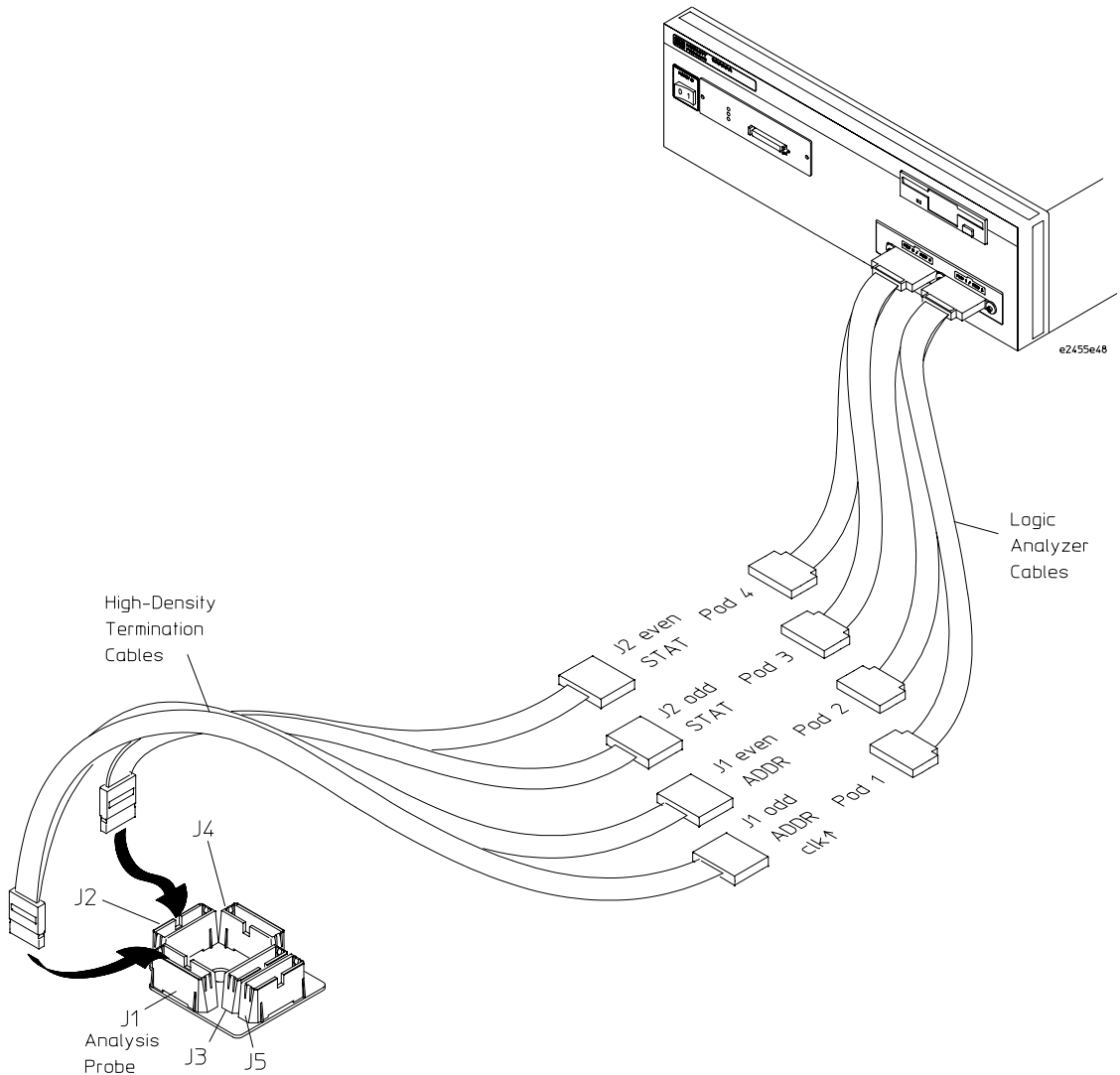
To connect to the HP 16602A logic analyzer for no-data analysis

Use the figure below to connect the analysis probe to the HP 16602A logic analysis system for no-data analysis.



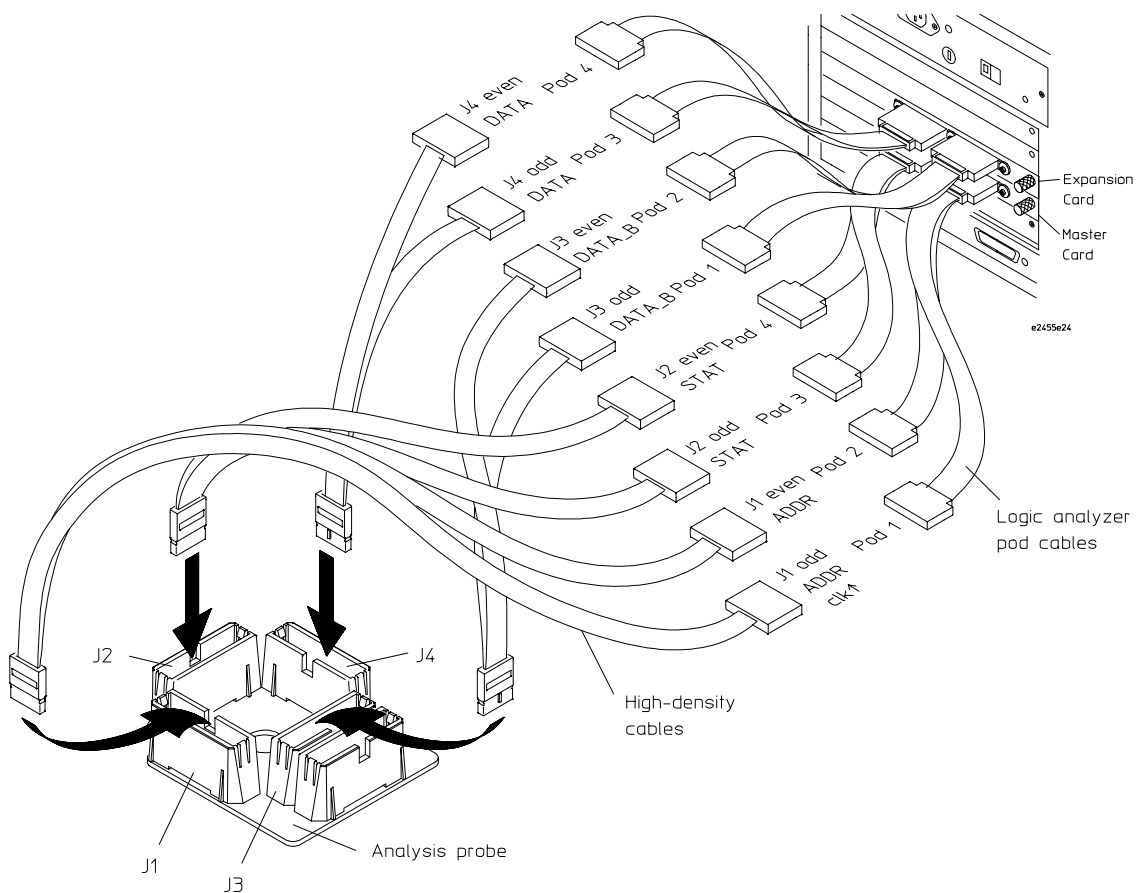
To connect to the HP 16603A logic analyzer for no-data analysis

Use the figure below to connect the analysis probe to the HP 16603A logic analysis systems for no-data analysis.



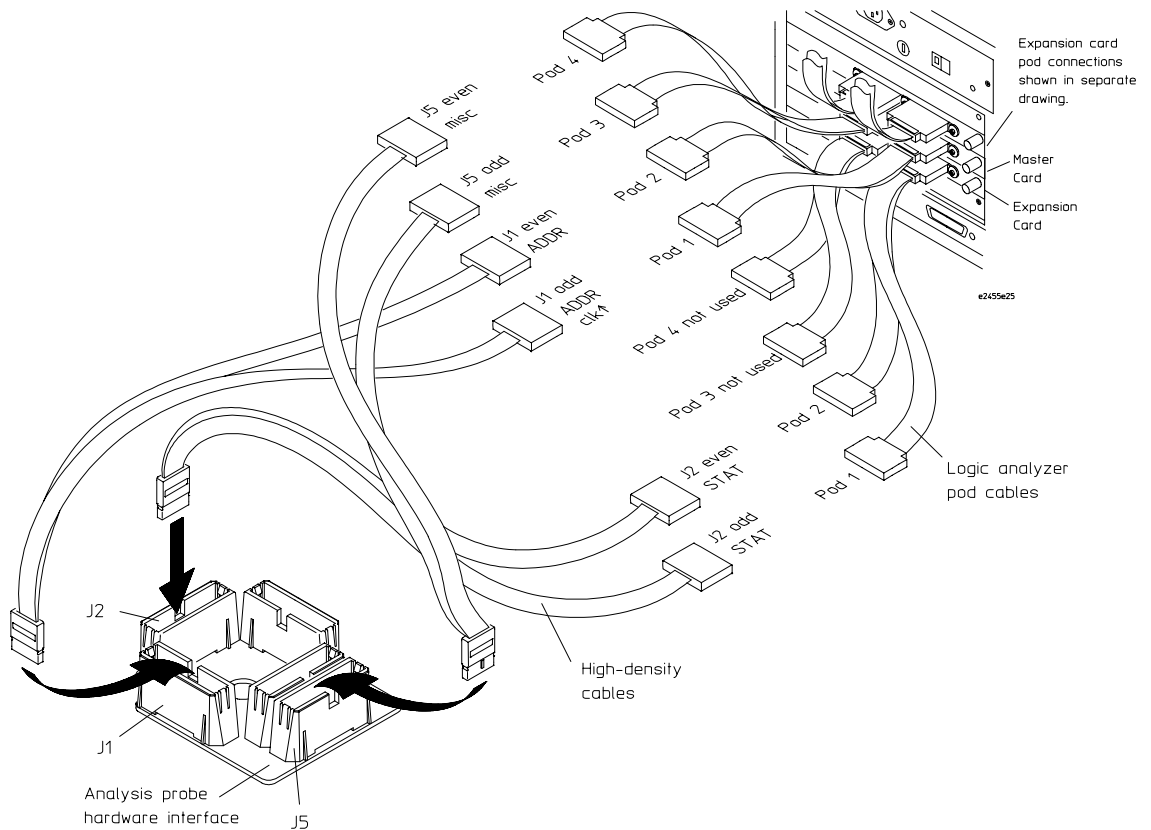
To connect to the HP 16554/55/56/57 (two-card) for 64-bit data analysis

Use the figure below to connect the analysis probe to the two-card
HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers.



To connect to the HP 16554/55/56/57 (three-card) for 64-bit data analysis

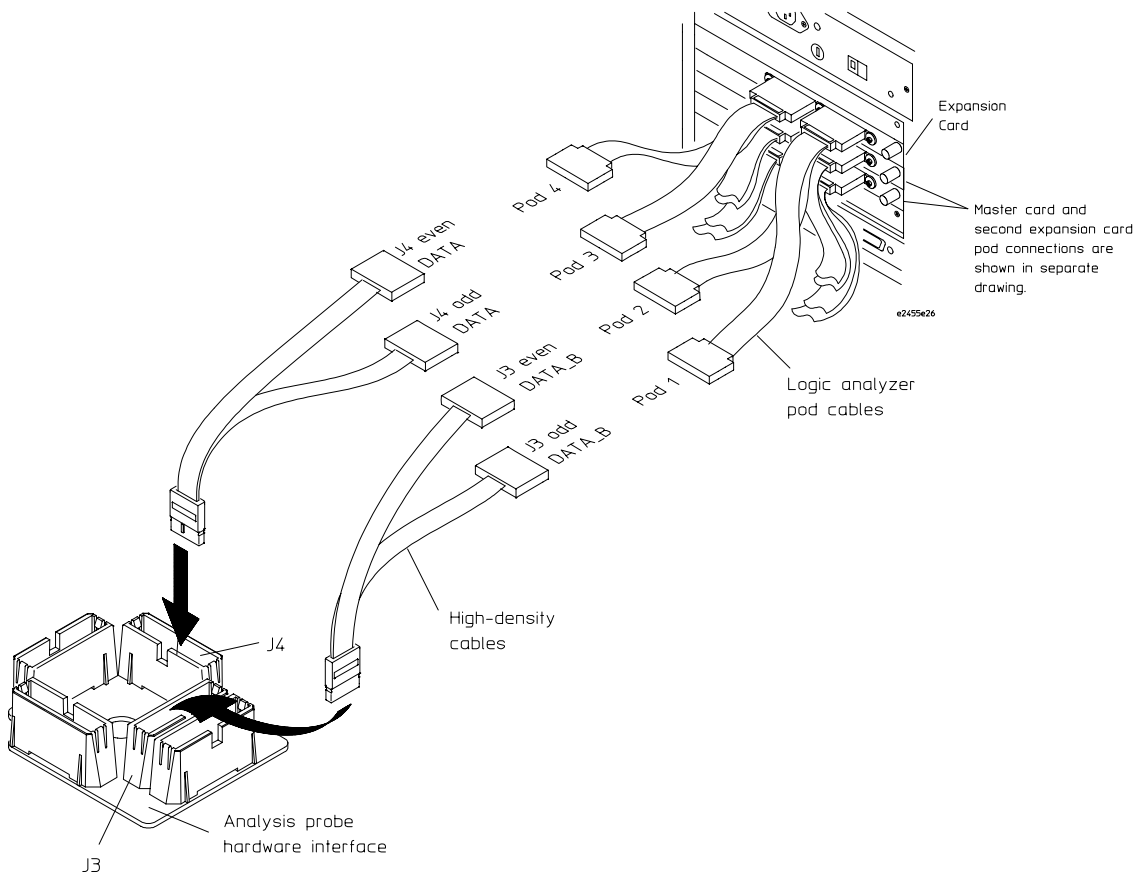
Use the figures below and on the following page to connect the analysis probe to the three-card HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers for 64-bit data analysis.



Continued on next page

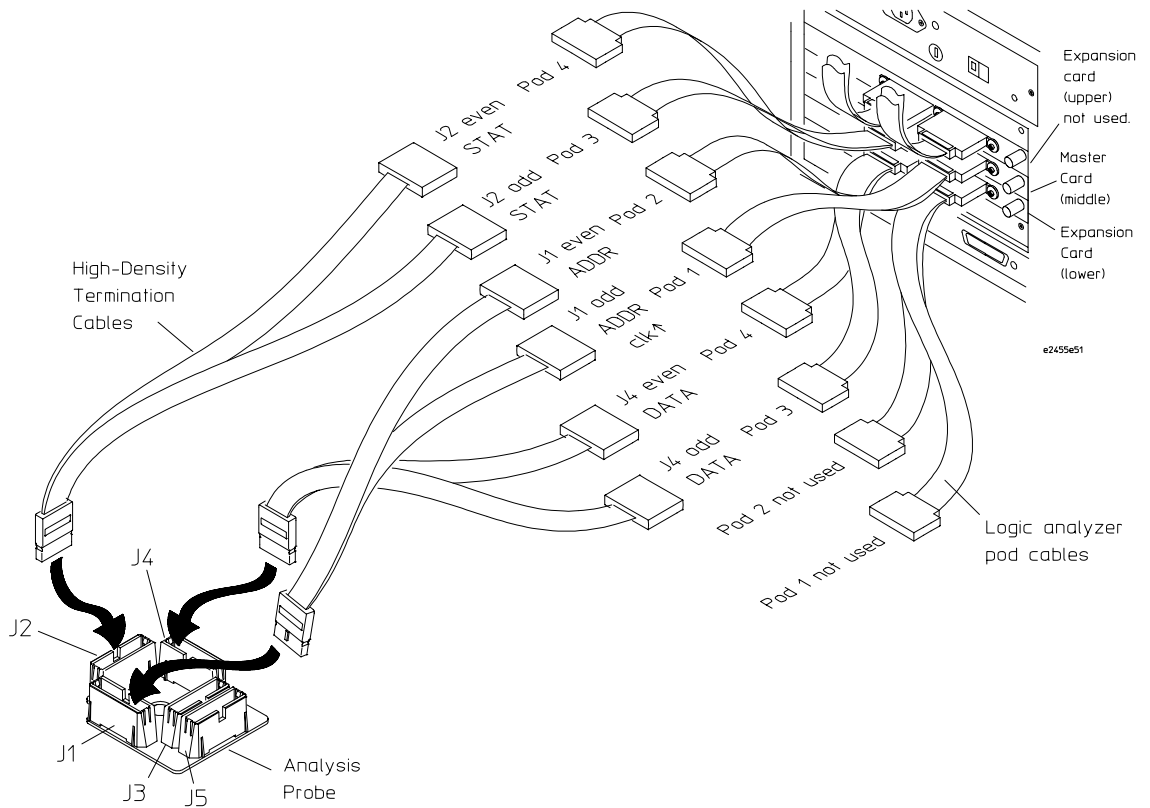
Chapter 3: Connecting and Configuring the Analysis Probe
Connecting the Analysis Probe to the Logic Analyzer

Three-card 64-bit HP 16554/55/56/57 connections (continued)



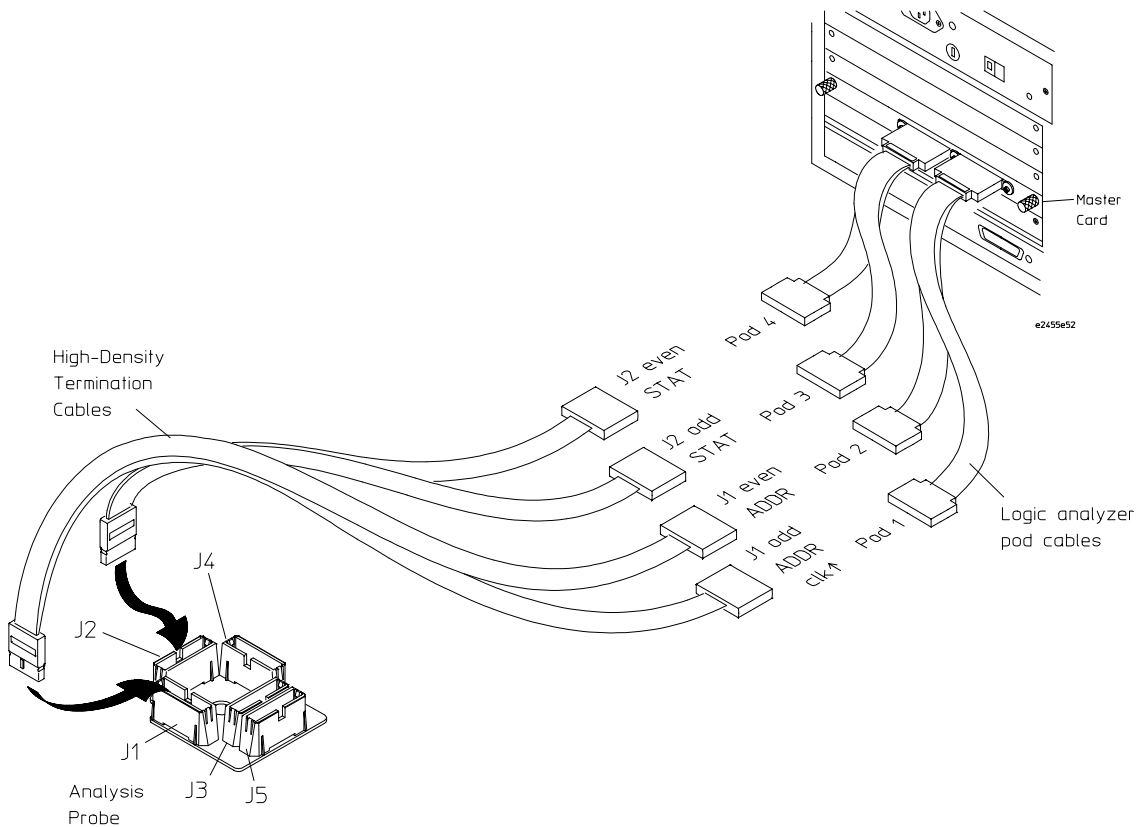
To connect to the HP 16554/55/56/57 (two or three-card) for 32-bit analysis

Use the figure below to connect the analysis probe to the two or three-card HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers for 32-bit data analysis.



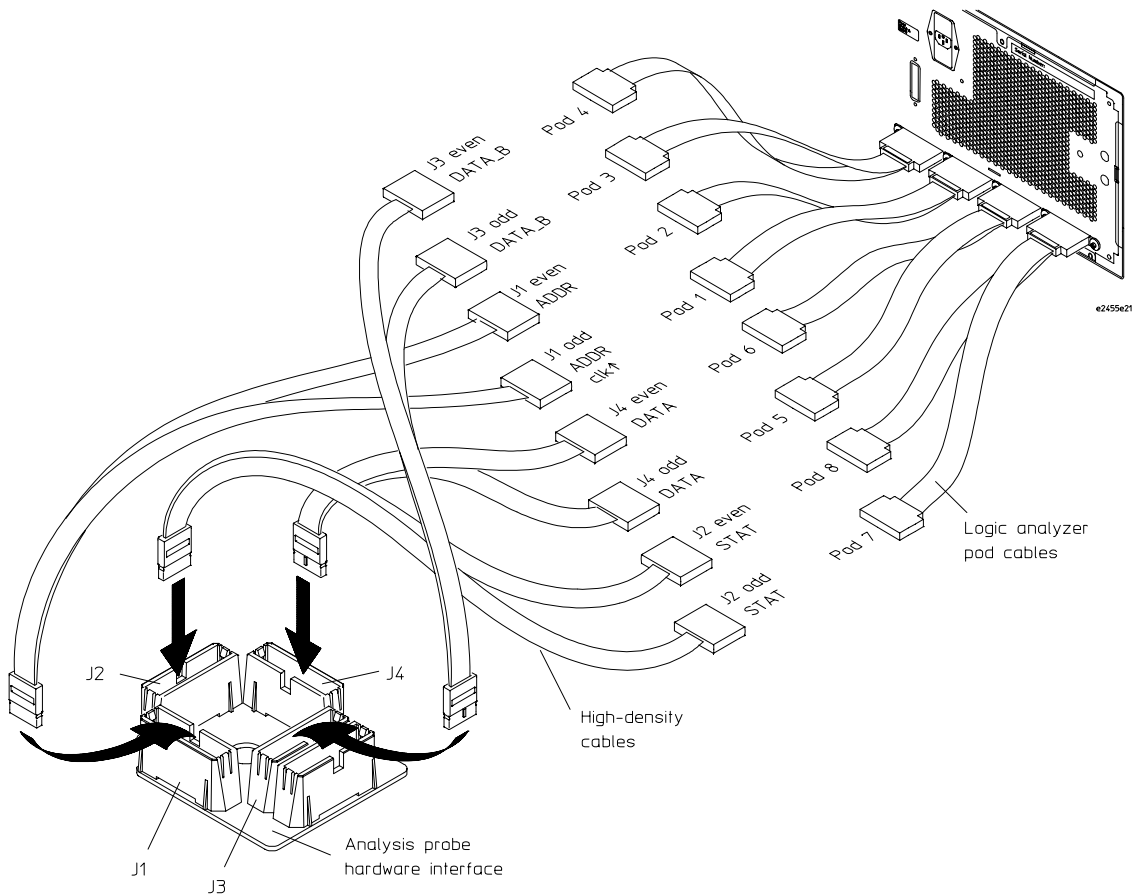
To connect to the HP 16554/55/56/57 (one-card) for no-data analysis

Use the figure below to connect the analysis probe to the one-card
HP 16554A/55A/56A and HP 16555D/56D/57D logic analyzers for no-data
analysis.



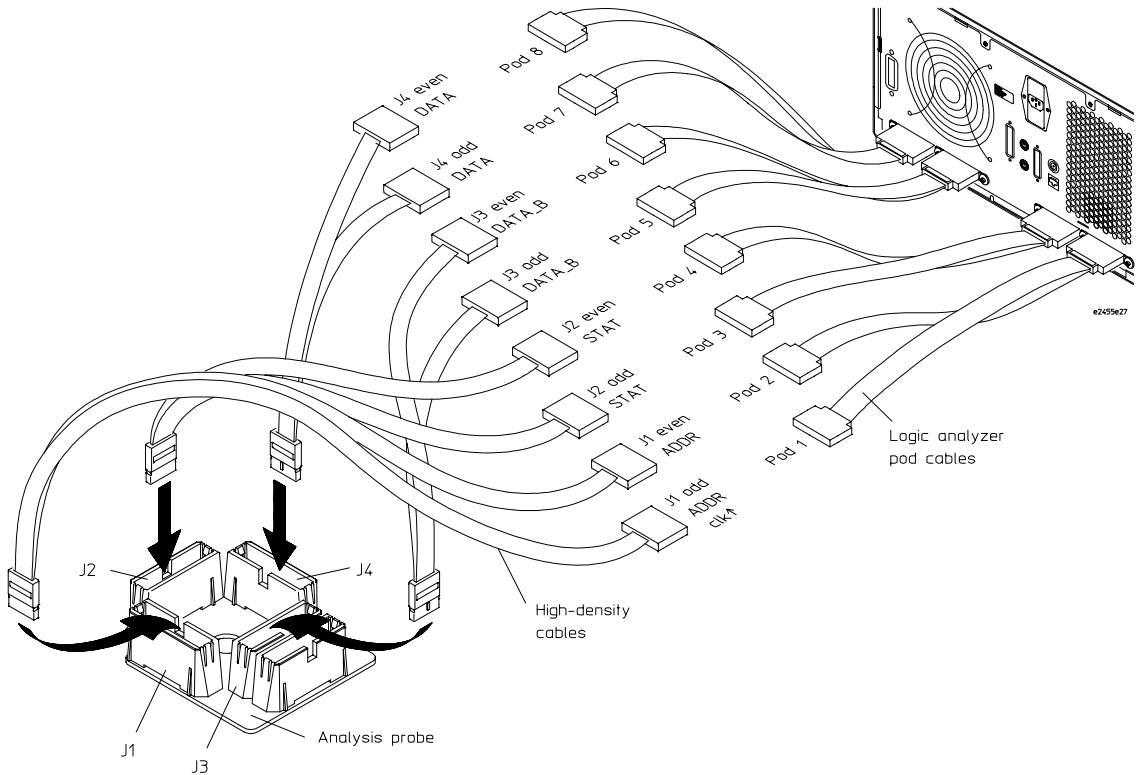
To connect to the HP 1660A/AS/C/CS/CP/E/ES/EP logic analyzers for 64-bit data analysis

Use the figure below to connect the analysis probe to the HP 1660A/AS/C/CS/CP/E/ES/EP logic analyzers for 64-bit analysis.



To connect to the HP 1670A/D/E logic analyzers for 64-bit data analysis

Use the figure below to connect the analysis probe to the HP 1670A/D/E logic analyzers for 64-bit data analysis. Configuring the Logic Analysis System



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 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.

If you are using the HP 1660-series, HP 1670-series, or HP 16500B/C mainframe, you must also determine whether your target system is “Delayed AACK” or “pipelined” (see below).

Delayed AACK

In “Delayed AACK” target systems, the memory system delays AACK until completion of the data phase. Use configuration files C603M, C603M3, C603J, or C603F for “Delayed AACK” target systems.

Pipelined

In “Pipelined” target systems, the microprocessor may assert AACK before completion of the data phase. Use configuration files CP603M, CP603M3, CP603J, or CP603F for “Pipelined” target systems.

To load configuration and inverse assembler files into HP 16600A/700A logic analysis systems from the system hard disk

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/ppc6xx/ppc603` exists.

<p>If the above directory does not exist, you need to install the POWERPC6XX Processor Support Package. Close File Manager, then use the procedure on the CD-ROM jacket to install the POWERPC6XX Processor Support Package before you continue.</p>

- 2 Using File Manager, select the configuration file you want to load in the `/hplogic/configs/hp/ppc6xx/ppc603` 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 PPC603 analysis by loading the appropriate PPC603 configuration file. Loading the indicated file also automatically loads the inverse assembler. The configuration file names are located at the bottom of the table showing the connections for your particular logic analyzer. They are also shown in the following table.

- 3 Close File Manager.

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

The preferred method is to install this functionality from the CD-ROM onto the hard disk and load from the hard disk, also described in this manual.

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 “*”.

Chapter 3: Connecting and Configuring the Analysis Probe

Configuring the Logic Analysis System

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 PPC603 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 appropriate disk in the front disk drive of the logic analyzer.

There are two disks provided for the PowerPC 603. The “Delayed AACK” disk is for target systems in which the memory system delays AACK until the completion of the data phase. The “Pipelines” disk is for target systems that may assert AACK before the completion of the data phase.

- 2 Go to 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.

There are two types of configuration files available for the PowerPC 603. The “Delayed AACK” files (C603M, C603M3, C603J, and C603F) are for target

systems in which the memory system delays AACK until completion of the data phase. The “Pipelined” files (CP603M, CP603M3, CP603J, and CP603F) are for target systems that may assert AACK before completion of the data phase.

The configuration file names are located at the bottom of the table showing the connections for your particular logic analyzer. They are also shown in the table on the following page.

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

The logic analyzer is configured for PPC603 analysis by loading the appropriate PPC603 configuration file. Loading the indicated file also automatically loads the inverse assembler.

7 If you are using the HP 16505A Prototype Analyzer, insert the “16505 Prototype Analyzer” flexible disk 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 HP E2455B.

Chapter 3: Connecting and Configuring the Analysis Probe
Configuring the Logic Analysis System

Logic Analyzer Configuration Files

Analyzer Model	Analyzer Module Description	Delayed AACK Configuration File	32-bit Configuration File	No-data Configuration File	Pipelined Config. File (16500, 1660, 1670 only)
16710/11/12A (1 or 2 cards)	100 MHz STATE 250 MHz TIMING	C603F	C603F1	C603F0	na
16715/16/17A (1 or 2 card)	167 MHz STATE 333/667 MHz TIMING	C603L	C603L	C603L0	na
16715/16/17A (3 card)	167 MHz STATE 333/667 MHz TIMING	C603L3	C603L	C603L0	na
16600A	na	C603F	C603F1	C603F0	na
16601A	na	C603F	C603F1	C603F0	na
16602A	na	C603F	C603F1	C603F0	na
16603A	na	C603F	C603F1	C603F0	na
16550A (1 or 2 card)	100 MHz STATE 500 MHz TIMING	C603F	C603F1	C603F0	CP603F2*
16554A (1 or 2 card)	0.5M SAMPLE 70/250 MHz LA	C603M	C603M	C603M0	CP603M2*
16555A/D (1 or 2 card)	1.0M SAMPLE 110/500 MHz LA	C603M	C603M	C603M0	CP603M2*
16556A/D (1 or 2 card)	1.0M SAMPLE 100/400 MHz LA	C603M	C603M	C603M0	CP603M2*
16557D (1 or 2 card)	1.0M SAMPLE 100/400 MHz LA	C603M	C603M	C603M0	CP603M2*
16554A (3 card)	0.5M SAMPLE 70/250 MHz LA	C603M3	C603M	C603M0	CP603M3
16555A/D (3 card)	1.0M SAMPLE 110/500 MHz LA	C603M3	C603M	C603M0	CP603M3
16556A/D (3 card)	1.0M SAMPLE 100/400 MHz LA	C603M3	C603M	C603M0	CP603M3
16557D (3 card)	1.0M SAMPLE 100/400 MHz LA	C603M3	C603M	C603M0	CP603M3
1660A/AS/C/CS/CP/E/ES/EP	na	C603J	C603F1	C603F0	CP603J
1670A/D/E	na	C603M	C603F1	C603F0	CP603M

* Two cards in 16500B/C mainframes only.

Designing Connectors for
Custom Probing for the
Inverse Assembler

Chapter 4: Designing Connectors for Custom Probing for the Inverse Assembler

This chapter shows you how to design logic analyzer connectors on your target system for use with the inverse assembler software.

If you are using an HP E2455B analysis probe, skip this chapter.

This chapter consists of the following sections:

- Using the General Purpose (GP) probes
- Designing logic analyzer connectors on your target system for the PowerPC 603 Inverse Assembler Software

Direct Probing with GP Probes

If you are using general-purpose (GP) probes, connect the individual probes to the signals according to the signal-to-connector mapping tables, as shown in Chapter 11, “Hardware Reference.”. Use the figures in Chapter 3, “Connecting and Configuring the Analysis Probe,” to determine which logic analyzer pods to use for the signal groups.

It is helpful to label the probe headers before installing the probes. You should connect the ground signal for the analyzer clock(s), and at least two to four signal grounds per pod. For optimal signal integrity, connect a ground for each signal probed.

Designing and Using Built-in Connectors

You can design analyzer-compatible connectors into the target board, and connect the logic analyzer cables to these connectors according to the figures in chapter 3. The primary concerns when using built-in connectors are:

- The board real estate required by the connectors
- Ensuring that the logic analyzer connection is properly terminated
- Ensuring that the microprocessor pins connect to the proper logic analyzer probes. See the “Hardware Reference” chapter for pinouts.

A number of connector schemes are available. These schemes are described in detail on the following pages. A brief summary of these schemes is covered in the following table.

Summary of Built-in Connectors	
2x20 Connectors	low density requires on-board RC termination plugs directly into logic analyzer cables
2x10 Connectors	medium density requires termination adapter (HP part number 01650-63203) not recommended above 50 MHz
2x19 Connectors (AMP Mictor 38 Connectors)	highest density (each connector supports two logic analyzer pods) requires HP E5346A cables

Some of these schemes require an RC termination network on board for each probed signal. The terminations are available in SIP (single inline package), DIP (double inline package), and SMT (surface mount technology) packages. Additional information is available in HP Application Notes 1244-1 and 5966-4167E.

AMP Mictor 38 connectors

The connection scheme shown in this section uses 38-pin connectors on the target system, and high-density termination cables to connect to the logic analyzer. Each Mictor 38 connector carries 32 signals plus two clocks (CLK1 for two logic analyzer pods). The high-density termination cables are required to connect the logic analyzer cables to the connector (HP part number E5346A). These cables contain the required termination. One cable is required for every two logic analyzer pods.

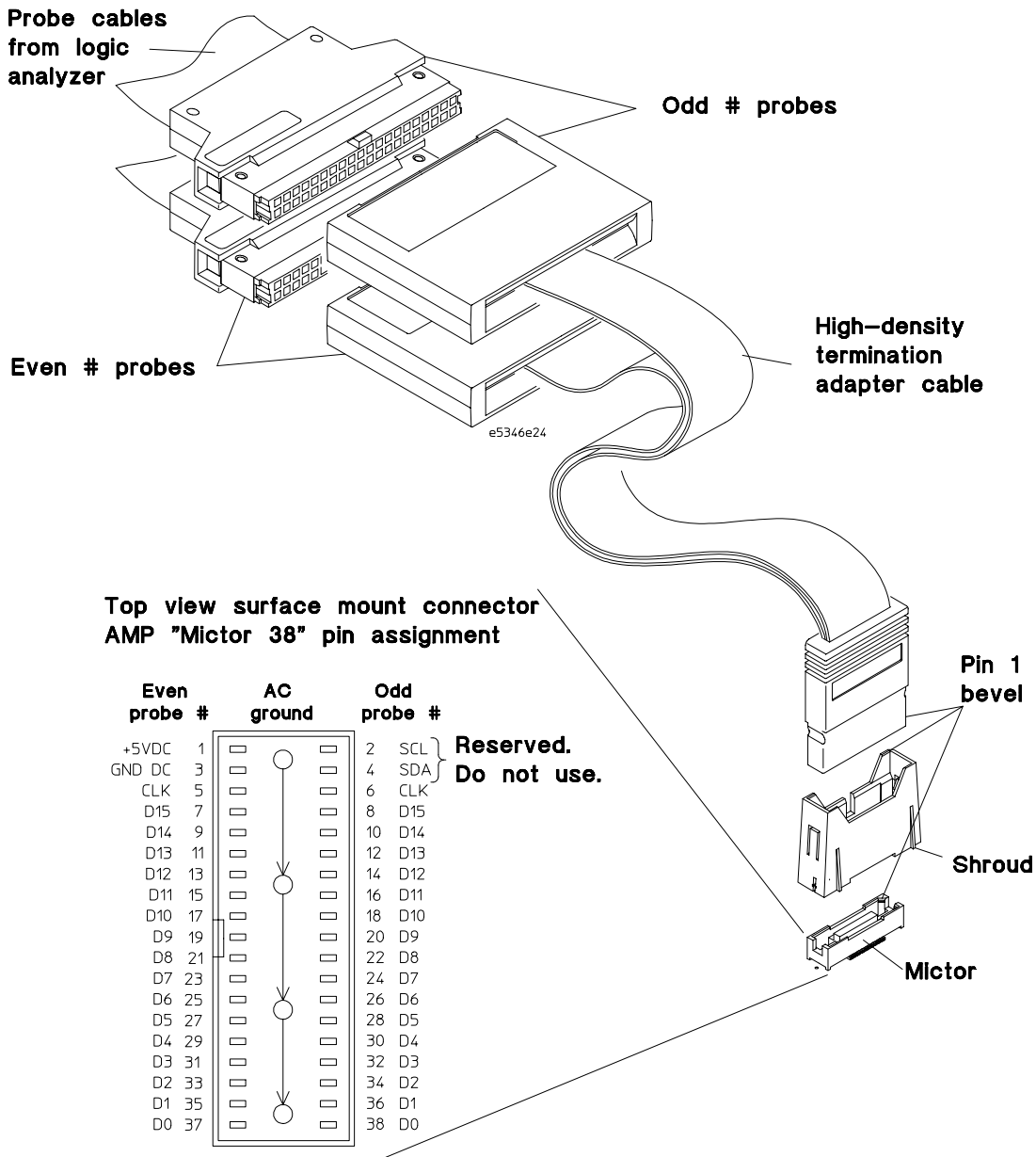
The part numbers for built-in connectors and cables are shown below. An illustration of the components is shown on the following page. Refer to the “Hardware Reference” chapter (page 259) for the tables showing the microprocessor signals for each pin. Note that the +5V pin (pin 1) is used to supply power from the logic analyzer to any active devices on an interface board. In most instances, this pin should not be used.

To increase the structural support for the cables, you can also use cable support shrouds (HP part number E5346-44701) on each connector. The figures on the following page show the mechanical layouts for the shrouds and headers.

Part Numbers for Built-in Connectors and Cables

Part Number	Description
HP 1252-7431, or AMP 2-767004-2	2 x 19 header. A minimum of three connectors (six logic analyzer pods) is required; up to six may be used.
HP E5346-44701	Optional connector-support shroud
HP E5346A	High-density termination cable. One required for each 2x19 connector.

Chapter 4: Designing Connectors for Custom Probing for the Inverse Assembler Designing and Using Built-in Connectors

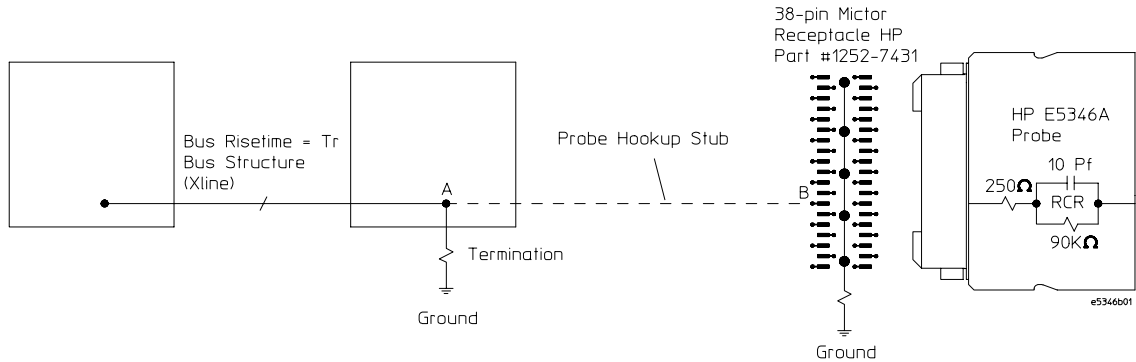


Connectors, Shroud, and High-density Termination Cables

Design Considerations

The connector must be located close enough to the target signal so that the stub length created is less than $\frac{1}{5}$ the T_r (bus risetime, see figure below). For PC board material, ($\epsilon_r = 4.9$) and Z_0 in the range of $50 - 80\Omega$, use a propagation delay of 160 ps/inch of stub.

Each probed signal line must be able to supply a minimum of 600 mV to the probe tip and handle a minimum of $90\text{ k}\Omega$ shunted by 10 pF. The maximum input voltage to the high-density cables is $\pm 40\text{V}$ peak.

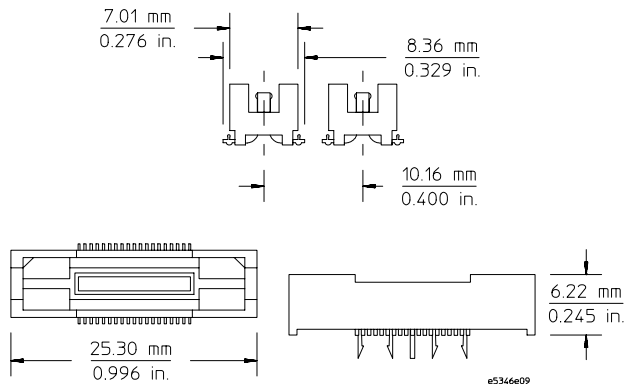
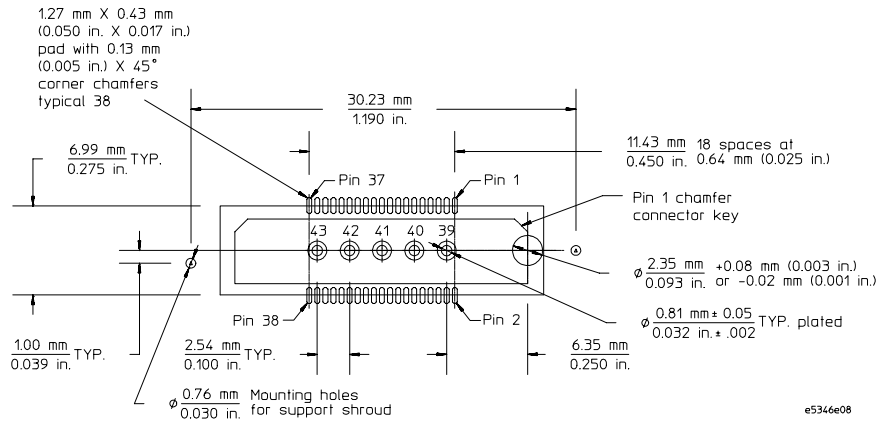


2x19 Header Design Rules

Chapter 4: Designing Connectors for Custom Probing for the Inverse Assembler Designing and Using Built-in Connectors

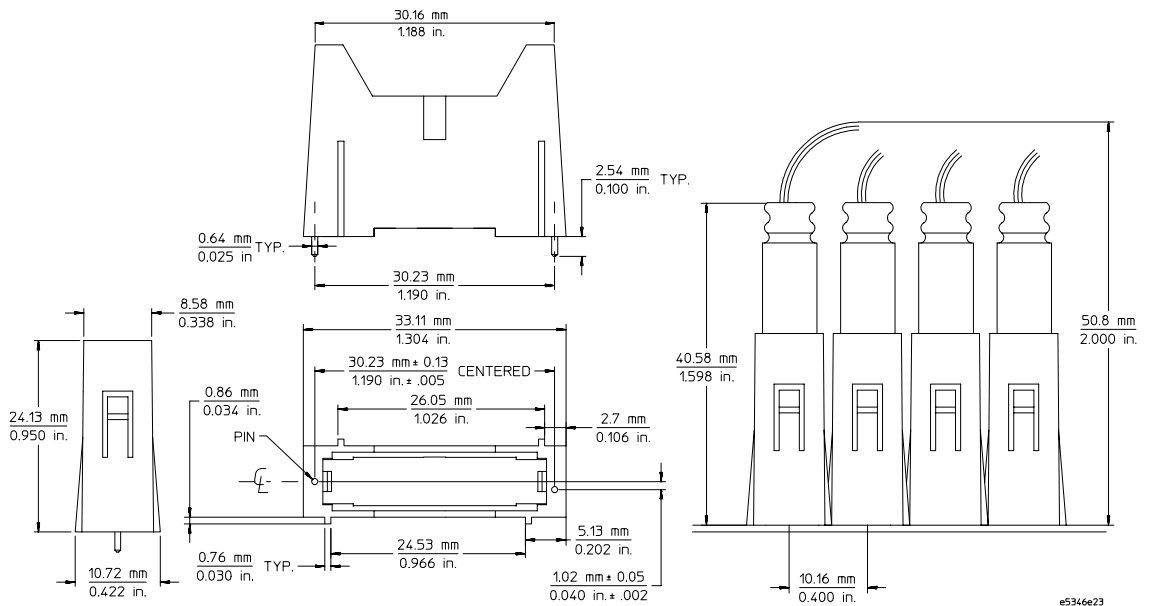
Support Shroud

The support shroud (HP part number E5346-44701) provides additional strain relief between the connectors and the high-density termination cables. The shroud requires two through-hole connections to the target board. It fits around the header, and mounts directly to the target board. The following figures show the mechanical connections for the shrouds and connectors.



Support Shroud Mechanical Information

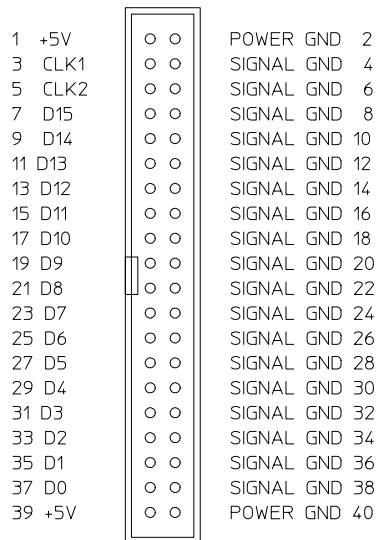
Chapter 4: Designing Connectors for Custom Probing for the Inverse Assembler
Designing and Using Built-in Connectors



2x19 Header Mechanical Information

2x20 headers

The 2x20 headers are low-density connectors. Each connector plugs directly into a standard logic analyzer cable, and carries 16 microprocessor signals and a clock (CLK1). This connection scheme requires on-board RC termination. The figure below shows the pinout for a 2x20 header. Refer to the “Hardware Reference” chapter (page 259) for the tables showing the microprocessor signals for each pin. Note that the +5V pins (1 and 39) supply power from the logic analyzer to active devices on an interface board. In most instances, these pins should not be used.



2x20 Header e2449b01

Pinout for 2x20 header

The following table contains part numbers for the components required for this connection scheme.

Part Numbers for Built-in Connectors			
Header Part No.	On-board Termination Required	Adapter to Analyzer Cable	
2x20 HP 1251-8828	DIP 1810-1278 (9 per part)	none	
	SIP 1810-1588 (5 per part)		

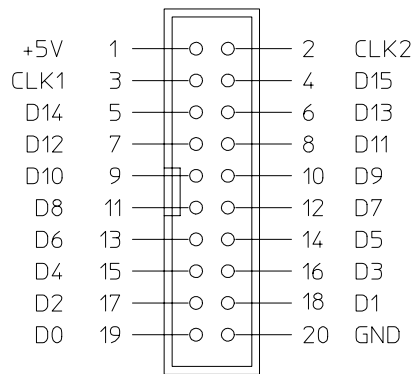
2x10 headers

The 2x10 headers are medium density connectors. Each connector carries 16 microprocessor signals plus one clock (CLK1). This connection scheme requires one HP 100 kOhm Termination Adapter per connector (HP part number 01650-63203). On-board RC termination is not required.

The figure below shows the pinout for a 2x10 header. Refer to the “Hardware Reference” chapter (page 259) for the tables showing the microprocessor signals for each pin. Note that the +5V pin (pin 1) supplies power from the logic analyzer to active devices on an interface board. In most instances, this pin should not be used.

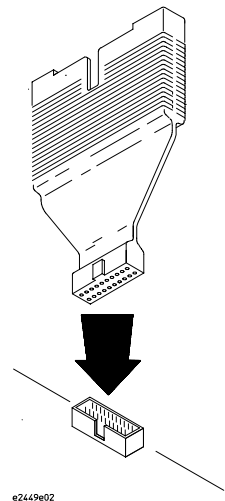
This connection scheme is not recommended for target systems operating above 50 MHz. The table below contains the part number for the component required for this connection scheme.

	Header Part No.	On-board Termination Required	Adapter to Analyzer Cable
2x10	HP 1251-8106	none	01650-63203 Termination Adapter (one per analyzer pod)



2x10 Header

e2449b02



e2449e02

2x10 header pinouts and Termination Adapter

Connecting the Logic Analyzer to the Target System Connectors

The procedures for connecting and configuring the logic analyzer for use with custom probing are the same as for the HP E2455B. They are listed in Chapter 3, “Connecting and Configuring the Analysis Probe,” beginning on page 41.

Analyzing the PPC603 with an
HP 16600A/16700A-series
Logic Analyzer

This chapter describes modes of operation for the HP E2455B analysis probe. It also describes data, symbol encodings, and information about the inverse assembler.

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

- Modes of operation
- Logic analyzer configuration
- Using the inverse assembler

Modes of Operation

The HP E2455B analysis probe can be used in three different analysis modes: State-per-ack, State-per-clock, or Timing. The following sections describe these modes and how to configure the logic analyzer for each mode.

State-per-ack mode

In State-per-ack mode, the logic analyzer uses trigger sequencer 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:

- 1 Click on the logic analyzer icon.
- 2 Select **Setup...** from the menu.
- 3 Select the **Trigger** tab.
- 4 Change the **Store by default** mode to **Anything**.
- 5 Ensure that the trigger sequence is set to find the pattern “XXXXXXXX” one time and then trigger and fill memory.

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:

- 1** Click on the logic analyzer icon.
- 2** Select **Setup...** from the menu.
- 3** Click on the **Sampling** tab.
- 4** Select the **Timing Mode** button.

Modes of Analysis

The inverse assembler offers two modes of analysis for PowerPC 603 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 PowerPC 603 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 inverse assembler requires that an S-Record executable file is loaded. This enables the logic analyzer to display the inverse assembled data in mnemonics.

For cache-on trace reconstruction, set the operating mode to State-per-ack and enable the branch trace mode by setting the MSR.BE bit 22.

Logic Analyzer Configuration

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

It is strongly recommended that you do not change the setup related to the PPC603 sampling, format, pod assignment or configuration dialogs. The configuration file (loaded by the Setup Assistant in HP 16600/700A-series logic analysis systems) will configure the logic analyzer for making measurements of the PPC603.

Format menu

This section describes the organization of PPC603 signals in the logic analyzer's Format menu.

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 <i>PowerPC</i>
ADDR31	ADDR0 <i>Logic Analyzer</i>

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.

The configuration software sets up the analyzer format dialog to display either eight or ten pods of data, depending on the analyzer.

Status Encoding

Each of the bits of the STAT label is described in the table below. Most of the status and control signals on the PowerPC 603 are active low (“-” suffix). To conserve display space, the “-” is omitted in many of the Format definitions.

The inverse assembler uses STAT bits TCO, TSIZE...2, TTO...3, TBST, TA, AACK, ARTRY, DRTRY, ABB, TEA, and TS. The signal-to-connector tables in the “Hardware Reference” chapter list all the PPC603 signals probed and their corresponding analyzer channels.

Status Bit Description

Status Bit	Description
BR-	The PowerPC 603 asserts Bus Request to indicate that it has business to conduct on the address bus
BG-	The memory system asserts Bus Grant to allow the 603 onto the address bus
ABB-	Address Bus Busy indicates that the address bus is in use
TS-	The PowerPC 603 asserts TS- for one cycle to commence a transaction. It also serves as the data bus request signal if the TT signals indicate a data transfer.
XATS-	XATS commences a "programmed i/o" (PIO) sequence in the extended address transfer protocol (not available on 603e).
DBG-	The memory system asserts Data Bus Grant to allow the 603 onto the data bus.
DBWO-	The memory system may assert Data Bus Write Only to allow the 603 to envelope a data write (snoop push, typically) between the address and data phases of a data read.
DBB-	Indicates Data Bus Busy.
AACK-	The memory system asserts AACK for one cycle to acknowledge an address.
ARTRY-	The memory system may assert ARTRY to cause the 603 to back off the bus and retry the transaction.
TA-	The memory system asserts TA to acknowledge a data transaction.
DRTRY-	The memory system may assert DRTRY to cancel the effect of a TA in the previous cycle.

Chapter 5: Analyzing the PPC603 with an HP 16600A/16700A-series Logic Analyzer Logic Analyzer Configuration

Status Bit	Description
TEA-	The memory system may assert TEA to indicate a transfer error, e.g. an unmapped part of the address space.
TT 0:3	The Transfer Type signals indicate the direction and purpose of a bus transaction.
Atomic(TT0)	Usually, TT0 asserted indicates an atomic (e.g., stwctx.) transfer.
R/-W (TT1)	TT1 is high for a read, low for a write.
InvlDt (TT2)	Usually, the PowerPC 603 asserts TT2 to indicate that the corresponding cache line should be invalidated by other processors.
A Only (TT3)	TT3 high indicates that there is data associated with the current address. TT3 low usually indicates an address-only transaction.
TC 0:1	The Transfer Code outputs provide further information about the current transfer. For a read, they indicate whether instructions or operands are being fetched.
TBST-	When asserted, TBST- indicates a four-beat burst transfer of eight words.
TSIZ 0:2	Indicates the size for the data transfer in conjunction with TBST-.
WT-	Write Through indicates a write-through transaction that should be pushed through local caches to shared memory.
CI-	Cache Inhibit indicates that the 603 will not cache a read.
GBL-	Global indicates the transaction is global, i.e., should be snooped by all other caching devices on the bus.
SRESET-	A falling-edge input causes the 603 to undergo a soft reset.
HRESET-	Input asserted causes the 603 to undergo a hard reset.
CKSTP-	Input asserted causes the 603 to undergo machine check processing.
INT-	Input indicates an external interrupt is pending.
CHECKSTOP	Output indicates that the 603 has entered the machine check state, i.e., stopped.
QREQ-	The PowerPC 603 asserts Quiescent Request to indicate that it wants to be, or is, in a snooze mode.

If you are probing a PowerPC 603e, the signal labeled XATS- is actually CSE1 (and CSE, on pod J5odd, is actually CSE0). You may wish to change the name of this label. In the format menu, select the XATS label, and use the Modify Label feature to change the name to CSE1. (DO NOT modify the bit assignments to the STAT label.)

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
acks	idle	1111
	ARTRY	xxx0
	DRTRY	0xxx
	TA AACK	x00x
	AACK	xx0x
	TA	x0xx
R/-W	rd	1
	wr	0
TSIZ	burst	xxx0
	8 byte	0001
	1 byte	0011
	2 byte	0101
	3 byte	0111
	4 byte	1001
	5?byte	1011
	6?byte	1101
7?byte	1111	
TT	Kill Block	0110
	Wr Graphics	1010
	Rd Graphics	1110
	Clean Block	0000
	Write	0001
	Wr/Kill	0011
	Read	0101
	Rd/Flush	0111
	Wr/Atomic Flush	1001
	Read Atomic	1101
	Rd/Flush Atomic	1111
	?Flush Block	0010
	?DSYNC	0100
	?eieio	1000
?(reserved)	1011	
?TLB Invalidate	1100	
STAT	inst fetch xxxx xxxx xxxx xxx1 xxxx 1xxx xxxx 0xxx	

Chapter 5: Analyzing the PPC603 with an HP 16600A/16700A-series Logic Analyzer Logic Analyzer Configuration

The least significant bit of the TSIZ label is the TBST- signal. The symbols prefixed by “?” represent signal outputs defined by the PowerPC architecture but not asserted by the PowerPC 603. An instruction fetch is indicated by AACK asserted (address & qualifiers valid), R/-W (TT1) asserted for read, and TC0 asserted.

Trigger dialog

This section describes some PowerPC 603-specific considerations in triggering the analyzer. You can use the Trigger dialog to change the triggering and storage qualification to include or exclude specified cycles. The trigger specification set up by the software stores all states.

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

Qualifying Stored Data

The trigger dialog determines what will be acquired by the analyzer and when it will be acquired. The HP E2455B software pre-configures a storage qualification term to exclude wait and idle states from the analyzer's memory.

The configuration creates a custom default trigger term called "idle" and changes the default storing to "store if not idle". The custom term "idle" is defined as AACK, ARTRY, TA, and DTRTY all high (not asserted). The trigger sequence stores states that are not idle.

Configuring for State-per-clock mode

To configure the analyzer to store all states including wait and idle states, change the storage qualification to capture all states (state-per-clock).

Go to the trigger dialog and select the **Default Storing** tab. Change the default storing mode from **Custom** to **Anything**.

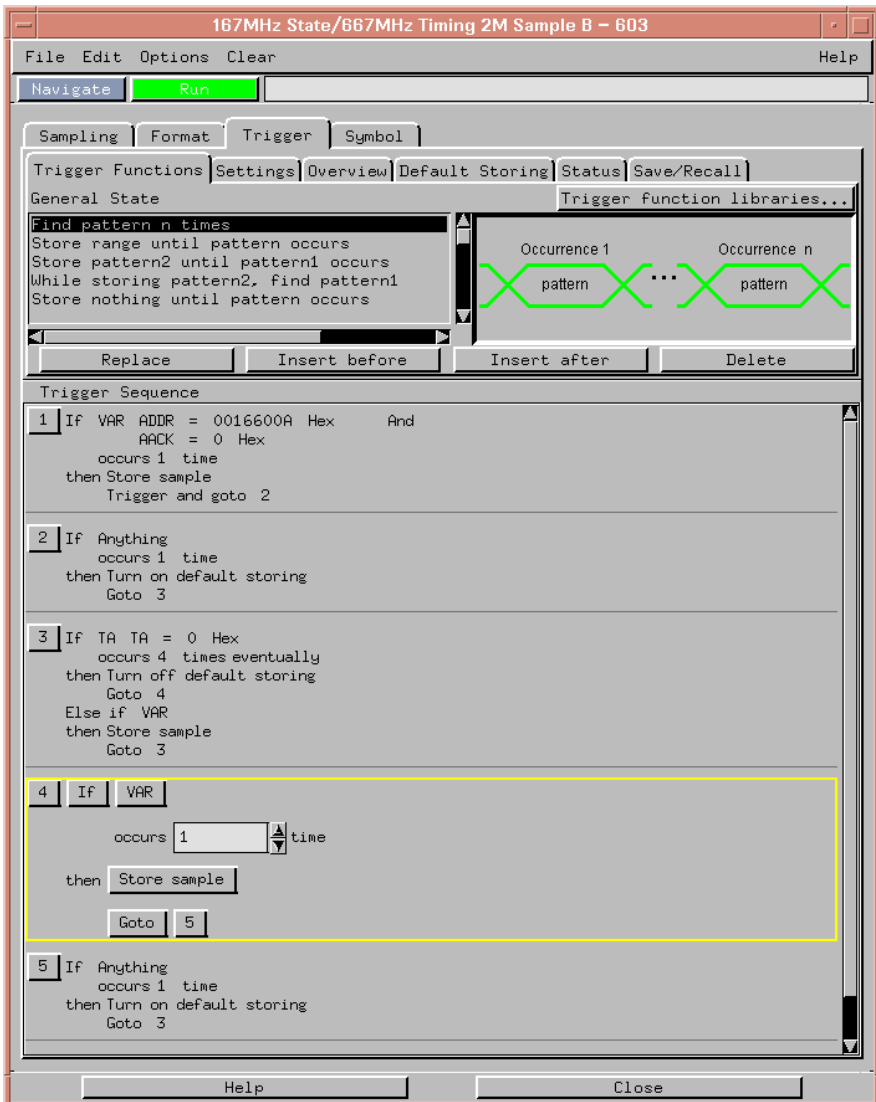
Capturing an address

To accurately trigger on a specific address, create a term with two labels: ADDR and AACK. Enter the address in the ADDR field of a trigger term and enter 0 in the AACK field of the term. This will prevent false triggering on a floating address bus.

The instruction addresses presented on the PowerPC 603 address bus always end in hex 0 or hex 8. When the instruction cache is enabled, the 603 will burst four data beats per address and will not update the address as it bursts. To reliably trigger on the fetch of a particular address when bursting, the least significant three bits of the address must be "don't cares." Change the base of the ADDR label to Binary to enter the 3 X's.

Capturing Isolated Addresses

The loose coupling of the address and data buses on the PowerPC 603 makes it more difficult to trace only activities associated with a given address, such as writes to a variable. Depending on how deep the pipeline is, an address of interest may be followed by up to four data beats before the data associated with the address appears on the bus. One technique to trace writes to a variable is shown below. (The data cache is off or in write-through mode.)



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 branch trace mode. In order to trace in the cache the user must set the MSR.BE bit 22. This BE bit enables a branch trace exception to be taken after a successful completion of a branch instruction. This feature also requires that the data bus is connected and an S-Record executable file is loaded.

The branch exception is located at 0x00000D00 for an exception prefix MSR.IP=0 or 0xFFFF00D00 for an exception prefix MSR.IP=1. The interrupt routine writes the branch target address SRR0 to the tracking address (location in RAM which is non-cached or write-through mode is enabled for that memory block) so that the IA can track the program flow. Also, the tracking address must be on a word boundary.

example branch exception routine:

```
0x00000d00: mfspr    r7, d26
0x00000d04: addis   r8, r0, 0x0000
0x00000d08: stw    r7, 0x0100(r8)
0x00000d0c: rfi
```

This branch exception writes the branch target address to a tracking address of 0x00000100.

Customers who wish to nest interrupts must save and restore the SRR0 special purpose register before writing it out to the tracking address. Also, the customer must write out the exception address at the beginning of the exception.

Chapter 5: Analyzing the PPC603 with an HP 16600A/16700A-series Logic Analyzer Using the Inverse Assembler

example program exception routine:

```
0x00000700: addis r6, r0, 0x0000
0x00000704: addi  r6, 0x0700
0x00000708: addis r8, r0, 0x0000
0x0000070C: stw   r6, 0x0100(r8)
0x00000710: .
0x00000714: .
0x00000718: .
0x0000071C: mfspr r7, d26
0x00000720: stw   r7, 0x0100(r8)
0x00000724: rfi
```

To enable cache-on trace reconstruction:

In the **External Bus Decoding** dialog, located in the **Decoding Options** tab:

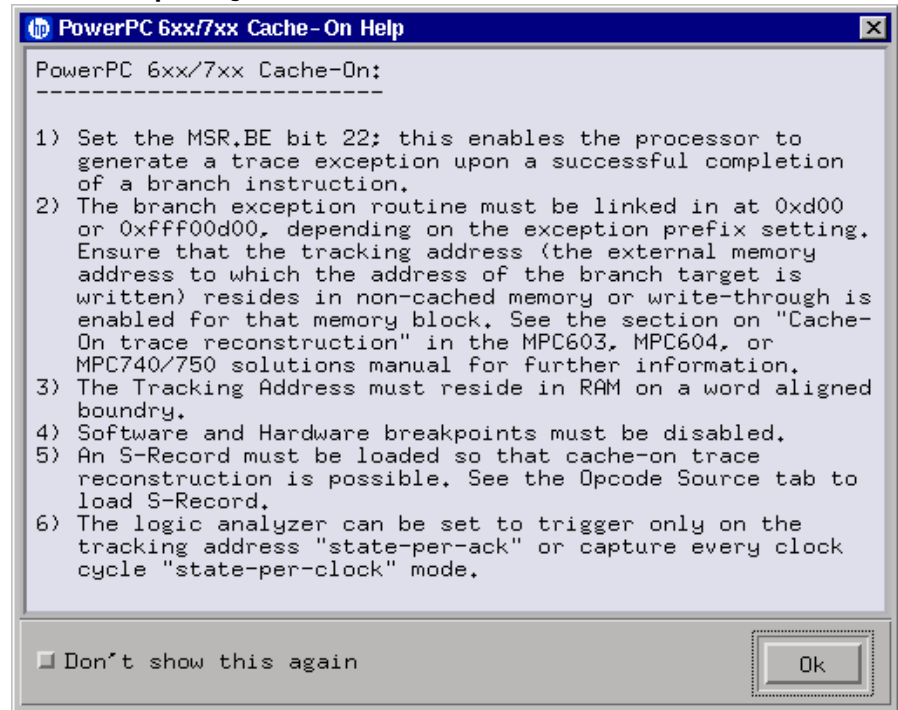
- 1** Set the cache-on mode
- 2** Set data bus connected
- 3** Provide the tracking address

In the **Opcode Source** tab

- 4** Load an S-Record executable file

When cache-on mode is enabled the following dialog will appear.

Cache-on help dialog



NOTE:

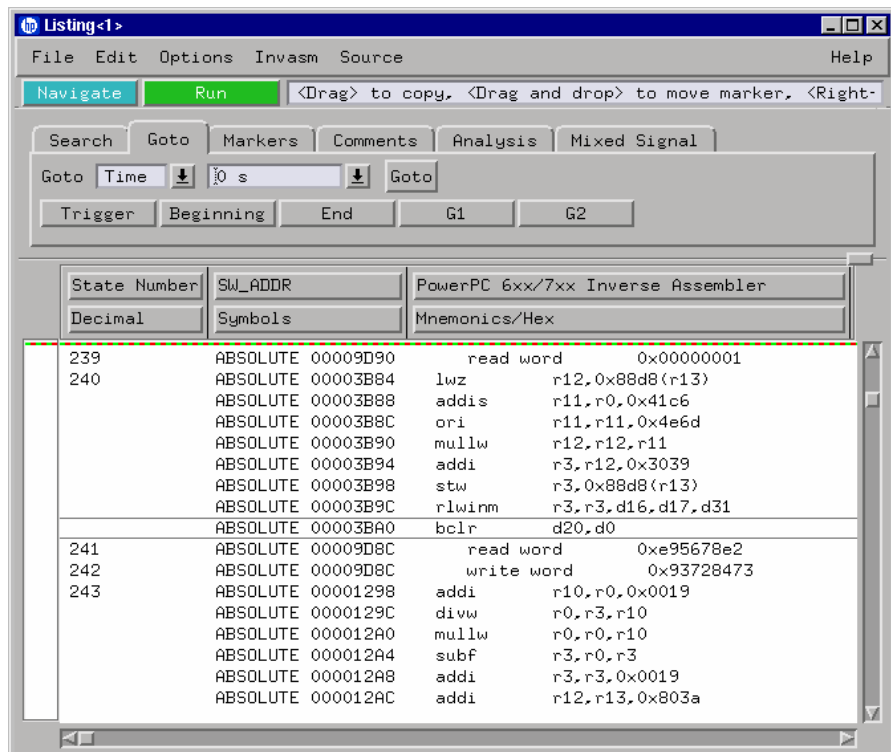
PPC6xx7xx, When using HP run-control the user must issue an "rstssm" (reset soft stop mode) to disable soft stop mode. Also, this command must be issued whenever registers are modified, since soft stop mode is re-enabled automatically.

Enabling branch exception disassembly

The following trace shows cache-on execution using branch trace exception disassembly. See page 109 for an explanation of this feature.

To enable branch trace exception, set the MSR.BE bit 22.

Cache-on trace, S-Record executable file loaded, data bus connected, tracking address 0x00000100



The screenshot shows the 'Listing<1>' window of the PowerPC 6xx/7xx Inverse Assembler. The window title is 'Listing<1>' and it has a menu bar with 'File', 'Edit', 'Options', 'Invasm', 'Source', and 'Help'. Below the menu bar are buttons for 'Navigate', 'Run', and a text field for '<Drag> to copy, <Drag and drop> to move marker, <Right-'. There are also buttons for 'Search', 'Goto', 'Markers', 'Comments', 'Analysis', and 'Mixed Signal'. A 'Goto' section contains a 'Time' dropdown, a '↓' button, a text field with '0 s', another '↓' button, and a 'Goto' button. Below this are 'Trigger' buttons for 'Beginning', 'End', 'G1', and 'G2'. The main display area is a table with columns for 'State Number', 'SW_ADDR', and 'PowerPC 6xx/7xx Inverse Assembler'. The table has sub-columns for 'Decimal', 'Symbols', and 'Mnemonics/Hex'. The data is as follows:

State Number	SW_ADDR	PowerPC 6xx/7xx Inverse Assembler
Decimal	Symbols	Mnemonics/Hex
239	ABSOLUTE 00009D90	read word 0x00000001
240	ABSOLUTE 00003B84	lwz r12,0x88d8(r13)
	ABSOLUTE 00003B88	addis r11,r0,0x41c6
	ABSOLUTE 00003B8C	ori r11,r11,0x4e6d
	ABSOLUTE 00003B90	mullw r12,r12,r11
	ABSOLUTE 00003B94	addi r3,r12,0x3039
	ABSOLUTE 00003B98	stw r3,0x88d8(r13)
	ABSOLUTE 00003B9C	rlwinm r3,r3,d16,d17,d31
	ABSOLUTE 00003BA0	bclr d20,d0
241	ABSOLUTE 00009D8C	read word 0xe95678e2
242	ABSOLUTE 00009D8C	write word 0x93728473
243	ABSOLUTE 00001298	addi r10,r0,0x0019
	ABSOLUTE 0000129C	divw r0,r3,r10
	ABSOLUTE 000012A0	mullw r0,r0,r10
	ABSOLUTE 000012A4	subf r3,r0,r3
	ABSOLUTE 000012A8	addi r3,r3,0x0019
	ABSOLUTE 000012AC	addi r12,r13,0x803a

Inverse Assembler Modes of Operation

The following table 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 executable file is loaded.
off	no	yes	Opcodes are fetched from the S-Record executable file 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 executable file and decoded into instruction mnemonics. R/W data will be displayed.
on	no	no	Error message: cache-on decoding requires that the data bus is connected and that an S-Record executable file is loaded.
on	no	yes	Error message: cache-on decoding requires that the data bus is connected and that an S-Record executable file is loaded.
on	yes	no	Error message: cache-on decoding requires that the data bus is connected and that an S-Record executable file is loaded.
on	yes	yes	Cache-on Trace Reconstruction: Tracking address data provides the address so opcodes can be fetched from the S-Record executable file 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.

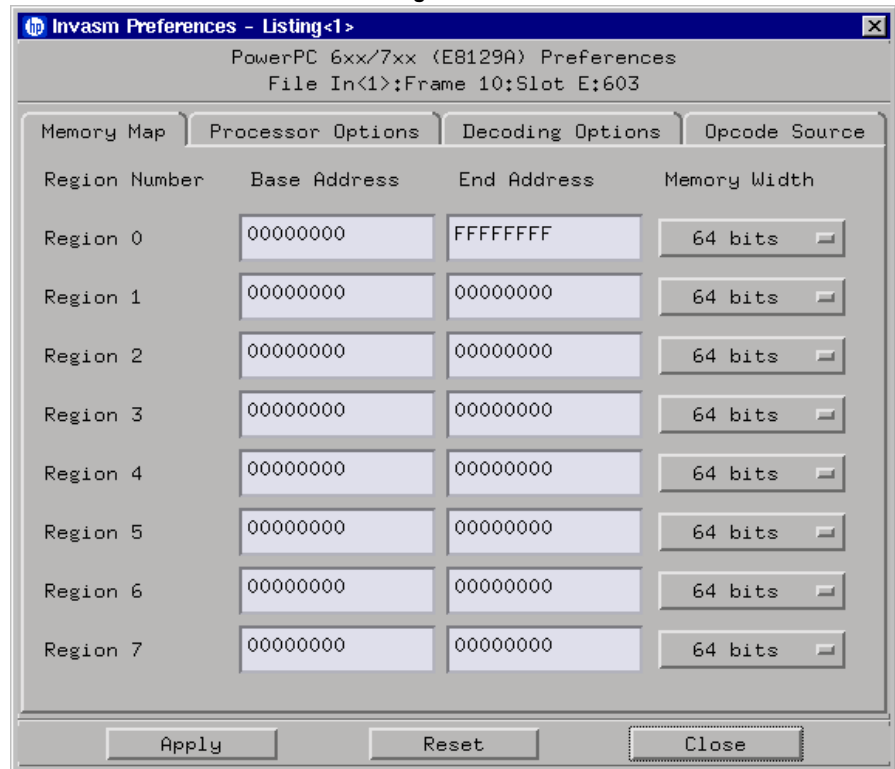
The memory controller operates by mapping every address to one of eight memory regions. Each memory region can be set up to drive different external signals, to have different write permissions, etc. The memory regions are numbered from 0 to 7. Memory region 0 has the highest priority and region 7 has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

The base register and option register for each memory region hold information that describes the width of the memory accessed through that region and the addresses that will be accessed through that region. 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.

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



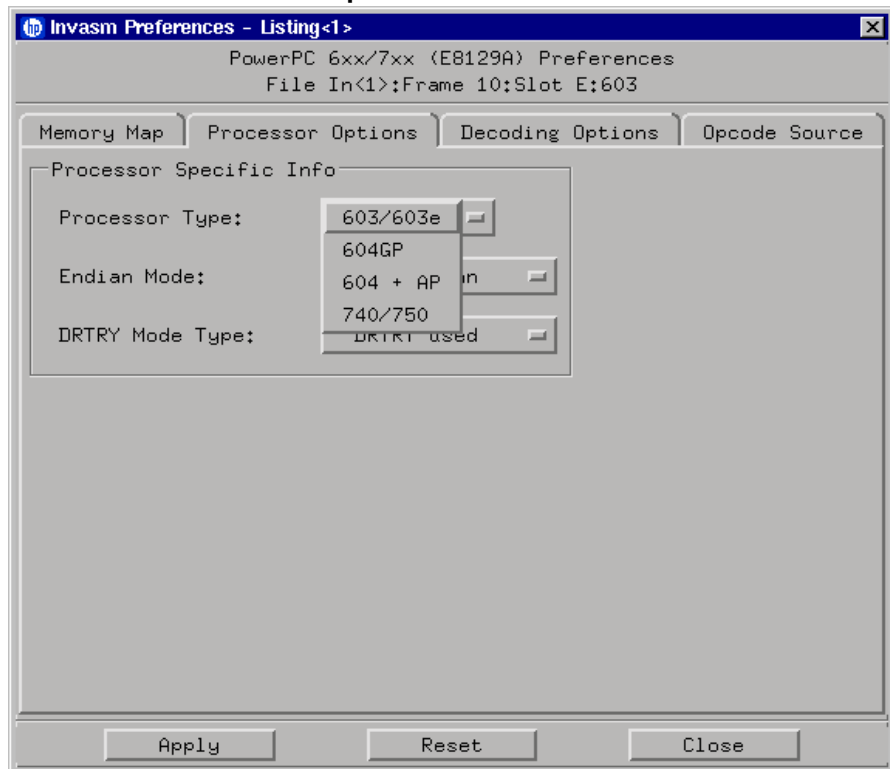
Click **Apply** when you have finished configuring the memory map.

To set the processor options preferences

Processor Options. This inverse assembler is designed to work with the PPC603, PPC604, and PPC740/750 PowerPC microprocessors. Because of this wide range of support, you must specify which processor type is currently being used. Use the following table to determine how to specify the processor type.

Processor Used	Set Processor Type
All PowerPC 603 processors	603/603e
All PowerPC 604 processors with no Analysis Probe	604GP
All PowerPC 604 processors with an Analysis Probe	604 + AP
PowerPC 740/750	740/750

Inverse Assembler Processor Options



Endian Mode

The inverse assembler is designed to support both the native big endian mode and the little endian mode of operation. When operating in little-endian mode, the processor uses a technique known as “address munging” to convert internal little endian addresses into external big endian addresses. Internal and external addresses may differ from one another in the three least significant bits.

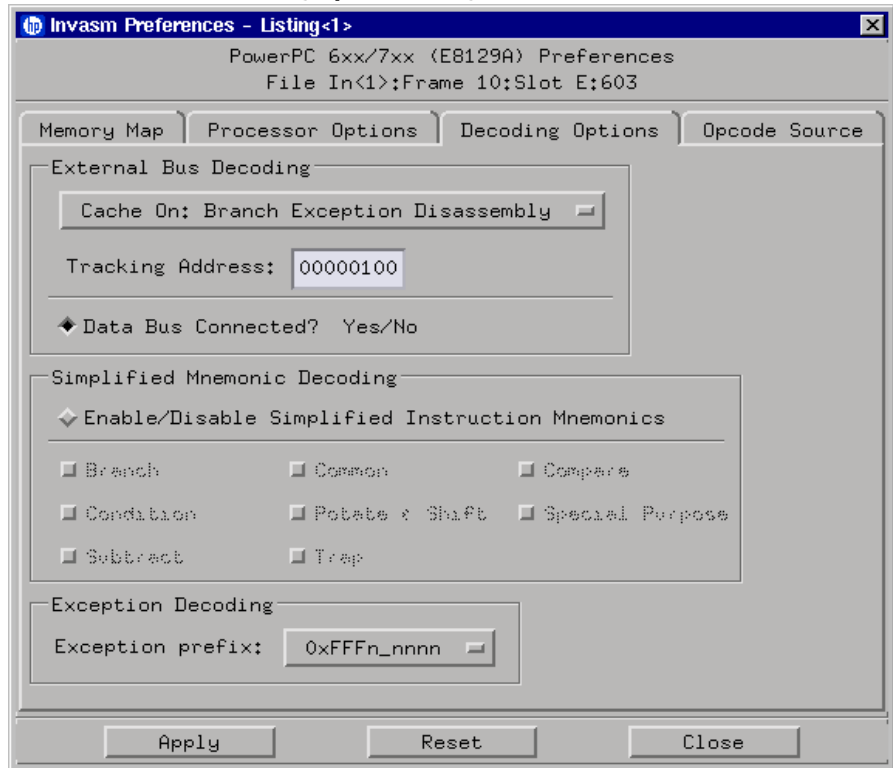
Little endian mode causes the instruction word from DL0...31 (DATA_B label; external address xxx4) to be dispatched before the instruction word from DH0...31 (DATA label; external address xxx0). It also causes byte and half-word reads and writes to appear on the opposite side of the bus and swaps the halves of double-word reads and writes. Setting the endian mode to **Little Endian** automatically compensates for these little endian operations.

DRTRY Mode

The inverse assembler also allows you to specify inclusion of the DRTRY signal in its decoding. There are certain versions of the PowerPC microprocessors that have a no-DRTRY mode. If your processor is currently running in this mode, be sure to select the no-DRTRY mode.

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: Branch Exception Disassembly** for cache-on trace reconstruction, and provide the tracking address.

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 113

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.

- 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 mtrcf instruction field mask specifies the entire cr, it is decoded as mtrc.

Chapter 5: Analyzing the PPC603 with an HP 16600A/16700A-series Logic Analyzer Using the Inverse Assembler

The Extended dialect adds several extended opcodes for the rotate instructions. For example, the function of the `rlwinm` instruction

```
rlwinm r30 r30 16. 16. 31.
```

is to shift right word immediate, e.g.

```
srwi r30 r30 16.
```

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

Mnemonic	Decoded As
<code>rlwimi</code> (rotate left word immediate then mask insert)	<code>inslwi</code> insert from left immediate <code>insrwi</code> insert from right immediate
<code>rlwinm</code> (rotate left word immediate then AND with mask)	<code>rotlwi</code> rotate left immediate <code>rotrwi</code> rotate right immediate <code>slwi</code> shift left immediate <code>srwi</code> shift right immediate <code>extlwi</code> extract and left justify immediate <code>extrwi</code> extract and right justify immediate <code>clrlwi</code> clear left immediate <code>clrrwi</code> clear right immediate <code>clrlslwi</code> clear left and shift left immediate
<code>rlwnm</code> (rotate left word then AND with mask)	<code>rotlw</code> rotate left

The inverse assembler supports the following extensions of dialect-sensitive instructions.

Instruction Types	raw	extended
branches	bc %00100,2,FFF00230	bne cr0,FFF00230
trap	tw %10000,r5,r6	tw lt,r5,r6
compare	cmp cr1,0,r0,r16	cmpw cr1,r0,r16
	ori r0,r0,0000	nop
subtract	addi r6,r6,FCFC	subi r6,r6,0304
	subf r7,r19,r16	sub r7,r16,r19
common	addi r3,0,7000	li r3,7000
	addis r3,0,7000	lis r3,7000
	or r4,r5,r5	mr r4,r5
	nor r4,r5,r5	not r4,r5
	xor r7,r7,r7	clr r7
	eqv r8,r8,r8	set r8
special purpose	mtrcf %1111111,r5	mtrcr r5
condition	creqv 7,7,7	crset 7
	crxor 8,8,8	crclr 8
	cror 7,8,8	crmv 7,8
	crnor 8,9,9	crnot 8,9
rotates and shifts	rlwnm r8,r7,r6,0,31.	rotlw r8,r7,r6
	rlwimi r3,r3,24.,8,23.	inslwi r3,r3,16.,8
	rlwimi r8,r3,17,8,23.	insrwi r8,r3,7,8
	rlwinm r6,r4,8,0,14	extlwi r6,r4,15,8
	rlwinm r6,r4,16,24,31	extrwi r6,r4,8,8
	rlwinm. r6,r4,4,0,31	rotlwi. r6,r4,4
	rlwinm r6,r4,28,0,31	rotrwi r6,r4,4
	rlwinm r6,r4,1,0,30	slwi r6,r4,1
	rlwinm r6,r4,31,1,31	srwi r6,r4,1
	rlwinm r6,r4,0,1,31	clrlwi r6,r4,1
	rlwinm r6,r4,0,0,7	clrrwi r6,r4,14
	rlwinm r6,r4,6,6,25	clrlslwi r6,r4,12,6

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 MSR.IP bit 25. This can be done by examining the initialization code or by using an emulator to view the MSR register.

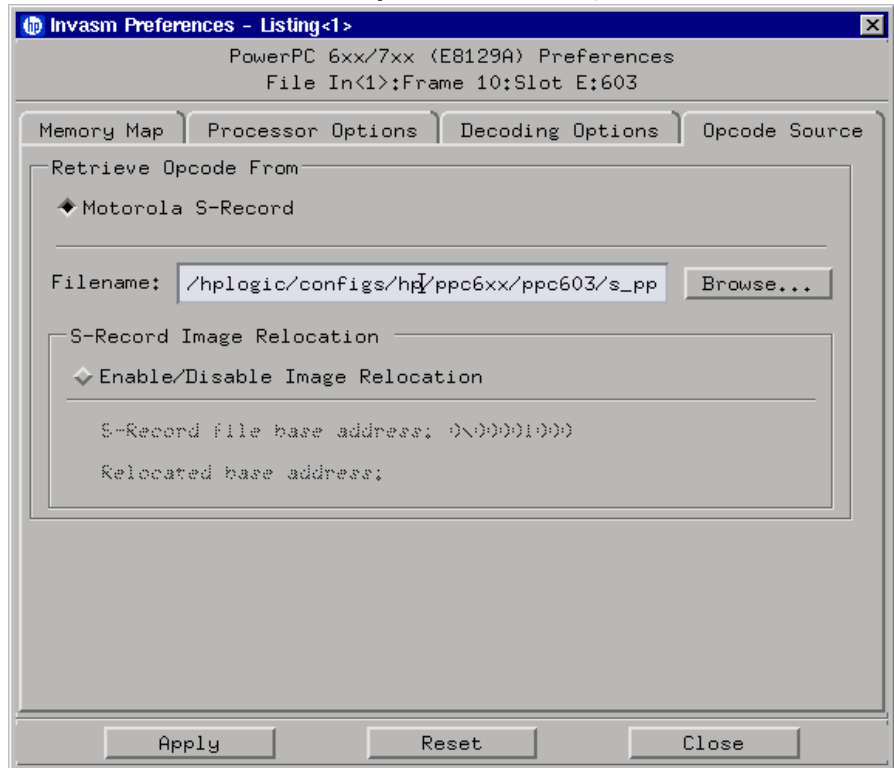
Listing Window Showing Trace with Data Bus Connected: Cache Off

State Number	SW_ADDR	PowerPC 6xx/7xx Inverse Assembler
Decimal	Symbols	Mnemonics/Hex
280	ABSOLUTE 00009D90	read word 0x00000001
281	ABSOLUTE 00003B78	? addi r3,r0,0x0001
282	ABSOLUTE 00003B7C	? stw r3,0x88dc(r13)
	ABSOLUTE 00003B80	? stw r3,0x88d8(r13)
	ABSOLUTE 00003B84	lwz r12,0x88d8(r13)
283	ABSOLUTE 00003B88	addis r11,r0,0x41c6
	ABSOLUTE 00003B8C	ori r11,r11,0x4e6d
284	ABSOLUTE 00009D8C	read word 0x237c228a
285	ABSOLUTE 00003B90	mullw r12,r12,r11
	ABSOLUTE 00003B94	addi r3,r12,0x3039
286	ABSOLUTE 00003B98	stw r3,0x88d8(r13)
	ABSOLUTE 00003B9C	rlwinm r3,r3,d16,d17,d31
287	ABSOLUTE 00003BA0	bclr d20,d0
	ABSOLUTE 00003BA4	* stw r3,0x88d8(r13)
288	ABSOLUTE 00009D8C	write word 0xaf1cf0fb
289	ABSOLUTE 00001298	addi r10,r0,0x0019
	ABSOLUTE 0000129C	divw r0,r3,r10

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 executable file. 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.

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 region 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 enable/disable the instruction cache on the PPC603

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 configure the HID0 register.

Register values for controlling the cache

Value	Meaning
0000 8000	Enable Instruction Cache
0000 4000	Enable Data Cache
0000 0800	Invalidate Instruction Cache
0000 0400	Invalidate Data Cache

To disable the cache with code:

- Disable the instruction cache with the following code:

```
mf spr    r3, hid0
rlwinm   r3, r3, 0, 17, 15 # clear bit 16 (ICE)
mt spr   hid0, r3
isync
```

- To also disable the data cache use:

```
mf spr    r3, hid0
rlwinm   r3, r3, 0, 18, 15 # clear ICE and DCE
mt spr   hid0, r3
isync
```

- To invalidate and disable both caches use:

```
mfspr    r3, hid0
ori      r3, 0C00# set ICFI and DCFI
mtspr    hid0, r3
rlwinm   r3, r3, 0, 22, 19 # clear ICFI and DCFI
mtspr    hid0, r3
rlwinm   r3, r3, 0, 18, 15 # clear ICE and DCE
mtspr    hid0, r3
isync
```

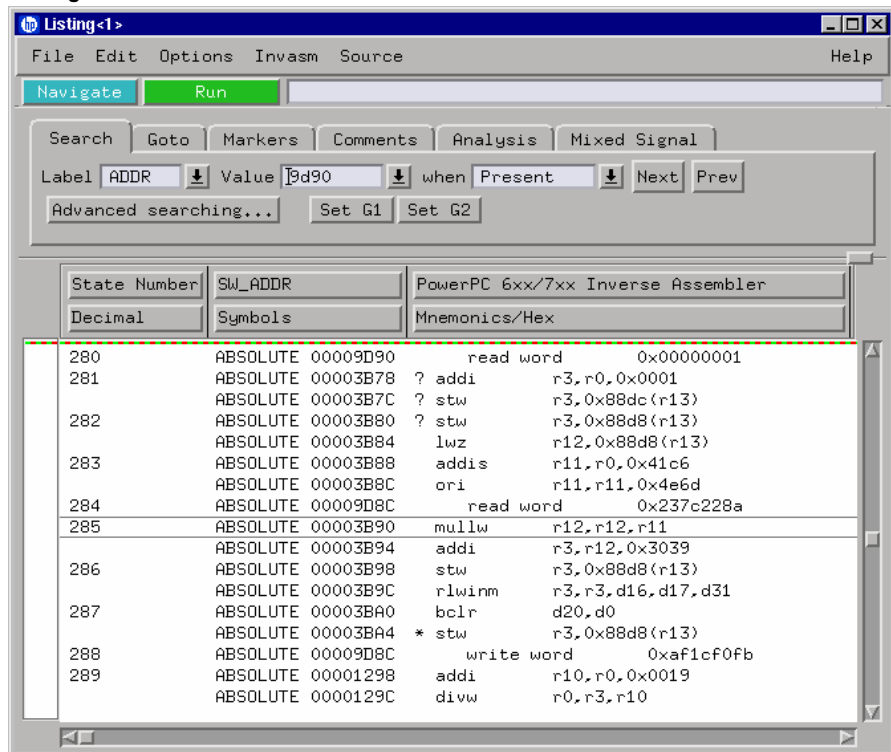
- Enable the instruction cache with the following code:

```
mfspr    r3, hid0
rlwinm   r3, r3, 1, 17, 15 # set ICE
mtspr    hid0, r3
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.



On the HP 16600A/16700A-series logic analysis systems, the entire synthesized address appears under the label "SW_ADDR". The actual address bits presented by the PowerPC 603 may be observed 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 PowerPC 603. 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.

Overfetch Marking

Overfetch refers to instructions which are fetched but not executed by the processor. They may arise from the following sources:

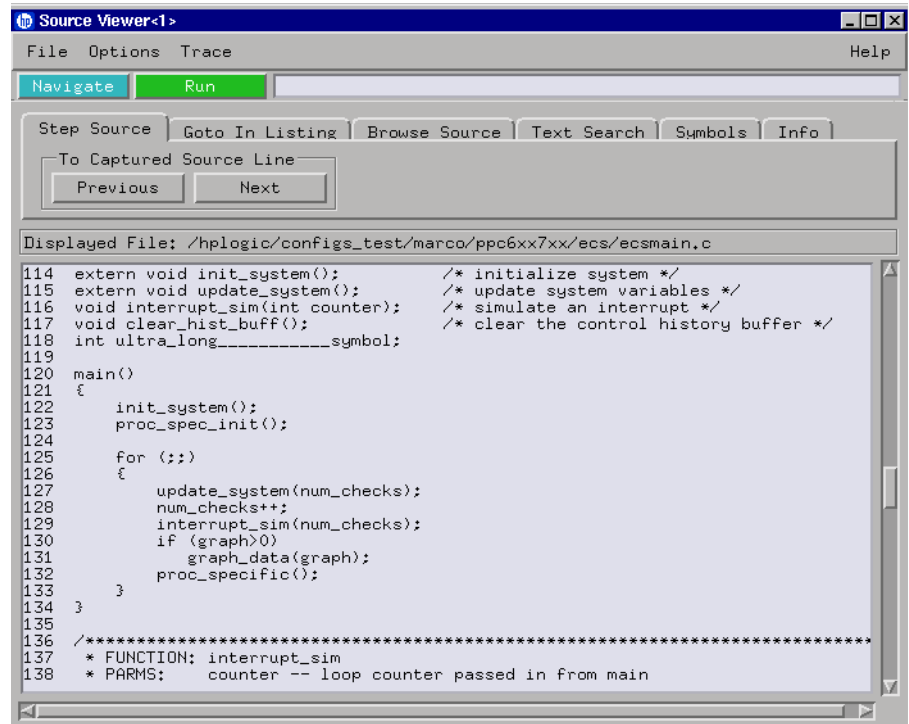
- When the 603 executes a branch instruction, the instructions between the branch and the branch target are not executed. These instructions are indicated with an asterisk “*”, or if the bus trace is ambiguous, with an interrogation point “?”. If the instruction cache is enabled, the branch target may already be in the cache and will not be fetched over the bus. The remaining cache line containing the branch will be marked as overfetch.

For conditional branches whose target addresses are not known, or are known but not seen in the bus traffic, the inverse assembler cannot always determine if the branch was taken and will not mark ensuing states as overfetch.

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



The screenshot shows a window titled "Source Viewer<1>". The menu bar includes "File", "Options", "Trace", and "Help". Below the menu bar are two buttons: "Navigate" (highlighted in green) and "Run" (highlighted in green). A toolbar contains buttons for "Step Source", "Goto In Listing", "Browse Source", "Text Search", "Symbols", and "Info". Below the toolbar is a section for "To Captured Source Line" with "Previous" and "Next" buttons. The main area displays the source code for the file "/hplogic/configs_test/marco/ppc6xx7xx/ecs/ecsmain.c". The code includes function declarations for init_system, update_system, interrupt_sim, and clear_hist_buff, followed by the main function which contains a loop of system updates and interrupt simulations. A comment at the bottom describes the interrupt_sim function.

```
114 extern void init_system();          /* initialize system */
115 extern void update_system();        /* update system variables */
116 void interrupt_sim(int counter);    /* simulate an interrupt */
117 void clear_hist_buff();             /* clear the control history buffer */
118 int ultra_long_____symbol;
119
120 main()
121 {
122     init_system();
123     proc_spec_init();
124
125     for (;;)
126     {
127         update_system(num_checks);
128         num_checks++;
129         interrupt_sim(num_checks);
130         if (graph>0)
131             graph_data(graph);
132         proc_specific();
133     }
134 }
135
136 /*****
137  * FUNCTION: interrupt_sim
138  * PARMS:   counter -- loop counter passed in from main
```

Analyzing the PPC603 with an
HP 1660A/1670A/16500B/C-series
Logic Analyzer

Chapter 6: Analyzing the PPC603 with an HP 1660A/1670A/16500B/C-series Logic Analyzer

This chapter describes modes of operation for the HP E2455B analysis probe. It also describes data, symbol encodings, and information about the inverse assembler.

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

- Modes of operation
- Logic analyzer configuration
- Using the inverse assembler

Modes of Operation

The HP E2455B analysis probe can be used in three different analysis modes: State-per-ack, State-per-clock, or Timing. The following sections describe these modes and how to configure the logic analyzer for each mode.

State-per-ack mode

In State-per-ack mode, the logic analyzer uses trigger sequencer 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:

- Select the Trigger dialog and change the store qualification to “anystate”.

Modes of Operation

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:

- 1** Select the Configuration menu of the logic analyzer.
- 2** Select the Type field for Analyzer 1.
- 3** Select Timing.

Logic Analyzer Configuration

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

It is strongly recommended that you do not change the setup related to the PPC603 sampling, format, pod assignment or configuration dialogs. The configuration file (loaded by the Setup Assistant in HP 16600/700A-series logic analysis systems) will configure the logic analyzer for making measurements of the PPC603.

Format menu

This section describes the organization of PPC603 signals in the logic analyzer's Format menu.

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 <i>PowerPC</i>
ADDR31	ADDR0 <i>Logic Analyzer</i>

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.

Logic Analyzer Configuration

The configuration software sets up the analyzer format dialog to display either eight or ten pods of data, depending on the analyzer.

Status Encoding

Each of the bits of the STAT label is described in the table below. Most of the status and control signals on the PowerPC 603 are active low (“-” suffix). To conserve display space, the “-” is omitted in many of the Format definitions.

The inverse assembler uses STAT bits TC0, TSIZE...2, TTO...3, TBST, TA, AACK, ARTRY, DRTRY, ABB, TEA, and TS. The signal-to-connector tables in the “Hardware Reference” chapter list all the PPC603 signals probed and their corresponding analyzer channels.

Status Bit Description

Status Bit	Description
BR-	The PowerPC 603 asserts Bus Request to indicate that it has business to conduct on the address bus
BG-	The memory system asserts Bus Grant to allow the 603 onto the address bus
ABB-	Address Bus Busy indicates that the address bus is in use
TS-	The PowerPC 603 asserts TS- for one cycle to commence a transaction. It also serves as the data bus request signal if the TT signals indicate a data transfer.
XATS-	XATS commences a "programmed i/o" (PIO) sequence in the extended address transfer protocol (not available on 603e).
DBG-	The memory system asserts Data Bus Grant to allow the 603 onto the data bus.
DBWO-	The memory system may assert Data Bus Write Only to allow the 603 to envelope a data write (snoop push, typically) between the address and data phases of a data read.
DBB-	Indicates Data Bus Busy.
AACK-	The memory system asserts AACK for one cycle to acknowledge an address.
ARTRY-	The memory system may assert ARTRY to cause the 603 to back off the bus and retry the transaction.
TA-	The memory system asserts TA to acknowledge a data transaction.
DRTRY-	The memory system may assert DRTRY to cancel the effect of a TA in the previous cycle.

Status Bit	Description
TEA-	The memory system may assert TEA to indicate a transfer error, e.g. an unmapped part of the address space.
TT 0:3	The Transfer Type signals indicate the direction and purpose of a bus transaction.
Atomic(TT0)	Usually, TT0 asserted indicates an atomic (e.g., stwctx.) transfer.
R/-W (TT1)	TT1 is high for a read, low for a write.
InvlDt (TT2)	Usually, the PowerPC 603 asserts TT2 to indicate that the corresponding cache line should be invalidated by other processors.
A Only (TT3)	TT3 high indicates that there is data associated with the current address. TT3 low usually indicates an address-only transaction.
TC 0:1	The Transfer Code outputs provide further information about the current transfer. For a read, they indicate whether instructions or operands are being fetched.
TBST-	When asserted, TBST- indicates a four-beat burst transfer of eight words.
TSIZ 0:2	Indicates the size for the data transfer in conjunction with TBST-.
WT-	Write Through indicates a write-through transaction that should be pushed through local caches to shared memory.
CI-	Cache Inhibit indicates that the 603 will not cache a read.
GBL-	Global indicates the transaction is global, i.e., should be snooped by all other caching devices on the bus.
SRESET-	A falling-edge input causes the 603 to undergo a soft reset.
HRESET-	Input asserted causes the 603 to undergo a hard reset.
CKSTP-	Input asserted causes the 603 to undergo machine check processing.
INT-	Input indicates an external interrupt is pending.
CHECKSTOP	Output indicates that the 603 has entered the machine check state, i.e., stopped.
QREQ-	The PowerPC 603 asserts Quiescent Request to indicate that it wants to be, or is, in a snooze mode.

If you are probing a PowerPC 603e, the signal labeled XATS- is actually CSE1 (and CSE, on pod J5odd, is actually CSE0). You may wish to change the name of this label. In the format menu, select the XATS label, and use the Modify Label feature to change the name to CSE1. (DO NOT modify the bit assignments to the STAT label.)

Logic Analyzer Configuration

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
acks	idle	1111
	ARTRY	xxx0
	DRTRY	0xxx
	TA AACK	x00x
	AACK	xx0x
	TA	x0xx
R/-W	rd	1
	wr	0
TSIZ	burst	xxx0
	8 byte	0001
	1 byte	0011
	2 byte	0101
	3 byte	0111
	4 byte	1001
	5?byte	1011
	6?byte	1101
7?byte	1111	
TT	Kill Block	0110
	Wr Graphics	1010
	Rd Graphics	1110
	Clean Block	0000
	Write	0001
	Wr/Kill	0011
	Read	0101
	Rd/Flush	0111
	Wr/Atomic Flush	1001
	Read Atomic	1101
	Rd/Flush Atomic	1111
	?Flush Block	0010
	?DSYNC	0100
	?eieio	1000
?(reserved)	1011	
?TLB Invalidate	1100	
STAT	inst fetchxxxx xxxx xxxx xxx1 xxxx 1xxx xxxx 0xxx	

The least significant bit of the TSIZ label is the TBST- signal. The symbols prefixed by “?” represent signal outputs defined by the PowerPC architecture but not asserted by the PowerPC 603. An instruction fetch is indicated by AACK asserted (address & qualifiers valid), R/-W (TT1) asserted for read, and TC0 asserted.

Trigger dialog

This section describes some PowerPC 603-specific considerations in triggering the analyzer. You can use the Trigger dialog to change the triggering and storage qualification to include or exclude specified cycles. The trigger specification set up by the software stores all states.

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

Qualifying Stored Data

The trigger dialog determines what will be acquired by the analyzer and when it will be acquired. The HP E2455B software pre-configures a storage qualification term to exclude wait and idle states from the analyzer's memory.

The configuration software renames a pattern term to “idle” and assigns it a pattern with AACK, ARTRY, TA, and DRTRY, all high (not asserted). The sequencer is programmed to store only states \neq idle. That is, only states where one or more of these signals is asserted will be stored.

Configuring for State-per-clock mode

To configure the analyzer to store all states including wait and idle states, change the storage qualification to capture all states (state-per-clock).

Change storage qualification from **≠ idle** to **anystate**.

Capturing an address

To accurately trigger on a specific address, create a term with two labels: ADDR and AACK. Enter the address in the ADDR field of a trigger term and enter 0 in the AACK field of the term. This will prevent false triggering on a floating address bus.

The instruction addresses presented on the PowerPC 603 address bus always end in hex 0 or hex 8. When the instruction cache is enabled, the 603 will burst four data beats per address and will not update the address as it bursts. To reliably trigger on the fetch of a particular address when bursting, the least significant three bits of the address must be “don't cares.” Change the base of the ADDR label to Binary to enter the 3 X's.

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 PPC603 address bus, you must make sure the target system's cache has been disabled.

To disable the instruction cache on the PPC603

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 configure the HID0 register.

Register values for controlling the cache

Value	Meaning
0000 8000	Enable Instruction Cache
0000 4000	Enable Data Cache
0000 0800	Invalidate Instruction Cache
0000 0400	Invalidate Data Cache

To disable the cache with code:

- Disable the instruction cache with the following code:

```
mf spr    r3, hid0
rlwinm   r3, r3, 0, 17, 15  # clear bit 16 (ICE)
mt spr   hid0, r3
isync
```

- To also disable the data cache use:

```
mf spr    r3, hid0
rlwinm   r3, r3, 0, 18, 15  # clear ICE and DCE
mt spr   hid0, r3
isync
```

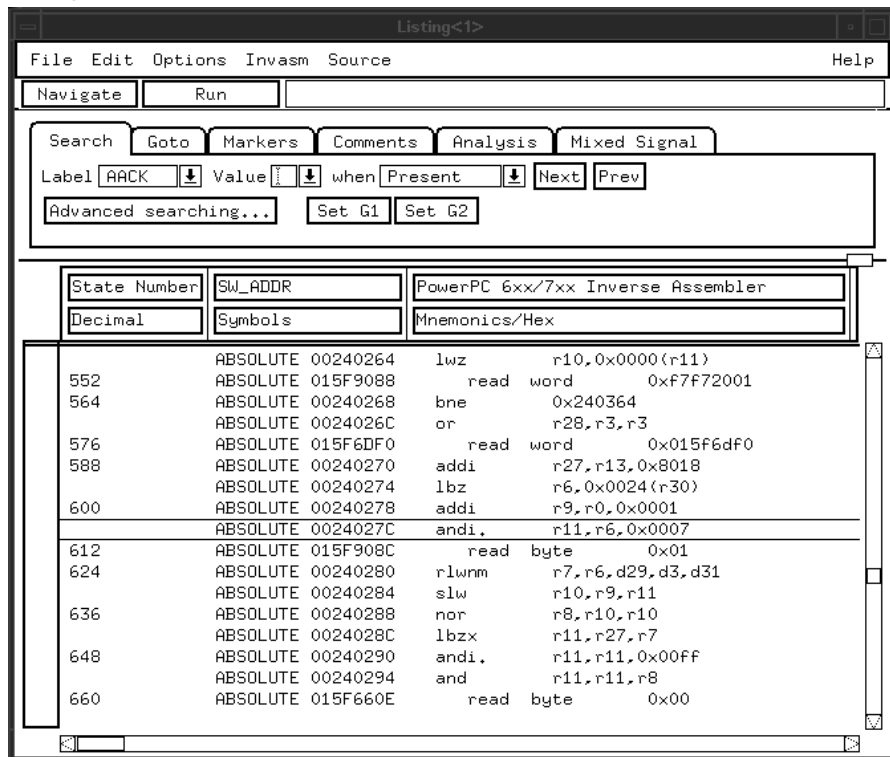
- To invalidate and disable both caches use:

```
mf spr    r3, hid0
ori      r3, 0C00# set ICFI and DCFI
mt spr   hid0, r3
rlwinm   r3, r3, 0, 22, 19  # clear ICFI and DCFI
mt spr   hid0, r3
rlwinm   r3, r3, 0, 18, 15  # clear ICE and DCE
mt spr   hid0, r3
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.



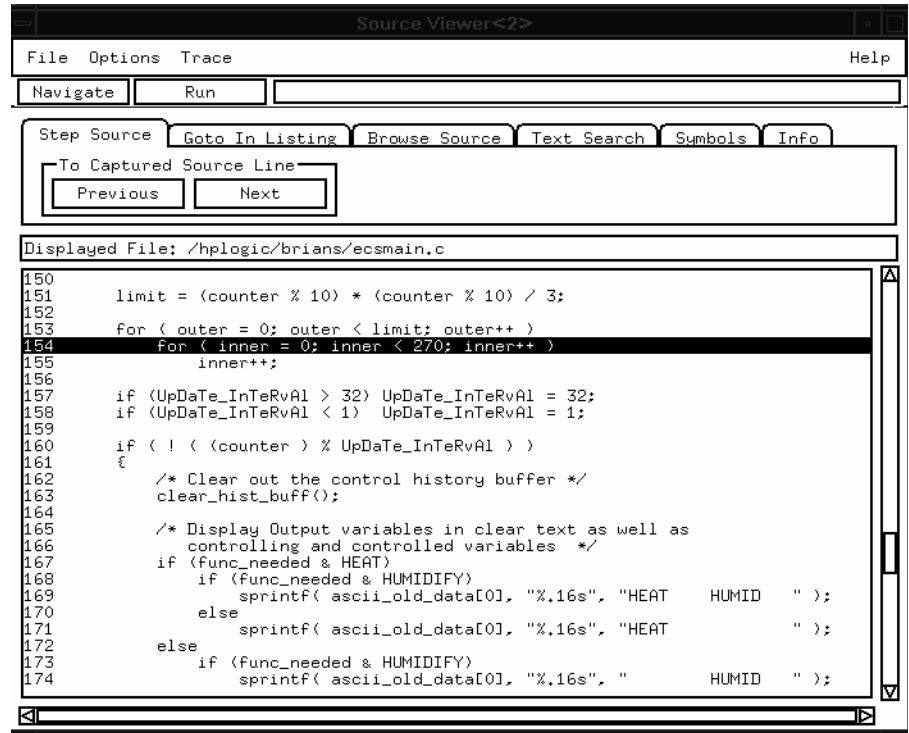
HP 16500C

This paragraph applies only when using the HP 16500C system. The columns on the left of the inverse assembly data display are the least significant hexadecimal digits of an instruction or burst address. These may be useful for matching an execution trace to an assembly listing. Because the PowerPC 603 presents one address and then reads two or eight instructions for each address, the less-significant bits are synthesized by the disassembler.

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.



```
Source Viewer<2>
File Options Trace Help
Navigate Run
Step Source Goto In Listing Browse Source Text Search Symbols Info
To Captured Source Line
Previous Next
Displayed File: /hplogic/brians/ecsmain.c
150
151     limit = (counter % 10) * (counter % 10) / 3;
152
153     for ( outer = 0; outer < limit; outer++ )
154         for ( inner = 0; inner < 270; inner++ )
155             inner++;
156
157     if (UpDaTe_InTeRvAl > 32) UpDaTe_InTeRvAl = 32;
158     if (UpDaTe_InTeRvAl < 1) UpDaTe_InTeRvAl = 1;
159
160     if ( ! ( (counter) % UpDaTe_InTeRvAl ) )
161     {
162         /* Clear out the control history buffer */
163         clear_hist_buff();
164
165         /* Display Output variables in clear text as well as
166            controlling and controlled variables */
167         if (func_needed & HEAT)
168             sprintf( ascii_old_data[0], "%.16s", "HEAT    HUMID    " );
169         else
170             sprintf( ascii_old_data[0], "%.16s", "HEAT          " );
171         else
172             if (func_needed & HUMIDIFY)
173                 sprintf( ascii_old_data[0], "%.16s", "          HUMID    " );
174
```

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. Floating point registers are displayed as f0, f1,..., f31. Condition registers are displayed as cr0, cr1,..., cr7. Special purpose registers are displayed using their mnemonic.

Most numerical data is displayed in hexadecimal, for example, "lwzr28, 0x0044(r1)."

Bit numbers and shift counts are displayed in decimal with a "d" prefix, for example, "cror d31,d31,d31."

A few instructions display their operands in binary with a "b" prefix, for example, "mtfsfi 4,b0101."

The inverse assembler decodes the full 32-bit mode PowerPC instruction set architecture. Instructions that are 64-bit mode or optional instructions not implemented on the PPC603 are decoded as "illegal". Any instruction that does not decode to a valid opcode is shown as "unknown".

When these un-implemented 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.

Overfetch Marking

Overfetch refers to instructions which are fetched but not executed by the processor. They may arise from the following sources:

- When the 603 executes a branch instruction, the instructions between the branch and the branch target are not executed. These instructions are indicated with an asterisk "*", or if the bus trace is ambiguous, with an interrogation point "?". If the instruction cache is enabled, the branch target may already be in the cache and will not be fetched over the bus. The remaining cache line containing the branch will be marked as overfetch.

For conditional branches whose target addresses are not known, or are known but not seen in the bus traffic, the inverse assembler cannot always determine if the branch was taken and will not mark ensuing states as overfetch.

Using the Inverse Assembler

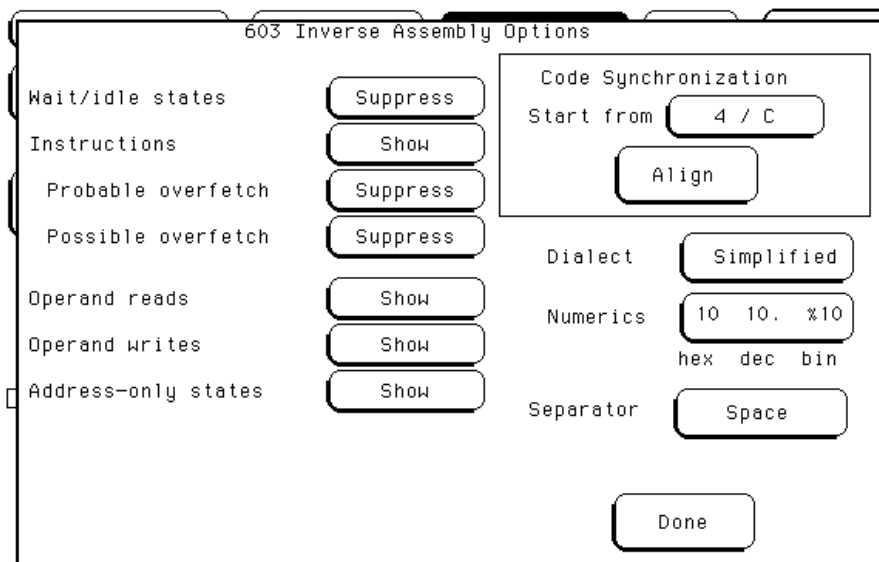
Little-Endian Mode

The inverse assembler is designed to support the native big-endian mode of operation on the PowerPC 603. When operating in little-endian mode, the 603 uses a technique known as “address munging” to convert internal little-endian addresses into external big-endian addresses. Internal and external addresses may differ from one another in the three least significant bits.

In little-endian operation, in a given data beat, the instruction word from DL0...31 (DATA_B label; external address xxx4) will be dispatched before the instruction word from DH0...31 (DATA label; external address xxx0). You can compensate for this by exchanging the DATA and DATA_B labels in the Format menu. However, while this will correctly order 32-bit word reads on the 64-bit data bus, it will cause byte- and half- word reads and writes to appear on the opposite side of the bus, and swap the halves of double-word reads and writes.

To use the Invasm menu (HP 1660, HP 1670, and HP 16500B/C mainframes)

The HP 1660, HP 1670, and HP 16500B/C mainframes provide an Inverse Assembly Options menu, which is accessed by pressing the Invasm button in the Listing window. The Inverse Assembly Options menu contains three functions: display filtering with Show/Suppress selections, Code Synchronization, and Display Options. The figure below shows the Inverse Assembly Options menu.



Display Filtering

Display filtering is similar to the description on page 124.

Code Synchronization

Code Synchronization allows you to correct the display when the inverse assembler incorrectly predicts a conditional branch as taken and incorrectly marks subsequent states as overfetch.

Using the Inverse Assembler

Extended Mnemonics

PowerPC assemblers support a number of extended mnemonics for some popular assembly language instructions as described in the PPC603 User's Manual. The HP E8129A and HP E2455B inverse assemblers support 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 mtrcf instruction field mask specifies the entire cr, it is decoded as mtrc.

The Extended dialect adds several extended opcodes for the rotate instructions. For example, the function of the rlwinm instruction

```
rlwinm r30 r30 16. 16. 31.
```

is to shift right word immediate, e.g.

```
srwi r30 r30 16.
```


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

Chapter 6: Analyzing the PPC603 with an HP 1660A/1670A/16500B/C-series Logic Analyzer

Using the Inverse Assembler

The inverse assembler supports the following extensions of dialect-sensitive instructions.

Instruction Types	raw	extended
branches	bc %00100,2,FFF00230	bne cr0,FFF00230
trap	tw %10000,r5,r6	tw lt,r5,r6
compare	cmp cr1,0,r0,r16	cmpw cr1,r0,r16
	ori r0,r0,0000	nop
subtract	addi r6,r6,FCFC	subi r6,r6,0304
	subf r7,r19,r16	sub r7,r16,r19
common	addi r3,0,7000	li r3,7000
	addis r3,0,7000	lis r3,7000
	or r4,r5,r5	mr r4,r5
	nor r4,r5,r5	not r4,r5
	xor r7,r7,r7	clr r7
	eqv r8,r8,r8	set r8
special purpose	mtrcf %1111111,r5	mtrcr r5
condition	creqv 7,7,7	crset 7
	crxor 8,8,8	crclr 8
	cror 7,8,8	crmv 7,8
	crnor 8,9,9	crnot 8,9
rotates and shifts	rlwnm r8,r7,r6,0,31.	rotlw r8,r7,r6
	rlwimi r3,r3,24.,8,23.	inslwi r3,r3,16.,8
	rlwimi r8,r3,17,8,23.	insrwi r8,r3,7,8
	rlwinm r6,r4,8,0,14	extlwi r6,r4,15,8
	rlwinm r6,r4,16,24,31	extrwi r6,r4,8,8
	rlwinm. r6,r4,4,0,31	rotlwi. r6,r4,4
	rlwinm r6,r4,28,0,31	rotrwi r6,r4,4
	rlwinm r6,r4,1,0,30	slwi r6,r4,1
	rlwinm r6,r4,31,1,31	srwi r6,r4,1
	rlwinm r6,r4,0,1,31	clrlwi r6,r4,1
	rlwinm r6,r4,0,0,7	clrrwi r6,r4,14
	rlwinm r6,r4,6,6,25	clrlslwi r6,r4,12,6

Symbols and Source Code in the Analyzer

Chapter 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.

You can download symbols from certain object file formats into HP logic analyzers.

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

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 symbols, the rest of this chapter describes the requirements and considerations for displaying object file symbols and source code for PPC603 address values captured by a logic analyzer.

Symbols

Symbols represent values in measurements. For example, the symbol INTERRUPT might represent the value 1FF04000 found on the ADDR label, the address where your interrupt handler begins. You can trigger a measurement on the occurrence of a symbol, and you can have the logic analyzer show symbols in place of values in Listing displays.

Three symbol sources may be used in the logic analyzer:

- Object-file symbols
- Predefined PPC603 symbols
- User-defined symbols

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.

Predefined PPC603 symbols

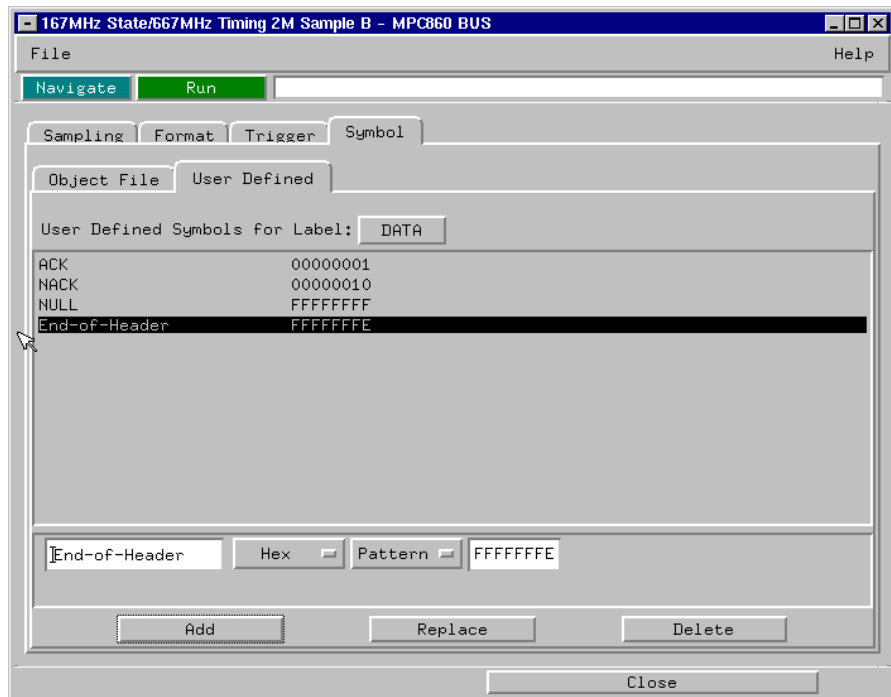
If you're using an analysis probe for the PPC603 microprocessor, the logic analyzer configuration files include predefined symbols.

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

The predefined PPC603 symbols are listed on page 105.

User-defined symbols

User-defined symbols are symbols you create from within the logic analyzer user interface by assigning names to values that can be found on the labeled bits. Typically, you assign symbol names to address label values, but you can also define symbols for values on the data, status, or other labels as well. The screen below shows a set of symbols for values found on the DATA label.



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

Symbol use requirements

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

An accurate bus trace

An HP E2455B analysis probe can help the logic analyzer capture PPC603 microprocessor data.

Direct address translation

The Memory Management Unit must perform direct address translation. Otherwise, captured addresses may not be correlated to the correct symbols.

An inverse assembler for trace lists

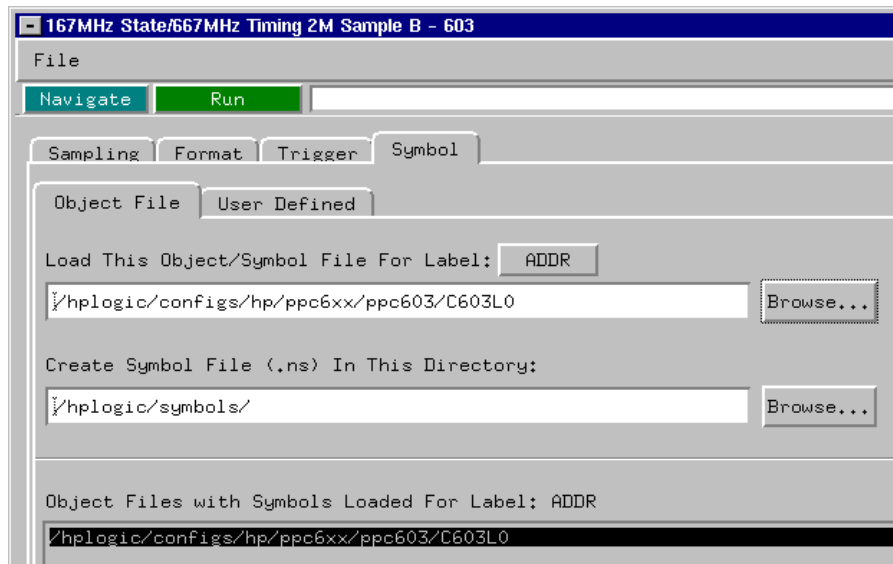
The inverse assembler software is included with HP analysis probes. The PPC603 inverse assembler decodes captured data into program counter (PC) addresses (also known as software addresses) and assembly language mnemonics. Refer to the previous chapter on PPC603 inverse assembly.

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 271).

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

To load object file symbols for address values 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.

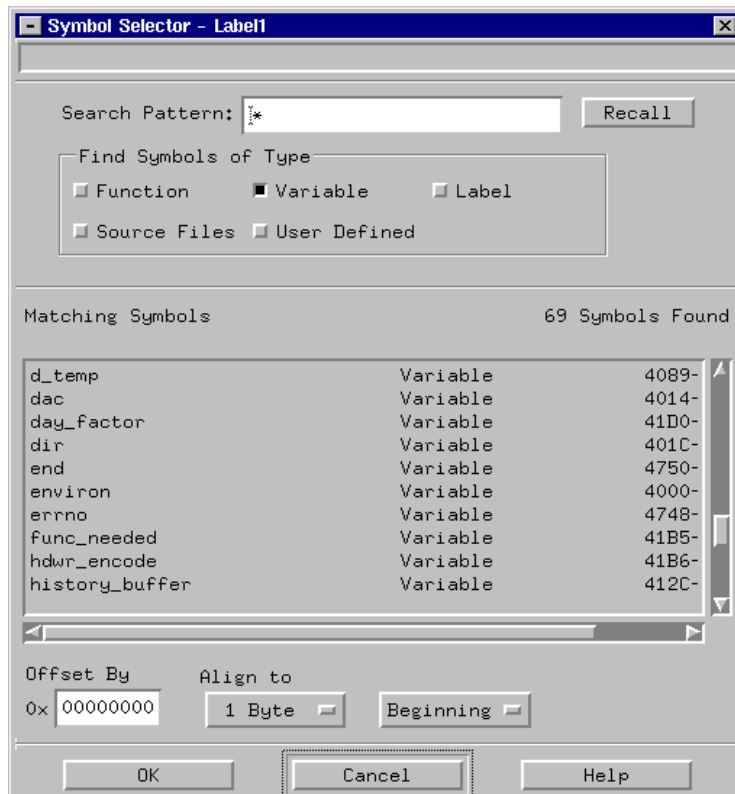


When you load object file symbols into a logic analyzer, a database that correlates symbols and line numbers to addresses in the object file is generated. The Symbol Selector dialog allows you to view the database so you can find a symbol to use in place of a hexadecimal value when defining trigger patterns, trigger ranges, and so on.

To access the Symbol Selector dialog:

The Symbol Selector dialog may be accessed in various ways. One way to access the Symbol Selector dialog is from the **Search** tab in the Listing display.

- 1 Under the **Search** tab in the Listing display, click **Advanced Searching**.
- 2 In the Goto Pattern dialog, click **Define**.
- 3 In the Search Pattern dialog, select the **Symbols** numeric base.
- 4 Select **Pattern**, **Range**, **Not Pattern**, or **Not Range**.
- 5 Click the field to the right of the **Pattern/Range** selection button.
- 6 In the Symbol Selector dialog that appears, select the symbol you want to use.



If your language tool is not one of those listed on page 163, you can create a

Symbols

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.

- If you have an HP 16600/700A-series logic analysis system, refer to the online help.
- If you have another logic analyzer, refer to your logic analyzer documentation.

Compilers for PPC603

The following PPC603 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 PPC603 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=PPC603	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 = ppc603 (or -cpu = ppc603e)	Specifies code generation for the PPC603 (or PPC603e) 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 PPC603 (or PPC603e) 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.
-p603	Specifies code generation for the PPC603 processor.

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

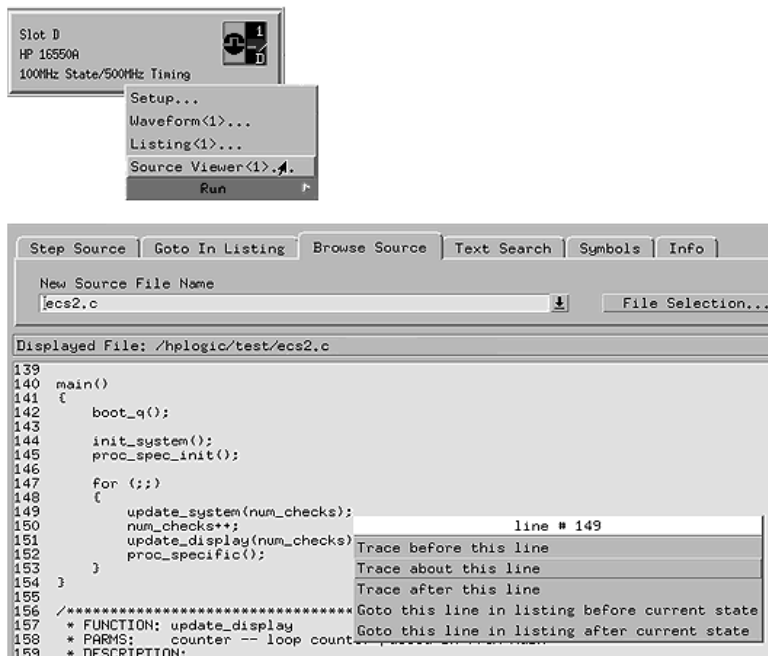
More information is available on the World Wide Web at: **<http://www.mentorg.com/microtec>**

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.



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 E2455B analysis probe meets this requirement.

The logic analyzer's inverse assembler may need to reconstruct incomplete address bus information and/or filter out unexecuted instructions. Also, the Memory Management Unit must perform direct address translation.

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” paragraph in the “Symbols” section, page 157, 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 PPC603 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 set up 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 PPC603 execution:

- Use the logic analyzer trigger alignment to avoid missed triggers.
- 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 trigger alignment

You should use an 8-bit alignment to avoid missed triggers. The PPC603 has a 64-bit data bus. Instructions for the PPC603 are 32 bits long and must be located on even address boundaries. This means that an instruction will often be fetched as the lower 32 bits of one 64-bit memory cycle. When this happens, the address of the instruction in the lower 32 bits of the fetch will not be seen on the address bus. If a trigger was set to occur on this instruction's address, the trigger will not be found by the logic analyzer. For example:

Bus Activity			High Level	
Address	Data	Mnemonic	Line	C-Source
00000080	39600000	li r11,0	#13	i = 0;
00000084	39400001	li r10,1	#14	j = 1;
00000088	7D4A5A14	add r10,r10,r11	#15	k = j+i;

In the above example, instruction fetches will occur at addresses 80 and 88; a trigger set on line #14 (address 84) will not be detected. The instruction at address 84 was actually fetched with the 64-bit memory fetch at address 80, so you needed to trigger on address 80 to catch the fetch of address 84.

To help avoid these missed triggers, the trigger dialogs for symbol addresses allow you to “Align” the address to a 1-, 2-, 4-, or 8-byte boundary.

Alignment affects the least significant address bits of the trigger specification, either setting them to a “don't care” or “zero” value, depending on the logic analyzer.

Set the alignment for program fetches to the width of the program memory in bytes. For the PPC603 with 64-bit (8-byte) wide program memory, use 8-byte alignment. Eight-byte alignment will change the least significant three bits ($2^3 = 8$) of the trigger. Address 84 with 8-byte alignment results in a trigger address range of 80 through 87 for some logic analyzers (3 don't care bits), or an address of 80 on the other analyzers (three 0 bits). Note that either of these triggers would catch line #14 in the example above.

To correlate relocatable code 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 offset field in the trigger menu allows you to offset the symbol address. Entering the appropriate address offset will cause the source correlation tool set to reference the correct symbol information for the relocatable or offset code.

To adjust for prefetches, use a trigger offset of 8 (prefetch queue depth) to avoid triggering on prefetched instructions. Note that this is not a foolproof scheme, since this may result in a missed trigger if a branch takes place between the base address and the offset address. For the PPC603, an offset of 8 is large enough to overcome the prefetch queue.

Be aware of prefetches and adjust your triggering to compensate for them. Note that the PPC603 has a good Branch Prediction Unit (PBU), and often, when executing loops, the processor will NOT fetch instructions beyond the end of the loop. This means that on most loops, an offset may not be required to avoid a false trigger.

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.). To set up the store qualification to store only non-idle states, first verify that the Store Qualify field is set to "Term Selection", (as opposed to "Anystate"). Next, replace the STAT label with the ACKs label. While the cursor is over the STAT label, hold down the right mouse button, select the "Replace label..." option, select the ACKs label from the list. Now change the format of the ACKs label to Binary Not Pattern. Finally, edit the pattern field to become "X1111". Only states that are not Idle will be stored.

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.

Connecting and Configuring the Emulation Module

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 178.)
- 2** Install the emulation module in your logic analysis system, if necessary. (Page 185.)

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. (Page 190.)
- 4** Update the firmware of the emulation module, if necessary. (Page 192.)
- 5** Verify communication between the emulation module and the target. (Page 193.)
- 6** Configure the emulation module. (Page 194.)
- 7** Test the connection between the emulation module and the target. (Page 209.)
- 8** Connect a debugger to the emulation module, if applicable. (Page 211.)

See Also

“Using the Emulation Module with a Debugger” beginning on page 211 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.

Chapter 8: Connecting and Configuring the Emulation Module

Using the Emulation Control Interface

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:

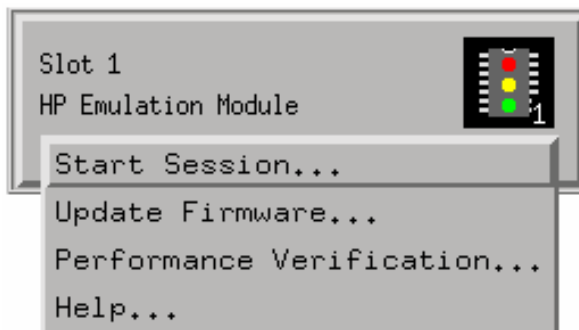
- The interfaces can get out of synchronization when commands are issued from both interfaces. This causes windows to be out-of-date and can cause confusion.
- Most debuggers cannot tolerate another interface issuing commands and may not start properly if another interface is running.

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.

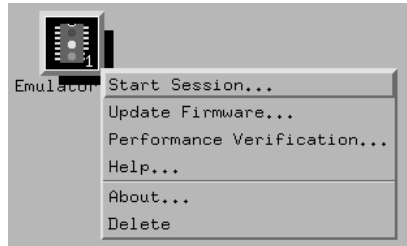
To start the Emulation Control Interface from the main System window

- 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 Emulation Module 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

The following sections describe design considerations for your target system to operate properly with the emulation module. Note that there are different requirements for PPC603 and PPC603e target systems.

Design requirements for PowerPC 603 and PowerPC 603e

QACK signal

If the target development board does not use the $\overline{\text{QACK}}$ signal, the board must have a pull down resistor to drive this signal low. This allows the PowerPC 603 to enter the debug state, and the PowerPC 603e to enter the state required for reading and writing processor scan string data. Recommended value: 1K Ω or less.

If the target system uses reduced pinout mode, $\overline{\text{QACK}}$ must be high during $\overline{\text{HRESET}}$, but must be low to enter debug mode. If neither $\overline{\text{QACK}}$ nor $\overline{\text{QREQ}}$ are being used, these two pins can be tied together and pulled high with a 1 K Ω pullup. If $\overline{\text{QACK}}$ or $\overline{\text{QREQ}}$ is used by the target, logic must be supplied to make $\overline{\text{QACK}}$ low after $\overline{\text{QREQ}}$ is asserted so that the PowerPC can go into debug mode.

TDO, TDI, TCK, TMS and $\overline{\text{TRST}}$ signals

TDO, TDI, TCK, TMS and $\overline{\text{TRST}}$ signal traces between the JTAG connector and the PPC603 must be less than 3 inches long. If these signals are connected to other nodes, the other nodes must be daisy chained between the JTAG connector at one end and the PowerPC microprocessor at the other end. These signals are sensitive to crosstalk and must not be routed along active signals such as clock lines on the target board.

The TDI, TCK, TMS and $\overline{\text{TRST}}$ signals must not be actively driven by the target system when the debug port is being used.

Reset signals

The $\overline{\text{HRESET}}$, $\overline{\text{SRESET}}$ and $\overline{\text{TRST}}$ signals from the JTAG connector must be logically ORed with the $\overline{\text{HRESET}}$, $\overline{\text{SRESET}}$ and $\overline{\text{TRST}}$ signals that connect to the processor on the target system. They cannot be “dotted” or “wire-ORed” on the board. The ORed signals should only reset the processor and no other devices on the target system.

The HP emulation module adds capacitance to all target system signals routed to the JTAG 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.

Additional target requirements may be specified in the release notes in the “readme” file on the provided floppy disk.

Operating considerations for the PowerPC 603

Mask revision dd3

Mask revision dd3 of the PowerPC 603 chip can be used with the HP emulation module.

Unsupported modes

Target systems which use any of the following modes of operation are not currently supported:

- MMU when it is used for address translation. MMU may be used for memory protection as long as no address translation is being performed.
- Address parity is not generated on external address bus operations. Accesses to devices that check parity will fail.

Operating considerations for the PowerPC 603e

MMU Support for mask revisions 3, 4, and 5

Full MMU support is provided for PowerPC 603e silicon with Mask Revisions dd3, dd4, and dd5, and for the PowerPC 603ev

When the MMU is enabled in the PowerPC hardware, and the HP Emulator is configured for effective addresses, all memory addresses given to the emulator are assumed to be effective addresses (logical addresses). The emulator uses the MMU block address translation (BAT) registers, segment registers, hash tables, and other special-purpose MMU registers to compute each corresponding physical address. The requested memory operation is then performed using the physical address.

Operational notes:

- The emulator attempts to perform address translation only if the MSR[IR] and/or the MSR[DR] bits are set (=1) AND the emulator is configured to do translation (cf address=effective). The emulator configuration may be changed using the cf command:
 - cf address=effective (power up default value)
 - cf address=physical
- If both the MSR[IR] and MSR[DR] are set, the emulator will perform address translations by first searching the IBATs and then the DBATS, if no match is found in the IBATs. Note that the PowerPC silicon allows the IBAT and DBAT registers to specify overlapping effective address ranges. Avoid defining overlapping ranges. These make debugging more difficult because the emulator can use the IBATs to translate addresses intended for the DBATs.
- If an effective address is not found in the MMU translation tables, the emulator will return an error and will not perform the requested operation.
- Cache coherency is maintained during emulator MMU translations.
- Be sure the translation enable/disable condition is the same when you set and clear breakpoints. If a breakpoint is set while translation is enabled and then cleared while translation is disabled, the result will be erroneous and unpredictable. This is also true if a breakpoint is set while translation is disabled and then cleared while translation is enabled.

- The emulator ignores read-only restrictions defined in the MMU. (i.e. The emulator may attempt to write to memory that has been defined by the MMU as read-only.)
- MMU translation is automatic and transparent to debuggers connected to the emulator.

Cache support

Mask rev dd1 of the 603e silicon (no longer in production) does not work in debug mode with either cache enabled. Current versions (Mask rev dd3, IBM; Mask rev dd4, Motorola) all work with caches on. If cache is enabled during downloading, performance will be slowed by the activity required to maintain cache coherency. The default is to maintain cache coherency and accept the slower downloading performance. Note that the current 603ev is also fully supported.

Unsupported modes

Target systems which use any of the following modes of operation are not currently supported:

- No MMU support is offered for Mask Revision 1 when the MMU is used for address translation. Only physical memory addresses are accessible. MMU may be used for memory protection as long as no address translation is being performed.
- Address parity is not generated on external address bus operations. Accesses to devices that check parity will fail.
- If the processor runs to a branch to self instruction (op code 48000000H) and the instruction is at an address ending with 04H or 0CH, the emulation module will not be able to soft stop the processor. If the processor doesn't soft stop, it will be "hard stopped" and an error message will be generated. Once the processor is hard stopped, memory and register contents can be read, but the processor cannot be stepped or run at this point. A reset is required to restore operation.

Operating considerations for the PowerPC 603ev

MMU support

Full support for address translation is available using the MMU.

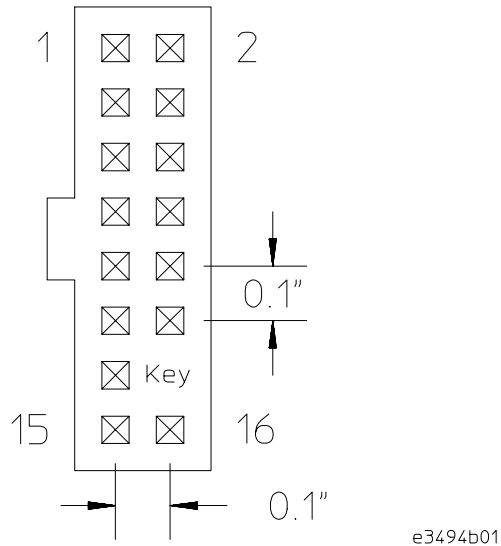
Unsupported modes

Target systems which use any of the following modes of operation are not currently supported:

- Address parity is not generated on external address bus operations. Accesses to devices that check parity will fail.
- If the processor runs to a branch to self instruction (op code 48000000H) and the instruction is at an address ending with 04H or 0CH, the emulation module will not be able to soft stop the processor. If the processor doesn't soft stop, it will be “hard stopped” and an error message will be generated. Once the processor is hard stopped, memory and register contents can be read, but the processor cannot be stepped or run at this point. A reset is required to restore operation.

PowerPC JTAG interface connections and resistors

The JTAG interface on the target system must be a 16-pin male 2x8 header connector with the following dimensions:



JTAG Header Connector (top view)

Position 14 of the JTAG connector on the target system must not contain a pin. The cable supplied with the emulation module can only be installed if pin 14 has been removed from the header.

Place the connector as close as possible to the processor to ensure signal integrity. Refer to the next page for signals to be connected to each of the connector pins.

Chapter 8: Connecting and Configuring the Emulation Module
Designing a Target System for the Emulation Module

PowerPC 6xx Connections

Header Pin Number	Signal Name	I/O	Board Resistor
1	TDO	Out	
2	Not connected		
3	TDI	In	1K Ω pulldown
4	$\overline{\text{TRST}}$	In	10K Ω pullup
5	Not connected		
6	+POWER ¹		1K Ω series ²
7	TCK	In	10K Ω pullup
8	Not connected		
9	TMS	In	10K Ω pullup
10	Not connected		
11	$\overline{\text{SRESET}}$	In	10K Ω pullup
12	Not connected		
13	$\overline{\text{HRESET}}$	In	10K Ω pullup
14	KEY		
15	$\overline{\text{CHECKSTOP}}$ ³	Out	1K Ω pullup
16	GND		
	$\overline{\text{QACK}}$ ⁴	In	1K Ω pulldown
	L2_TEST_CLK	In	10K Ω pullup
	L1_TEST_CLK	In	10K Ω pullup
	$\overline{\text{LSSD_MODE}}$	In	10K Ω pullup
	ARRAY_WR	In	10K Ω pullup

¹ The +POWER signal is sourced from the target system and is used as a reference signal. It should be the power signal being supplied to the processor (either +3.3V or +5V). It does not supply power to the HP emulation module.

² This 1K Ω series resistor provides short circuit current limiting protection only. If the resistor is present, it should be 1K Ω or less.

³ For the PowerPC 604 processors, this line is called $\overline{\text{CKSTP_OUT}}$.

⁴ If the target system does not use this signal, the board must have a 1K Ω pulldown resistor connected to this pin. This signal allows the HP emulation module to force the processor into soft stop mode. If the target system does use this signal, it should provide logic so that $\overline{\text{QACK}}$ goes low in response to a $\overline{\text{QREQ}}$.

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.

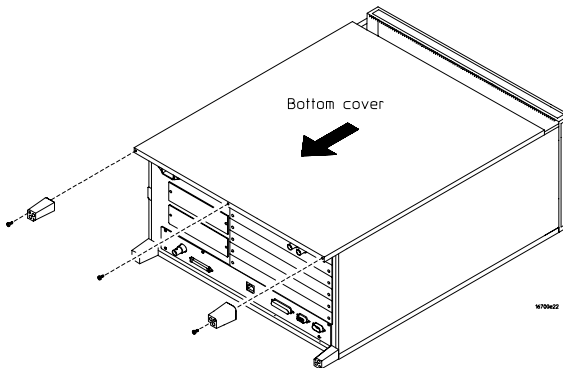
CAUTION:

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

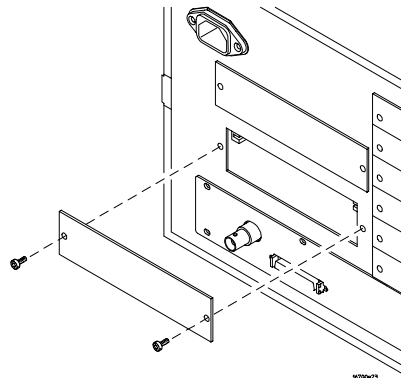
To install the emulation module in an HP 16700A-series logic analysis system or an HP 16701A expansion frame

You will need T-10 and T-15 Torx screwdrivers (supplied with the emulation module).

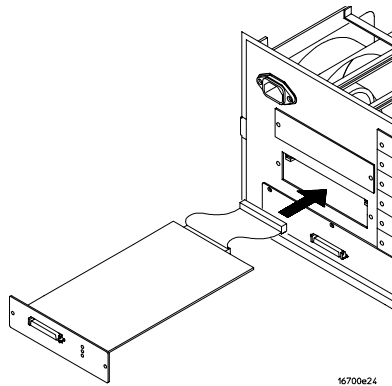
- 1 Turn off the logic analysis system and REMOVE THE POWER CORD.
Remove any other cables (including mouse or video monitor cables).
- 2 Turn the logic analysis system frame upside-down.
- 3 Remove the bottom cover.



- 4 Remove the slot cover.
You may use either slot.

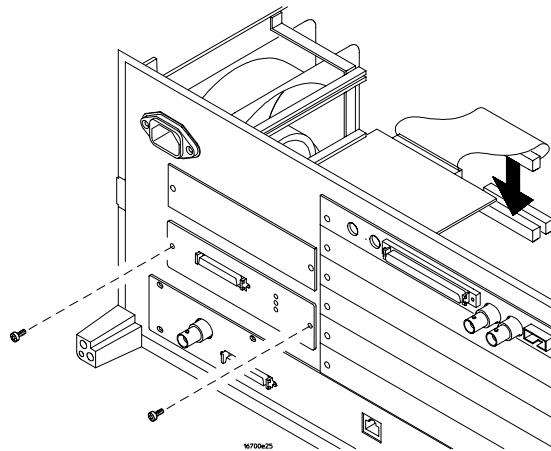


- 5 Install the emulation module.



- 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 192 for information on giving the emulation module a “personality” for your target processor.

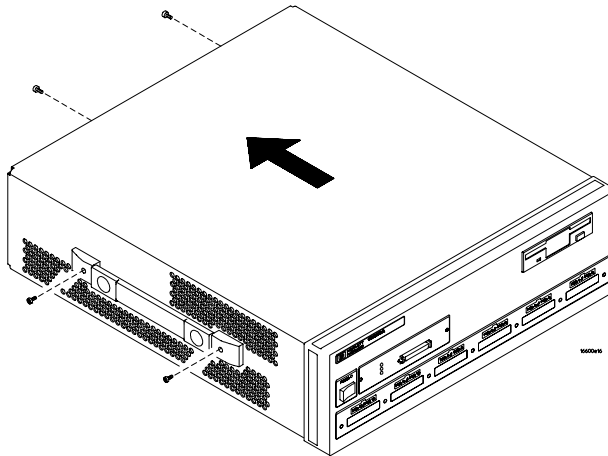
To install the emulation module in an HP 16600A-series logic analysis system

You will need T-8, T-10, and T-15 Torx screwdrivers (supplied with the emulation module).

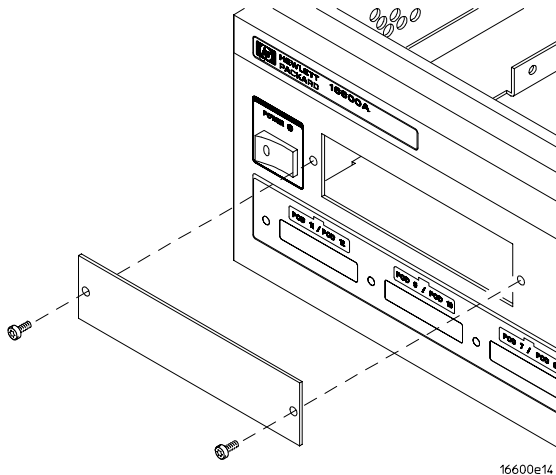
- 1 Turn off the logic analysis system and REMOVE THE POWER CORD.

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

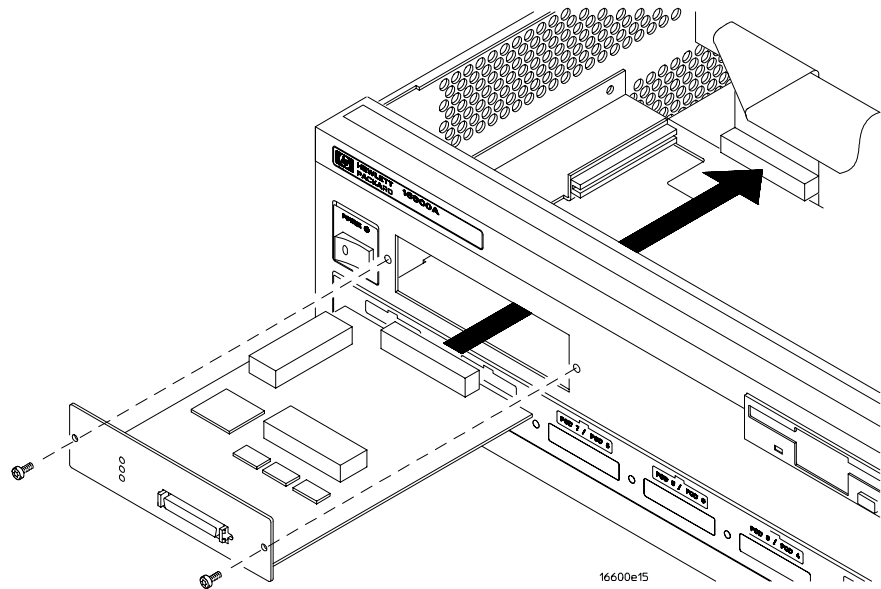
- 2 Slide the cover back.



- 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

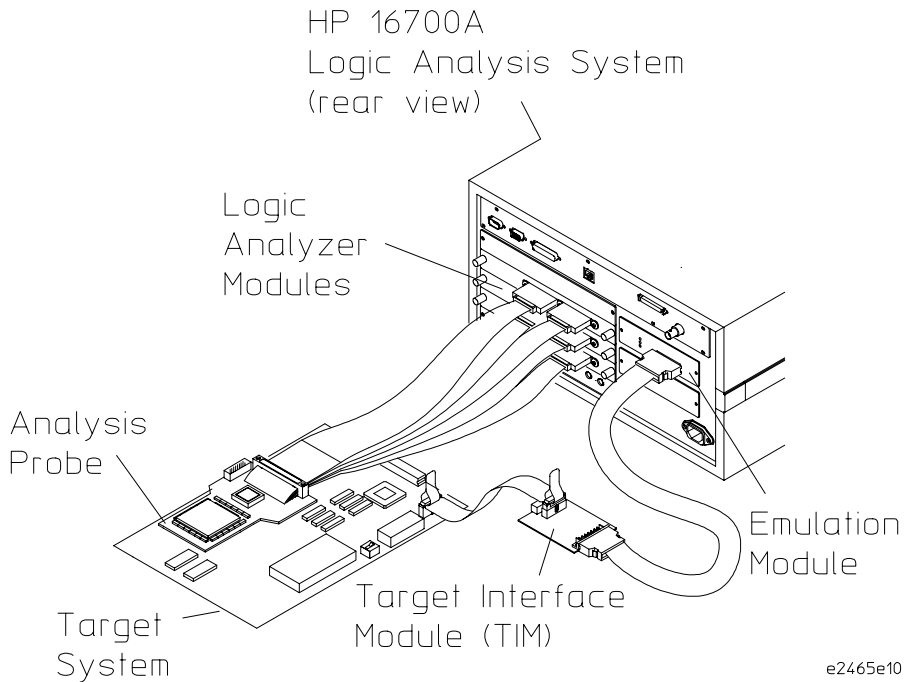
See page 192 for information on giving the emulation module a “personality” for your target processor.

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 319 for information on performance verification.

Connecting the Emulation Module to the Target System

Connect the emulation module and TIM to a target system directly through a JTAG connector on the target board.



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

See Also

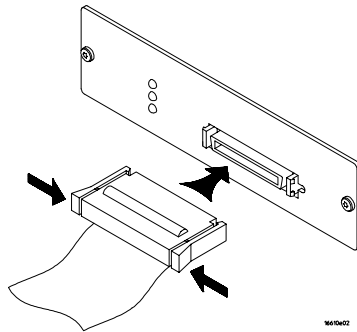
For information on designing a JTAG port on your target board, see page 178.

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

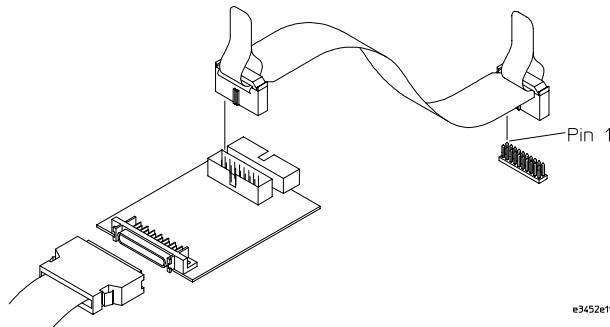
To connect to a target system using a JTAG port

The emulation module can be connected to a target system through a 16-pin JTAG port connector. The emulation module should be connected to a 16-pin male 2x8 header connector on the target system using the 16-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 16-pin cable into the target interface module.
- 5 Plug the other end of the 16-pin cable into the JTAG 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 178) for information on designing a target system for use with the emulation module.

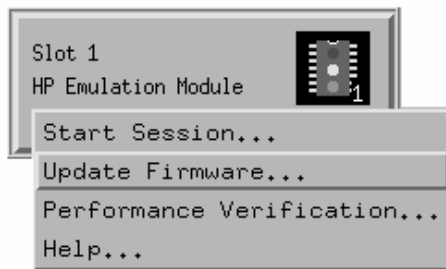
To Update Firmware

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

There are two categories of configuration items: general configuration and cache configuration.

The default power-up configuration will generally work with many target systems if the cache is turned off.

If the instruction and data caches are both turned off, the cache configuration items are meaningless and can be ignored.

The old mask rev dd1 of the 603e silicon does not work in debug mode with either cache enabled. The current mask rev of 603e silicon works with cache enabled.

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.

General Configuration

- JTAG clock speed
- Reset operation
- Memory read delays
- Memory write delays
- Parity bit information
- Data bus size

Cache Configuration

- Enable data retry mode
- Memory read operation
- Data memory write operations
- Instruction memory write operations

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 303.

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 ---
rrt              - Restrict to real-time runs
reset            - Configure reset actions
speed            - Set JTAG clock
mrdop            - Configure mem read operation
dmwrop           - Configure D mem write operation
imwrop           - Configure I mem write operation
mrddel           - Set memory read delay
mwrddel          - Set memory write delay
breakin          - Select BNC break input option
trigout          - Select BNC trigger output option
parity           - Enable/disable data parity
drtry            - Select DRTRY mode
32bitmode        - Enable/disable 32 bit mode

M>
```

Chapter 8: Connecting and Configuring the Emulation Module

Configuring the Emulation Module

To see a more detailed description of any configuration item, use the command “help cf <item>”. For example:

```
M>help cf rrt
```

```
Restrict to real-time runs
```

```
cf rrt=yes  
cf rrt=no
```

```
If yes (and while the processor is running the user program), any  
command that requires the processor to be stopped will be rejected.  
For example 'reg' and 'm'.
```

```
If no, commands that require the processor to be stopped will  
actually stop the processor, execute then resume running the  
processor.
```

```
M>
```

To see a list of the current configuration settings, use “cf”:

```
M>cf
```

```
cf rrt=yes  
cf reset=runrom  
cf speed=1  
cf mrdop=mm  
cf dmwrop=mm  
cf imwrop=upd_dcu  
cf mrddel=0  
cf mwrddel=0  
cf breakin=off  
cf trigout=fixhigh  
cf parity=off  
cf drtry=off  
cf 32bitmode=off
```

```
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 211).

Your debugger manual.

To configure restriction to real-time runs

Real-time runs configuration

Value	Emulation module configured for	Built-in command
no	Allows commands which break to the monitor. Examples include commands which display memory or registers. These commands break to the monitor to access the target processor, then resume the user program.	<code>cf rrt=no</code>
yes	No commands are allowed which break to the monitor, except "break," "reset," "run," or "step." The processor must be explicitly stopped before these commands can be performed. (Default)	<code>cf rrt=yes</code>

If your debugger allows displaying or modifying memory or registers while the processor is running, you must set `rrt=no` in order to use this feature.

To configure the JTAG clock speed (communication speed)

The HP emulation module needs to be configured to communicate at a rate which is compatible with your target processor. The JTAG Clock speed is independent of processor clock speed. In general, speed=1 can always be used and provides the best performance. With some target systems that have additional loads on the JTAG lines or with target systems that do not quite meet the requirements described in the “Designing a Target System” chapter (page 178), setting speed to a slower setting may enable the module to work.

Processor clock speed configuration

Value	Processor clock (TCK) is at least	Built-in command
1	10 MHz (default)	cf speed=1
2	5 MHz	cf speed=2
3	2.5 MHz	cf speed=3
4	1.25 MHz	cf speed=4
5	625 kHz	cf speed=5
6	312 kHz	cf speed=6
7	156 kHz	cf speed=7

To configure reset operation

The reset configuration item controls what kind of reset is performed and what state the processor will be in after the reset.

Reset configuration

Value	Effect of a reset from the emulation module	Built-in command
runrom	Reset the processor and cause it to start running user code at address FFF00100H.(Default)	<code>cf reset=runrom</code>
rom	Reset the processor and cause it to stop at address 0FFF00100H.	<code>cf reset=rom</code>
runram	Reset the processor and cause it to start running user code at address 00000100H.	<code>cf reset=runram</code>
ram	Reset the processor and cause it to stop at address 00000100H.	<code>cf reset=ram</code>
jtag	Just reset the JTAG interface on the processor. The processor itself will not be reset. This may help in some cases where communications are lost, however all the other reset settings reset the JTAG interface as part of the reset sequence so this setting will only rarely be useful.	<code>cf reset=jtag</code>

To set memory read delays

The memory read delay setting delays the number of microseconds specified during memory reads. It is provided for accessing slow devices like memory mapped IO.

- To set the memory read delay using the built-in terminal interface, use the `cf mrddel=<delay in usec>` command.

The *<delay in usec>* must be in the range 0-10000000. This should be set to the smallest number possible for best performance since it delays all reads by the number of microseconds specified.

Default: `cf mrddel=0`

To set memory write delays

The memory write delay setting delays memory writes by the number of microseconds specified. It is provided for accessing slow devices like memory mapped IO.

- To set the memory write delay using the built-in terminal interface, use the `cf mwrde1=<delay in usec>` command.

The *<delay in usec>* must be in the range 0-10000000. This should be set to the smallest number possible for best performance.

Default: `cf mwrde1=0`

To generate parity bits on memory operations

The PowerPC processor generates parity bits on both address and data lines when running user code. When used in debug mode, these bits must be generated separately, slowing down memory operations. Since memory operations on the PowerPC are slow as it is and many target systems do not check parity, parity is only generated if requested.

Parity configuration

Value	Emulation module configured for	Built-in command
off	Do not generate the parity bits for memory operations from the emulation module. This provides better performance, but will not work correctly when accessing devices that check the parity bits.(Default)	<code>cf parity=off</code>
on	Generate the parity bits for memory operations. Currently, only parity bits for the memory data lines are generated. Parity bits on the address lines are not. This may change in future firmware versions.	<code>cf parity=on</code>

To set data bus size

The PowerPC 603 and 603e processors support both 32-bit and 64-bit data buses. The processor determines bus size based on the value of the $\overline{\text{TLBISYNC}}$ pin during a reset. The 32bitmode cf option must match the data bus size of the target system for proper memory reads and writes from the emulation module.

Bus configuration

Value	emulation module configured for	Built-in command
off	Processor is in 64 bit mode.(Default)	cf 32bitmode=off
on	Processor is in 32 bit mode. Maximum access size used will be 32 bits. (Access size can still be set to 8, but the emulation module will only do 4 byte memory accesses.)	cf 32bitmode=on

To enable data retry mode

The PowerPC 603 processor can be powered on with data retry mode enabled or not, depending on the state of the $\overline{\text{DRTRY}}$ processor pin. This cannot be detected by the HP emulation module. This mode is only used when the cache is enabled. If either the instruction and/or data cache is enabled, this configuration item must match the state of the processor as determined by the $\overline{\text{DRTRY}}$ pin on power-up. If the cache is disabled, this setting has no effect. The debug mode of the PowerPC 603e does not require different actions based on the $\overline{\text{DRTRY}}$ mode so this configuration item does not exist for the 603e.

If either the instruction or data cache is enabled on the 603, the DRTRY setting must be set correctly for your target system. Failure to do so will cause incorrect data to be displayed or written and possibly communication problems between the emulation module and the target system.

DRTRY configuration

Value	Emulation module configured for	Built-in command
off	Disable the data retry mode of the PowerPC 603. (Default)	<code>cf drtry=off</code>
on	Enable the data retry mode of the PowerPC 603	<code>cf drtry=on</code>

To configure the memory read operation

The memory read operation configuration entry defines how the memory and cache interact during a memory read operation. If both instruction and data caches are turned off (bits ICE and DCE in the register HID0 are zero), this configuration setting has no effect and a memory read will always return the contents of physical memory.

Memory read configuration

Value	emulation module configured for	Built-in command
mm	A memory read from an address that is valid in either the data or instruction cache will return the contents of the cache. Memory reads from addresses not valid in either cache will return the contents of the physical memory.(Default)	cf mrdop=mm
phys	A memory read will always return the contents of physical memory.	cf mrdop=phys

Using the mrdop=phys setting with the cache enabled may show data that is no longer valid. Use this setting only for solving cache problems where you really need to see the contents of physical memory. For general operation, the "mm" setting should always be used.

The instruction cache in PPC603e is encoded. The emulator will decode the content of the instruction cache before displaying it. However, the emulator will only decode valid instructions. Invalid instructions in the cache will be displayed in coded form, which might not match the content of memory.

To configure data memory write operations

Although the PowerPC processor has one contiguous physical memory address space that can hold both data and instructions, it has separate caches for instructions and data. These separate caches must be considered in order to keep the caches and memory coherent during memory write operations. These settings are only used for memory write operations. Code download always writes to physical memory and disables any cache entries containing addresses written for improved performance. Some host interfaces use the code download mode for all memory write operations so this setting may or may not have any effect on your debugger.

Only the memory write command allows specifying instruction or data memory operations. This may not be provided by your debugger interface. If not specified, memory write operations are always instruction memory.

If the data cache is disabled, a data memory write will always write to physical memory and this configuration setting is ignored.

Memory write configuration

Value	emulation module configured for	Built-in command
mm	A data writes to addresses that are valid in the data cache will write the value only to the cache and mark the cache line modified as "dirty" which will indicate to the cpu that the cache line must be written to memory. A data write that is not valid in the data cache will only be written to physical memory.(Default)	<code>cf dmwrop=mm</code>
thru	A data memory write to an address that is valid in the data cache will write to both cache and physical memory. If the address is not valid in the cache, only physical memory will be modified.	<code>cf dmwrop=thru</code>
bypass	A data memory write will only be written to physical memory ignoring the cache.	<code>cf dmwrop=bypass</code>

The `cf dmwrop=bypass` setting should be used with extreme caution because dirty cache entries may be written by the processor over the new data value written to memory by the emulation module.

To configure instruction memory write operations

Although the PowerPC processor has one contiguous physical memory address space that can hold both data and instructions, it has separate caches for instructions and data. These separate caches must be considered in order to keep the caches and memory coherent during memory write operations. Code download always writes to physical memory and disables any cache entries containing addresses written for improved performance. Some host interfaces use the code download mode for all memory write operations so this setting may or may not have any effect on your debugger.

Only the memory write command allows specifying instruction or data memory operations. Access to this may not be provided by your debugger interface. If not specified, memory write operations are always instruction memory.

If the instruction and data caches are both disabled, an instruction memory write will always write to physical memory and this configuration setting is ignored. If the instruction cache is disabled, instruction memory writes will always write to physical memory and the data cache will be either updated or bypassed depending on this configuration setting.

This configuration setting controls the behavior of both caches when doing instruction memory writes so that instruction memory writes can be used for all memory operations if desired. See the table on the next page.

Chapter 8: Connecting and Configuring the Emulation Module

Configuring the Emulation Module

Instruction memory write configuration

Value	Emulation module configured for	Built-in command
upd_dcb	This stands for instruction cache update, data cache bypass. An instruction memory write to an address that is valid in the instruction cache will write the value to both the instruction cache and memory. The data cache will be bypassed even if the address is valid in the data cache.	cf imwrop = upd_dcb
upd_dcu	This stands for update instruction cache and update data cache. An instruction memory write to an address that is valid in both caches will write the value to both caches and physical memory. (Default)	cf imwrop = upd_dcu
inv_dcb	This stands for instruction cache invalidate and data cache bypass. An instruction memory write will invalidate the instruction cache if valid and write only to physical memory. The data cache is not modified even if valid.	imwrop = inv_dcb
inv_dcu	This stands for instruction cache invalidate and data cache update. An instruction memory write will invalidate the instruction cache if valid and write to physical memory. The data cache will also be updated if the address is valid in the data cache	imwrop = inv_dcu

Setting imwrop to upd_dcb or inv_dcb should be used with caution since dirty cache entries in the data cache may overwrite the memory just modified by the HP emulation module.

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 299 for information on testing the emulator hardware.

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.

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.

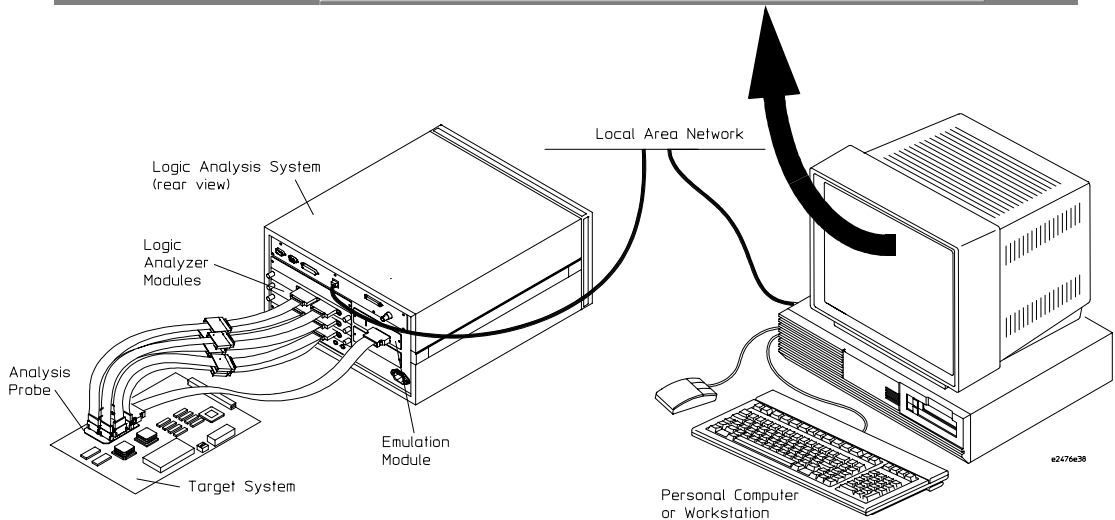
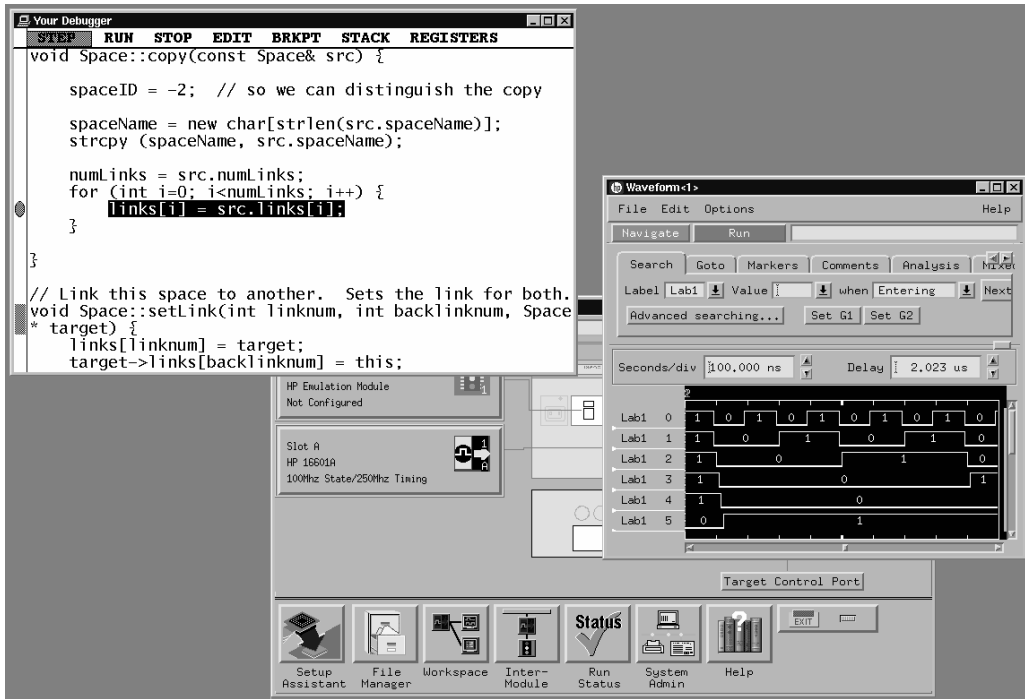
Using a debugger to connect the emulation module allows the entire design team to have a consistent interface from software development to hardware/software integration.

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 174).
- Connect the logic analysis system to the LAN (page 216).
- Export the logic analysis system's display to your PC or workstation (page 219).
- Configure the emulation module (page 194).
- 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 be greater than 1024 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 317, 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**.

To enable or disable processor caches

The Power PC 6xx processors have instruction and data caches. Debugging using an third party debugger will have the greatest performance if the caches are disabled during debugging. There are three ways to disable the caches prior to a debug session:

- Set bits 16 (ICE) and 17 (DCE) of register HID0 to zero (bit 0 being the MSB). This will turn off the I and D caches.

Ensure that your startup code does not reset the HID0 register. This could re-enable the caches.

The following code will invalidate and disable the caches:

```
mfscr    r3 hid0
ori      r3 0C00          # set ICFI and DCFI
mtscr    hid0 r3
rlwinm  r3 r3 0 22 19    # clear ICFI and DCFI
mtscr    hid0 r3
rlwinm  r3 r3 0 18 15    # clear ICE and DCE
mtscr    hid0 r3
isync
```

- Issue the following probe commands:

"cf reset=rom"

"rst" ("rst" will turn off all caches)

Ensure that your startup code does not reset the HID0 register after the "rst" command because this could re-enable the caches.

- Keep the caches enabled but tell the HP emulation module to bypass them. To do this, issue the probe commands:

"cf mrdop=phys" (so only physical memory is read)

"cf dmwrop=bypass" (to bypass the updating of the data cache) reference
all addresses with the @dmem modifier.

Example:

M> cf mrdop=phys

M> cf dmwrop=bypass

M> m -d4 -a4 0.. (this will read physical memory only)

M> m -d4 -a4 0@dmem=12345678 (this will write physical memory only)

When caches are bypassed (cf `dmwrop=bypass`), all memory write accesses are made to physical memory and the cache information is ignored. **This means that cache coherency is not maintained.**

If cache handling is not modified using one of the above three methods, execution with the third party debugger may be slower due to the HP emulation module making sure the cache information stays coherent with physical memory.

Using the Green Hills debugger

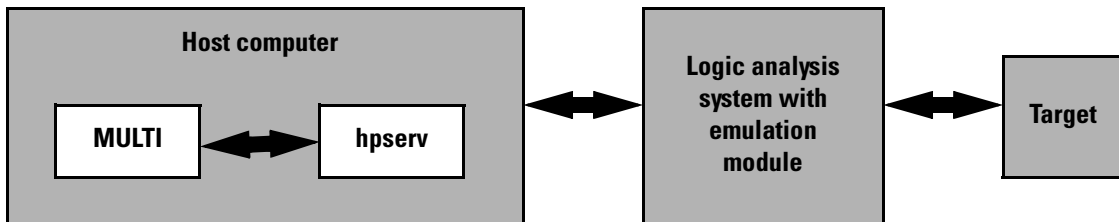
Compatibility

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 a PPC603 processor.

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a PowerPC 603 Emulator. If it is not, follow the instructions on page 192 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

Chapter 9: Using the Emulation Module with a Debugger

Using the Green Hills debugger

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 216 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.

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 226 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.

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 PPC603 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 PowerPC 603 Emulator. If it is not, follow the steps on page 192 to update the firmware in the Emulation module for PPC603 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 194 for more information on configuration options and the “cf” command.

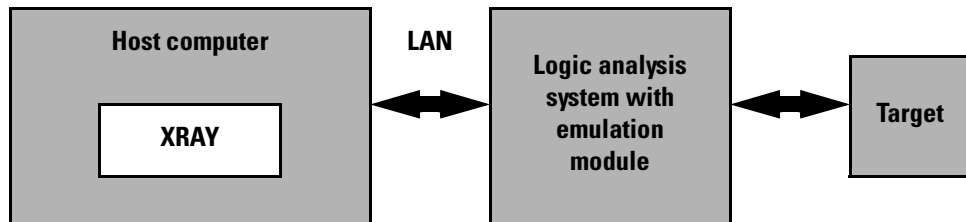
Using the Microtec Research debugger

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 PPC603 processor.

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a Motorola PowerPC 603 Emulator. If it is not, follow the instructions on page 192 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 216 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. If the configuration for your target board is very involved, 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 232 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. Microtec Research provides an example include file in its tools directory under the `xhippchip` 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 PPC603 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 PPC603 PowerPC Emulator. If it is not, follow the steps on page 192 to update the firmware in the Emulation module for PPC603 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 194 for more information on configuration options and the “cf” command.

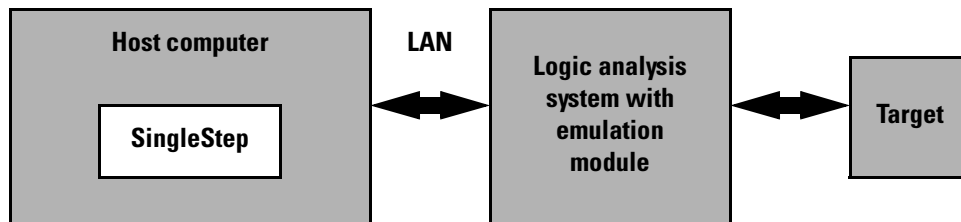
Using the Software Development Systems debugger

Compatibility

Version 7.2, 7.3, and 7.4 of the SingleStep debugger from Software Development Systems, Inc. are some of the debuggers which 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.
- 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 PPC603 processor:

Go to the system window of the logic analyzer interface and verify that the Emulation Module icon is described as a “PowerPC 603 Emulator”. If it is not, follow the instructions on page 192 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 217 for more information on port numbers.

- 3 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); it can list the actual commands used to initialize the executable, or it may point to a file of type .cfg which contains the actual initialization commands. The config aliases contained in sstrp.ini are provided by SDS for some common targets. Refer to the SDS SingleStep User's Guide for information about additional initialization techniques.

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.

Chapter 9: Using the Emulation Module with a Debugger

Using the Software Development Systems debugger

Initialization of the target (that is, execution of the `_config` alias) will not actually occur until the “Debug” dialog is successfully exited.

- 4 Set up the download and execution options in the Options tab of the Debug dialog.
- 5 Download the application and run:

You can significantly increase download performance by disabling the caches. Disable caches by writing the appropriate bits to the HIDD register (see page 126).

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 what is defined to happen at reset you would issue the following command in the SingleStep command window:

```
control -c "cf reset"
```

To change the reset definition you would issue the following command in the command window:

```
control -c "cf reset=runrom"
```

For more information about "terminal interface" commands see page 194.

Cache Disabling

(See "To enable or disable processor caches" section on page 220)

The default "sstep.ini" file sent with the PPC603 Single Step sets the HID0 register to 0, indicating that caches will be disabled.

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 PPC603. Verify that the emulation module is connected to the target through a TIM. Next, go to the system window of the logic analyzer interface and verify that the Emulation Module icon (stop-light) is described as a PowerPC 603 Emulator. If it is not, follow the steps on page 192 to update the firmware in the Emulation module for PPC603 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 317 for more information.

See Also

The SDS web site: <http://www.sdsi.com>

The *SDS SingleStep Users Guide*.

The configuration section beginning on page 194 for more information on configuration options and the "cf" command.

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

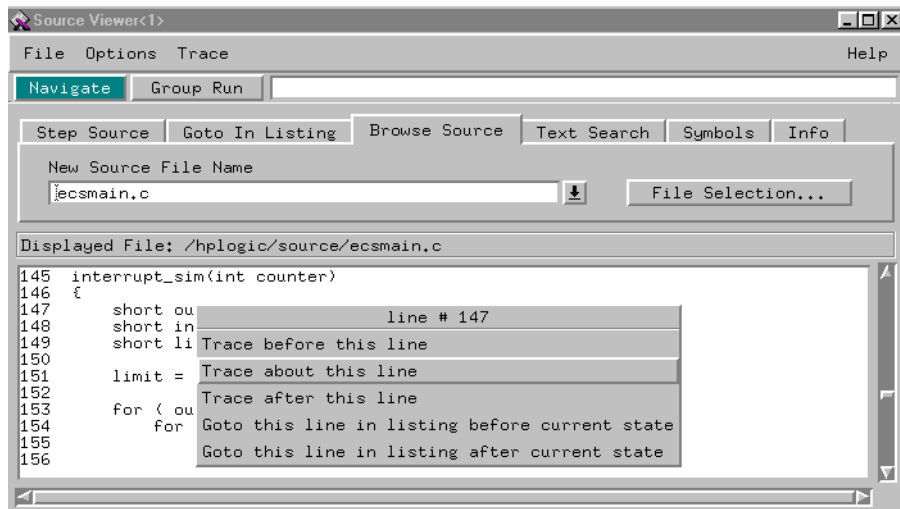
The logic analyzer may be used to signal the emulation module to stop (break) the target processor. This is done from either the Source Viewer window or the Intermodule window. If you are using the HP B4620B Source Correlation Tool Set, using the Source Viewer window is the easiest method.

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**.

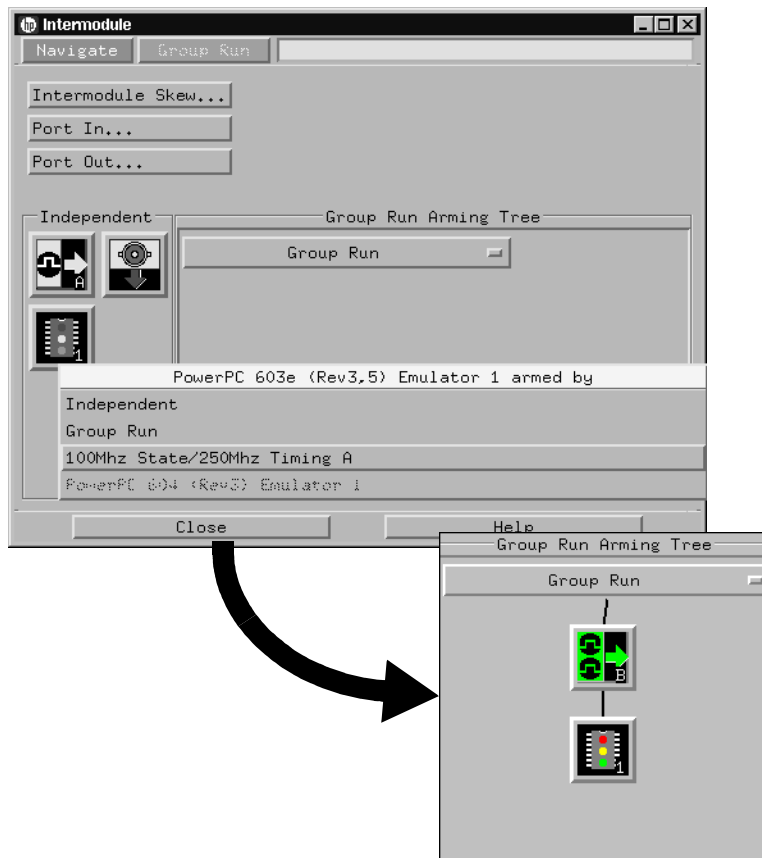
4 Click **Group Run** in the Source window (or other logic analyzer window).

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 200 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 trigger point (position) to **End**.
- 3 In a logic analyzer window, click **Run**.
- 4 In the Emulation Control Interface or debugger click **Run**.
- 5 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 250.

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 sent in the following circumstances:

- 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:

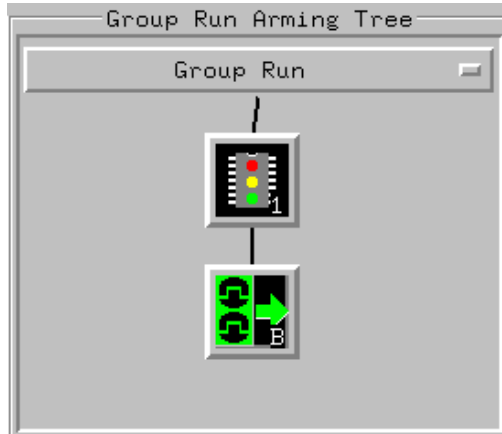
- If Group Run is armed from Port In and an emulation module is connected to Group Run, then any Port In signal will cause the emulation module to go into monitor. The Group Run button does not have to be clicked for this to operate.
- If two emulation modules are connected together so that one triggers 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 - timing mode

If your processor halts unexpectedly, and you would like to see timing information on your bus prior to the halt, set up this measurement.

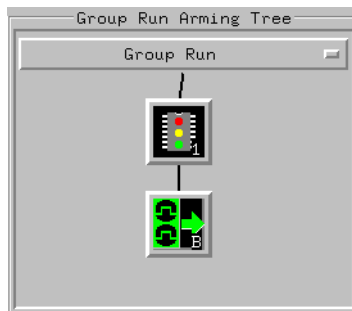
The following example shows how to set up an HP 16600A/16700A logic analysis system with VisiTrigger. This measurement can also be set up using HP 16600A/16700A logic analysis systems without VisiTrigger, and HP 1660/1670/16500-series logic analysis systems.

NOTE:

If you are only using a state analyzer to capture the processor bus, it will be much simpler to use “Tracing until processor halts” as described on page 250.

HP 16600A/16700A with VisiTrigger

- 1 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. This sets the logic analyzer to trigger when the processor halts..

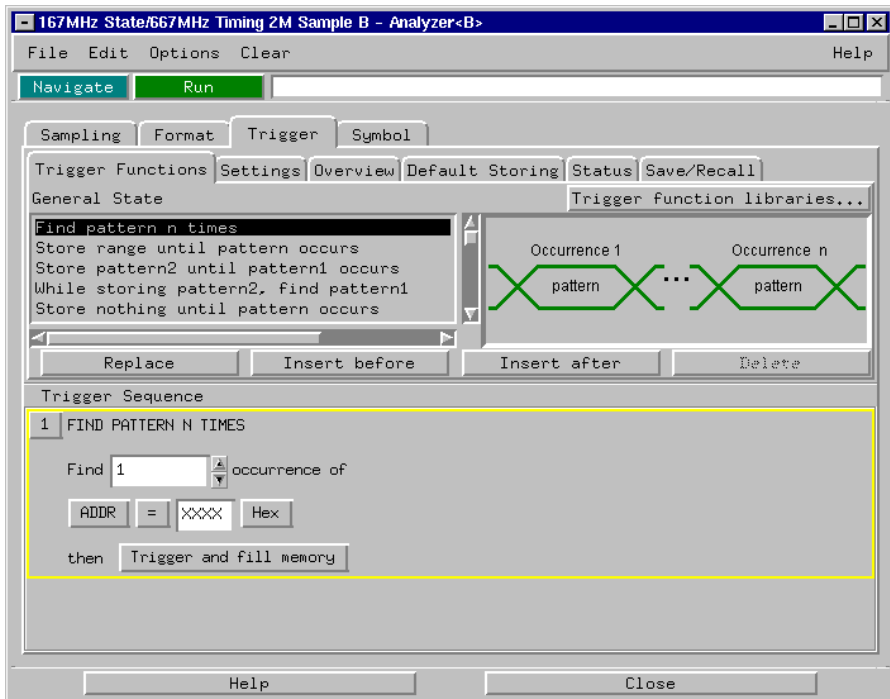


Now continue to step 2.

Chapter 10: Using the Analysis Probe and Emulation Module Together

Triggering the Logic Analyzer from the Emulation Module

- 2 Set the sampling mode to timing and set the trigger as shown below:



- 3 Set the trigger position to **end**.
- 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 halted.

The logic analyzer will store states until the processor halts.

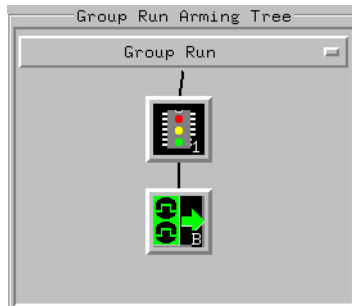
To trigger the analyzer when the processor reaches a breakpoint

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 250.

- 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).

Chapter 10: Using the Analysis Probe and Emulation Module Together

Triggering the Logic Analyzer from the Emulation Module

- 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.

Hardware Reference

Chapter 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 E2455B analysis probe and the inverse assembly 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 E2455B PPC603 analysis probe.

Operating Characteristics	
Microprocessor Compatibility	PowerPC 603, PowerPC 603e, and PowerPC 603ev
Package Supported	240-pin PQFP
Microprocessor Clock Speed	66 MHz maximum
Logic Analyzers Supported	HP 16715/16/17A, HP 16710/11/12A, (one or more cards). HP 16600A, HP 16601A, HP 1660A/AS/C/CS/CP, HP 1670A/D. HP 16550A (two cards). HP 16554A/55A/56A (two or three cards). HP 16555D/56D/57D (two or three cards).
Accessories Required	The QFP probe adapter (included with the HP E2455B) is required.
Pods Required	Eight 16-channel pods (four high-density adapter cables) are required for disassembly. Ten 16-channel pods (5 high-density adapter cables) are available.
Electrical Characteristics	
Power Requirements	None.
Signal Line Loading	10pF, 100 kohms on all signals.
Setup/Hold Requirement	For all signals, the logic analyzers require a minimum combined setup/hold window. For the HP 16600-series logic analysis system, the combined setup/hold must be at least 4.5 ns (such as 0/4.5, 1.0/3.5. etc.). For all other logic analyzers, the combined window must be at least 3.5 ns.
Pollution Degree	Pollution Degree 2: Normally only dry non-conductive pollution occurs. Occasionally a temporary conductivity caused by condensation must be expected.

Environmental Characteristics	
Temperature	Operating: 0 to + 50 degrees C
Altitude	Operating: 4,600 m
Humidity	Up to 75% noncondensing. Avoid sudden, extreme temperature changes which could cause condensation on the circuit board.
	For indoor use only.

Theory of operation

The HP E2455B analysis probe is a passive probe. All signals are routed through to the logic analyzer without passing through any additional circuitry.

Analysis probe—signal-to-connector mapping

The following tables show the electrical signal-to-connector mapping implemented by the HP E2455B PPC603 Analysis Probe and the Inverse Assembler Software.

PowerPC 603 Signal List

Connector	2x19 pin	2x10 pin	2x20 pin	Analyzer Bit	PowerPC 603 Pin #	Signal Name	Analyzer Label	Analyzer Label
J1odd	6	3	3	CLK1	212	SYSCLK	SYSCLK	
J1odd	8	4	7	15	166	A16	ADDR	
J1odd	10	5	9	14	15	A17	ADDR	
J1odd	12	6	11	13	165	A18	ADDR	
J1odd	14	7	13	12	16	A19	ADDR	
J1odd	16	8	15	11	164	A20	ADDR	
J1odd	18	9	17	10	17	A21	ADDR	
J1odd	20	10	19	9	160	A22	ADDR	
J1odd	22	11	21	8	21	A23	ADDR	
J1odd	24	12	23	7	157	A24	ADDR	
J1odd	26	13	25	6	22	A25	ADDR	
J1odd	28	14	27	5	158	A26	ADDR	
J1odd	30	15	29	4	23	A27	ADDR	
J1odd	32	16	31	3	151	A28	ADDR	
J1odd	34	17	33	2	30	A29	ADDR	
J1odd	36	18	35	1	144	A30	ADDR	
J1odd	38	19	37	0	37	A31	ADDR	
J1even	5	3	3	CLK1	--			
J1even	7	4	7	15	179	A0	ADDR	
J1even	9	5	9	14	2	A1	ADDR	
J1even	11	6	11	13	178	A2	ADDR	
J1even	13	7	13	12	3	A3	ADDR	
J1even	15	8	15	11	176	A4	ADDR	
J1even	17	9	17	10	5	A5	ADDR	
J1even	19	10	19	9	175	A6	ADDR	
J1even	21	11	21	8	6	A7	ADDR	
J1even	23	12	23	7	174	A8	ADDR	
J1even	25	13	25	6	7	A9	ADDR	
J1even	27	14	27	5	170	A10	ADDR	
J1even	29	15	29	4	11	A11	ADDR	
J1even	31	16	31	3	169	A12	ADDR	
J1even	33	17	33	2	12	A13	ADDR	
J1even	35	18	35	1	168	A14	ADDR	
J1even	37	19	37	0	13	A15	ADDR	

Chapter 11: Hardware Reference

Connector	2x19 pin	2x10 pin	2x20 pin	Analyzer Bit	PowerPC 603 Pin #	Signal Name	Analyzer Label	Analyzer Label
J2odd	6	3	3	CLK1	235	QACK	QACK-	
J2odd	8	4	7	15	214	HRESET	HRSET-	STAT
J2odd	10	5	9	14	215	CKSTP	CKSTP-	STAT
J2odd	12	6	11	13	216	CHECKSTOP	CHKOUT	STAT
J2odd	14	7	13	12	219	BR	BR-	STAT
J2odd	16	8	15	11	224	TC0	TC0 TC	STAT
J2odd	18	9	17	10	223	TC1	TC1 TC	STAT
J2odd	20	10	19	9	236	WT	WT-	STAT
J2odd	22	11	21	8	237	CI	CI-	STAT
J2odd	24	12	23	7	1	GBL	GBL-	STAT
J2odd	26	13	25	6	25	DBWO	DBWO-	STAT
J2odd	28	14	27	5	26	DBG	DBG-	STAT
J2odd	30	15	29	4	27	BG	BG-	STAT
J2odd	32	16	31	3	28	AACK	AACK- acks	STAT
J2odd	34	17	33	2	31	QREQ	QREQ-	STAT
J2odd	36	18	35	1	32	ARTRY	ARTRY- acks	STAT
J2odd	38	19	37	0	36	ABB	ABB-	STAT
J2even	5	3	3	CLK1	180	TT4	TT4	
J2even	7	4	7	15	197	TSIZ0	TSIZ	STAT
J2even	9	5	9	14	196	TSIZ1	TSIZ	STAT
J2even	11	6	11	13	195	TSIZ2	TSIZ	STAT
J2even	13	7	13	12	192	TBST	TSIZTBST-	STAT
J2even	15	8	15	11	191	TT0	AtomicTT	STAT
J2even	17	9	17	10	190	TT1	R/-WTT	STAT
J2even	19	10	19	9	185	TT2	InvldtTT	STAT
J2even	21	11	21	8	184	TT3	A OnlyTT	STAT
J2even	23	12	23	7	189	SRESET	SRSET-	STAT
J2even	25	13	25	6	188	INT	INT-	STAT
J2even	27	14	27	5	156	DRTRY	DRTRY-acks	STAT
J2even	29	15	29	4	155	TA	TA-acks	STAT
J2even	31	16	31	3	154	TEA	TEA-	STAT
J2even	33	17	33	2	150	XATS	XATS	STAT
J2even	35	18	35	1	149	TS	TS	STAT
J2even	37	19	37	0	145	DBB	DBB	STAT

Connector	2x19 pin	2x10 pin	2x20 pin	Analyzer Bit	PowerPC 603 Pin #	Signal Name	Analyzer Label	Analyzer Label
J3odd	6	3	3	CLK1				
J3odd	8	4	7	15	117	DL16	DATA_B	
J3odd	10	5	9	14	107	DL17	DATA_B	
J3odd	12	6	11	13	106	DL18	DATA_B	
J3odd	14	7	13	12	105	DL19	DATA_B	
J3odd	16	8	15	11	102	DL20	DATA_B	
J3odd	18	9	17	10	101	DL21	DATA_B	
J3odd	20	10	19	9	100	DL22	DATA_B	
J3odd	22	11	21	8	51	DL23	DATA_B	
J3odd	24	12	23	7	52	DL24	DATA_B	
J3odd	26	13	25	6	55	DL25	DATA_B	
J3odd	28	14	27	5	56	DL26	DATA_B	
J3odd	30	15	29	4	57	DL27	DATA_B	
J3odd	32	16	31	3	58	DL28	DATA_B	
J3odd	34	17	33	2	62	DL29	DATA_B	
J3odd	36	18	35	1	63	DL30	DATA_B	
J3odd	38	19	37	0	64	DL31	DATA_B	
J3even	5	3	3	CLK1	153	DBDIS	DBDIS-	
J3even	7	4	7	15	143	DL0	DATA_B	
J3even	9	5	9	14	141	DL1	DATA_B	
J3even	11	6	11	13	140	DL2	DATA_B	
J3even	13	7	13	12	139	DL3	DATA_B	
J3even	15	8	15	11	135	DL4	DATA_B	
J3even	17	9	17	10	134	DL5	DATA_B	
J3even	19	10	19	9	133	DL6	DATA_B	
J3even	21	11	21	8	131	DL7	DATA_B	
J3even	23	12	23	7	130	DL8	DATA_B	
J3even	25	13	25	6	129	DL9	DATA_B	
J3even	27	14	27	5	126	DL10	DATA_B	
J3even	29	15	29	4	125	DL11	DATA_B	
J3even	31	16	31	3	124	DL12	DATA_B	
J3even	33	17	33	2	123	DL13	DATA_B	
J3even	35	18	35	1	119	DL14	DATA_B	
J3even	37	19	37	0	118	DL15	DATA_B	

Chapter 11: Hardware Reference

Connector	2x19 pin	2x10 pin	2x20 pin	Analyzer Bit	PowerPC 603 Pin #	Signal Name	Analyzer Label	Analyzer Label
J4odd	6	3	3	CLK1				
J4odd	8	4	7	15	85	DH16	DATA	
J4odd	10	5	9	14	84	DH17	DATA	
J4odd	12	6	11	13	83	DH18	DATA	
J4odd	14	7	13	12	82	DH19	DATA	
J4odd	16	8	15	11	81	DH20	DATA	
J4odd	18	9	17	10	80	DH21	DATA	
J4odd	20	10	19	9	78	DH22	DATA	
J4odd	22	11	21	8	76	DH23	DATA	
J4odd	24	12	23	7	75	DH24	DATA	
J4odd	26	13	25	6	74	DH25	DATA	
J4odd	28	14	27	5	73	DH26	DATA	
J4odd	30	15	29	4	72	DH27	DATA	
J4odd	32	16	31	3	71	DH28	DATA	
J4odd	34	17	33	2	68	DH29	DATA	
J4odd	36	18	35	1	67	DH30	DATA	
J4odd	38	19	37	0	66	DH31	DATA	
J4even	5	3	3	CLK1	--			
J4even	7	4	7	15	115	DH0	DATA	
J4even	9	5	9	14	114	DH1	DATA	
J4even	11	6	11	13	113	DH2	DATA	
J4even	13	7	13	12	110	DH3	DATA	
J4even	15	8	15	11	109	DH4	DATA	
J4even	17	9	17	10	108	DH5	DATA	
J4even	19	10	19	9	99	DH6	DATA	
J4even	21	11	21	8	98	DH7	DATA	
J4even	23	12	23	7	97	vDH8	DATA	
J4even	25	13	25	6	94	DH9	DATA	
J4even	27	14	27	5	93	DH10	DATA	
J4even	29	15	29	4	92	DH11	DATA	
J4even	31	16	31	3	91	DH12	DATA	
J4even	33	17	33	2	90	DH13	DATA	
J4even	35	18	35	1	89	DH14	DATA	
J4even	37	19	37	0	87	DH15	DATA	

Connector	2x19 pin	2x10 pin	2x20 pin	Analyzer Bit	PowerPC 603 Pin #	Signal Name	Analyzer Label	Analyzer Label
J5odd	6	3	3	CLK1			CLKOUT ¹	
J5odd	8	4	7	15	231	AP0	AP	
J5odd	10	5	9	14	230	AP1	AP	
J5odd	12	6	11	13	227	AP2	AP	
J5odd	14	7	13	12	226	AP3	AP	
J5odd	16	8	15	11	186	MCP	MCP-	
J5odd	18	9	17	10	187	SMI	SMI-	
J5odd	20	10	19	9	198	TDO	TDO	
J5odd	22	11	21	8	200	TMS	TMS	
J5odd	24	12	23	7	199	TDI	TDI	
J5odd	26	13	25	6	202	TRST	TRST-	
J5odd	28	14	27	5	--			
J5odd	30	15	29	4	205	LSSDMODE	LSSDMO	
J5odd	v32	16	31	3	213	PLLCF0	PLLCFG	
J5odd	34	17	33	2	211	PLLCF1	PLLCFG	
J5odd	6	18	35	1	210	PLLCF2	PLLCFG	
J5odd	38	19	37	0	208	PLLCF3	PLLCFG	

¹ CLKOUT is not probed by the analysis probe.

J5even	5	3	3	CLK1	201	TCK	TCK	
J5even	7	4	7	15	204	L1TSTCLK	L1Tclk	
J5even	9	5	9	14	203	L2TSTCLK	L2Tclk	
J5even	11	6	11	13	218	APE	APE	
J5even	13	7	13	12	217	DPE	DPE	
J5even	15	8	15	11	225	CSE	CSE	
J5even	17	9	17	10	232	RSRV	RSRV-	
J5even	19	10	19	9	234	TBEN	TBEN	
J5even	21	11	21	8	233	TBLISYNC	TBLISY	
J5even	23	12	23	7	38	DP0	DP	
J5even	25	13	25	6	40	DP1	DP	
J5even	27	14	27	5	41	DP2	DP	
J5even	29	15	29	4	42	DP3	DP	
J5even	31	16	31	3	46	DP4	DP	
J5even	33	17	33	2	47	DP5	DP	
J5even	35	18	35	1	48	DP6	DP	
J5even	37	19	37	0	50	DP7	DP	

Emulation module—operating characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the HP 16610A emulation module and PPC603 target interface module.

Operating Characteristics	
Microprocessor Compatibility	PPC603, PPC603e, PPC603ei, and PPC603ev 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 PowerPC 603 emulation module	Symbol	Min	Max
TDO, CHECKSTOP	V_{ih}	2.0 V	5.5 V
	V_{il}		0.8 V
	I_i		$\pm 1 \mu\text{A}$
	C_{in}		15 pF
TDI, TCK, TMS, $\overline{\text{TRST}}$ ¹	$V_{oh} @ I_{oh} = -32 \text{ mA}$	2.0 V	2.8 V
	$V_{ol} @ I_{ol} = 64 \text{ mA}; V_{CC} = 4.5\text{V}$		0.55 V
TDI, TMS, $\overline{\text{TRST}}$	C_o		25 pF
TCK	C_o		45 pF
+3.3V Power Sense ²	V_{ih}	2.0 V	5.3 V
	V_{il}	-0.3 V	0.8 V
$\overline{\text{SRESET}}, \overline{\text{HRESET}}$ ³	$V_{ol} @ I_{ol} = 12 \text{ ma}$		0.5 V
	C_o		25 pF
TS0 - TS6, SYSCLK	C_{in}		10 pF
	V_{ih}	2.0 V	5.5 V
	V_{il}		0.8 V
	I_i		$\pm 1 \mu\text{A}$

¹ These signals must not be actively driven by the target system when the debug port is being used.

² Power Sense is used only to determine target powered status. The emulator module does not draw power from this source.

³ Open collector outputs, pulled up to a generated voltage equivalent to the Power Sense voltage with a 2.61 K pullup resistor.

General-Purpose ASCII (GPA)
Symbol File Format

Chapter 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 163, 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.

Troubleshooting the Analysis Probe

Chapter 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, “Connecting and Configuring the Analysis Probe,” beginning on page 41 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 126.

- ❑ 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 “Configuring the Logic Analysis System” on page 79 for details.

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”

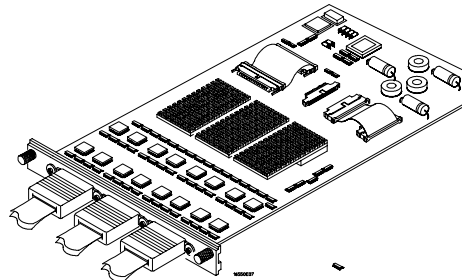
This error occurs if you rename or delete the inverse assembler file that is attached to the configuration file.

Ensure that the inverse assembler file is not renamed or deleted, and that it is located in the correct directory:

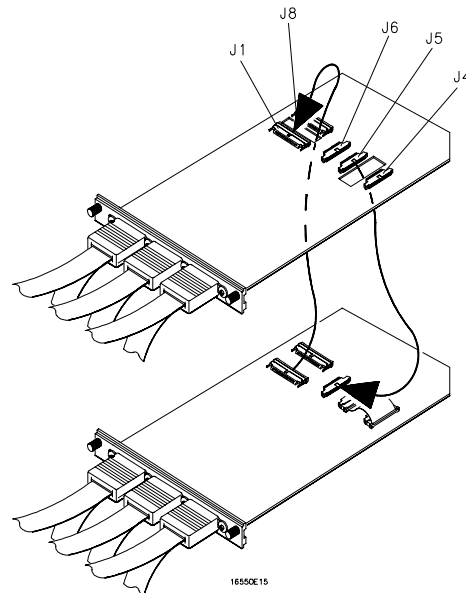
- For HP 16600A/700A-series logic analysis systems it should be in /hplogic/ia.
- For other logic analyzers it should be in the same directory as the configuration file.

“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

See page 79 for information on 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, “Connecting and Configuring the Analysis Probe,” beginning on page 41, 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 Parts

HP Part Number	Description
E2455-68703	Software Disk Pouch
E5363A #201	Retainer Kit
E5363A #202	Locator Tool
E5350-23801	Cam Tool
E5363A	Adapter, Elastomeric, 240-pin QFP
E5396A	High-density Adapter 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.

Chapter 13: Troubleshooting the Analysis Probe
Cleaning the Instrument

Troubleshooting the Emulation
Module

Chapter 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 303
Emulation module built-in commands do not work	<ol style="list-style-type: none">1 Check that the emulation module has been properly configured for your target system.2 Run the emulation module performance verification tests.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 153 page 319, page 320 page 32, page 306
"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 160
Some commands fail	Check the "restrict to real-time runs" configuration.	page 162

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
<input type="radio"/> Reset <input type="radio"/> Break <input type="radio"/> Run	No target system power, or emulation module is not connected to the target system
<input checked="" type="radio"/> Reset <input type="radio"/> Break <input type="radio"/> Run	Target system is in a reset state
<input type="radio"/> Reset <input checked="" type="radio"/> Break <input type="radio"/> Run	The target processor is executing in Debug Mode
<input type="radio"/> Reset <input type="radio"/> Break <input checked="" type="radio"/> Run	The target processor is executing user code
<input type="radio"/> Reset <input checked="" type="radio"/> Break <input checked="" type="radio"/> 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
c	Target is checkstopped
p	No target power
d	No target interface module connected to emulation module
?	Unknown state

Examples

To set register R0, then view R0 to verify that it was set, enter:

```
R>rst -m
M>reg GPR0=ffff
M>reg GPR0
    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.

Solving Target System Problems

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 GPR1  
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 303 for information on entering built-in commands.

To check the debug port connector signals

- Check for the following logic levels on the target debug port. The signal names are for the PPC 6xx.

Levels with the emulation module not connected

Header Pin	Signal Name	Level
3	TDI	Low
4	$\overline{\text{TRST}}$	High
6	+POWER	V_{DD}
7	TCK	High
9	TMS	High
11	$\overline{\text{SRESET}}$	High
13	$\overline{\text{HRESET}}$	High
15	$\overline{\text{CHECKSTOP}}$	High
16	GND	Low

Levels with the emulation module connected

Header Pin	Signal Name	I/O
1	TDO	Toggle with "es" command
3	TDI	Toggle with "es" command
4	$\overline{\text{TRST}}$	Low pulse with "rst" command
6	+POWER	V_{DD}
7	TCK	10+ MHz clock (default)
9	TMS	Low, pulse with "es" command
11	$\overline{\text{SRESET}}$	High, pulse low with "rst" command
13	$\overline{\text{HRESET}}$	High, pulse low with "rst" command
15	$\overline{\text{CHECKSTOP}}$	High
16	GND	Low

For the COP on the PowerPC 604 to work reliably, the following pins must be tied high: L1_TSTCLK, L2_TSTCLK, ARRAY_WR, LSSD_MODE, DRVMOD0, and DRVMOD1.

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 6 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 “c>”

Processor is checkstopped. Something caused a machine exception before the emulation module connected or $\overline{\text{CHECKSTOP}}$ is being pulled or held low.

If the initial prompt is “?>” with “ERROR 171!”

A bad status code (0xXX) was received from the processor. Valid status is 0x01 or 0x05. Any other status indicates a bad scan of the instruction register. Check TCK, TDO, TDI, TMS, and $\overline{\text{TRST_L}}$ signals. Check the firmware revision.

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  
U>
```

The “U>” prompt is a good response that indicates SRESET and HRESET are working. Continue with “If the prompt after rst is U>”.

If the prompt after rst is “?>” with “ERROR 171!”

A bad status code (0xXX) was received from the processor. Valid status is 0x01, any other status indicates bad scan of IR or failure of the reset signals. Verify TCK, TDO, TDI, TMS, and $\overline{\text{TRST}}$ are all changing state on an $\overline{\text{HRESET}}$.

If the rst command fails

Set “cf reset=rom” (no external bus cycles used in this mode), then enter the “rst” command again:

```
*>cf reset=rom  
*>rst  
M>
```

You can enter these commands at any prompt, shown here as “*>”.

- If the prompt is “M>” with no error messages, all scans worked. We have control as long as we don't try to run code. Continue with “If you can get to the ‘M>’ prompt.”
- If an error message is displayed, verify that $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ are being driven.
- If the prompt is “c>”, there was bad scanning of the data scan chain. Check processor mask revision.
- If the prompt is “U>”, the processor failed to stop soft or hard. Check reset lines, mask revision, processor type and firmware version.

If the prompt after rst is “U>”

The $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ lines are working. Continue with more tests:

- 4 Enter the break command:

```
U>b  
M>
```

If the prompt after b is “M>” with error messages

If you see: “!ERROR 145! Unable to soft stop - freezing the processor clocks” the processor is hard stopped. Check the mask revision, processor type, and firmware version. If all of these look good, then the target may not be terminating cycles (pending external bus cycles). Successive run (“r”) and step (“s”) commands will fail. The processor may have fetched an invalid instruction.

Check the value of the PC (IAR):

```
M>reg PC  
reg PC=xxxxxxxx  
M>
```

If the value is fff00100, the processor had a problem accessing the boot ROM and crashed during boot.

Processor and/or board level reset is required to recover from “freezing processor clocks” -- register and memory commands should still work.

If the prompt after b is “M>” with no error messages

Everything is still working correctly. Continue with more tests:

If you can get to the “M>” prompt

- 5 At the “M>” prompt, check register and memory access:

```
M>reg GPR0  
reg GPR0=xxxxxxxx  
M>reg GPR0=12345678  
M>reg GPR0  
reg GPR0=12345678  
M>
```

Chapter 14: Troubleshooting the Emulation Module

Solving Target System Problems

If the returned value is equal to the written value, then the dd level of the chip is probably correct.

Now enter:

```
M>m -d4 -a4 0=11111111,22222222,33333333,44444444
M>m -d4 -a4 0..
  00000000  11111111  22222222  33333333  44444444
  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>

- Returned value is equal to the written value implies that memory is working.
- Returned value is not equal to the written value implies that memory control may not be initialized. Try to initialize by:

```
M>cf reset=runrom;rst;w 5
  #waiting for 5 seconds...
```

U>b

M>

Repeat above memory test.

- If every other word is wrong, set 32 bit mode:

```
M>cf 32bitmode=on
```

M>

Repeat above memory test.

6 At the “M>” prompt, check the processor's revision level:

This procedure currently works only on PPC603 and 602. The target must support burst cache fill from where PC is pointing.

If you have a Cogent board or a PPC 603 target, set the PC to a location in RAM. For example:

```
M>reg PC=100
```

M>

Now enter:

```
M>reg PVR
```

```
reg PVR=xxxxxxxx
```

M>

The returned value is in the form VVVVRrr where VVVV is the processor family, and Rrr is mask revision level.

VVVV:

```
0003 -> 603
0004 -> 604
0005 -> 602
0006 -> 603e
0007 -> 603ev
0009 -> 604ev
0020 -> 403
```

For example reg PVR=00030302 means 603 Mask Revision 3.2

If you see memory-related problems

1 Set caches and translation off:

```
M>reg HID0=0
M>reg MSR=0
M>
```

If these commands fail, just try again.

2 Now enter:

```
M>m -d4 -a4 0=11111111,22222222,33333333,44444444
M>m -d4 -a4 0..
00000000 11111111 02222222 33333333 44444444
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>
```

- If you do not see correct values written in memory, try increasing memory delay (page 201).
- If the read value is not equal to the written value, the memory controller may not be set up correctly.
- If the read value is equal to the written value, but you still suspect memory problems, the emulator firmware might not be working with cache.

Chapter 14: Troubleshooting the Emulation Module

Solving Target System Problems

3 Enter:

```
M>cf reset=rom
M>rst
M>m -d4 -a4 0..
```

- Read value not equal to the written value implies that reset is tied to memory controller. Check $\overline{\text{HRESET}}$ and $\overline{\text{SRESET}}$ for correct connections.

4 If you have memory problems running Windows NT, you may have this problem:

- System normally runs in little endian mode
- “rst” returns processor to big endian, memory controller on target still little endian, so memory access doesn't work.

5 Hand load a little program:

```
M>m -d4 -a4 100=38210001,60000000,60000000,4bffffff4
M>reg GPR1=0
M>
```

This means: Add 1, GPR1, NOP, NOP, JMP .-4

Set the PC to this program:

```
M>reg PC=100
M>
```

Step, then check the register:

```
M>s
PC=00000104
M>reg GPR1
reg GPR1=00000001
M>
```

This should return "reg GPR1=00000001".

Step some more and verify that GPR1 increments after every four steps:

```
M>s 4
PC=00000104
M>reg GPR1
reg GPR1=00000002
M>
```

If running from reset causes problems

Running from reset may cause some problems once background is entered. To ensure 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 'ORing' in 0x00000002 into the SRR1 register and resuming the run.

To test the target system

The following program can be placed into memory.

```
start:      addi      GPR1,1      - 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 GPR1=0
M>m -a2 -d2 10000=3821,1,6000,0,6000,0,4bff,fff4
M>r 10000
U>reg GPR1
    reg GPR1=00034567    # or some number
U>reg GPR1
    reg GPR1=00102333    # or some number
U>
```

This program will loop forever, incrementing GPR1. 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.

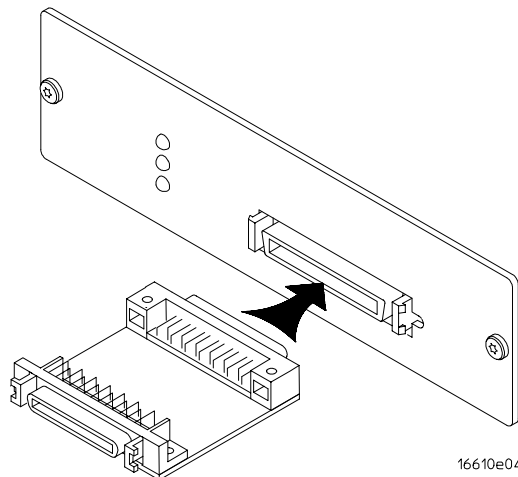
If there are many subnet masks in use on the local subnet, the logic analysis system may take a very long time to connect to the network after it is turned on.

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 pv” 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.07.54 22Apr98
Location: Generics
```

```
HPE3498A PowerPC 603 (Rev3)JTAG Emulator
Version: A.02.04 14Apr98
```

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 3: Boundary Scan Master Test

TEST 4: 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.

Exchange Assemblies

Part Number	Description
E5901A #060	Emulation module

Replacement Assemblies

Part number	Description
16700-61608	Expansion cable
E3494-61604	16-pin cable
E3496-61601	50-pin cable
E3496-66502	emulation module loopback test board
E3481-61601	20-pin cable
E3452-66502	Target interface module (PPC JTAG board)

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.

Cleaning the Instrument

Glossary

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 Migration By loading new firmware and connecting a

different TIM, your emulator migrates from support of one PowerPC model to support of another PowerPC model.

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.

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.

Emulator An emulation module or an emulation probe

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

Glossary

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.

Label Labels are used to group and identify logic analyzer channels. A label consists of a name and an associated bit or group of bits.

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.

Glossary

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.

Symbol Symbols represent patterns and ranges of values found on labeled sets of bits. Two kinds of symbols are available:

- 1) Object file symbols — Symbols from your source code, and symbols generated by your compiler. Object file symbols may represent global variables, functions, labels, and source line numbers.
- 2) User-defined symbols — Symbols you create.

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.

Symbols

- % prefix, 129
- * symbol in listing, 130, 149
- ? symbol in listing, 130, 149

A

- ADDR label, modifying, 102, 137
- address
 - triggering on a specific address, 107, 143
- addresses
 - branch target, 129, 149
 - mask, 114
 - offset, 171
 - PC label, 168
 - type, 103, 138
- aliases, PowerPC, 4
- analysis probe
 - connecting assembly and probe head, 50
 - connection illustration, 49
 - connection overview, 47
 - definition, 327
 - dimensions, 44
 - disconnecting from the target, 49
 - equipment required, 26
 - equipment supplied, 24
 - inverse assembly, 109, 144
 - microprocessors supported, 261
 - modes of analysis, 101
 - modes of operation, 99, 135
 - operating characteristics, 261
 - overview, 2
 - pod locations, 53
 - power on/power off sequence, 45
 - processors supported, 4
 - product numbers, 4
 - storage qualification, 107, 142
 - target system requirements, 43
 - to connect to a QFP target, 48

- analysis probe problems, 286
 - erratic trace measurements, 287
 - target system will not boot, 286
- analyzer problems, 283
 - capacitive loading, 287
 - intermittent data errors, 283
 - unwanted triggers, 284
- analyzing the processor, 97, 133
- assistant
 - See* setup assistant
- AT status bit, 103, 138

B

- b prefix, 148
- background debug monitor, 327
- background mode, 301
- BB status bit, 103, 138
- BDM port
 - See* debug port
- BGA socket, 43
- bits
 - labels, 102, 137
 - LSB and MSB, 102, 137
 - STAT, 103, 138
- BKG light, 302
- branch exception disassembly, 112
- branch instructions, 129, 149
- branches, displaying, 124
- breakpoints
 - tracing until, 257
- built-in commands, 301
 - configuration, 197
 - list of commands, 303
- built-in terminal interface, 301
- bus size, configuring, 203

C

- cables
 - emulator, 191
 - replacing, 297, 324

- cache
 - disabling, 126, 144
 - enabling, 126
 - trace problems and, 289
- cache, configuration, 195
- cache-on execution tracker
 - configuration, 101
 - overview, 101
- cache-on trace reconstruction, 109, 113
- caches
 - enabling and disabling, 126, 144, 220
- cards
 - See* logic analyzers
- CD-ROM, installing software from, 38
- cf commands, 197
- characteristics
 - analysis probe, 261
 - emulation module, 269
- checklist, setup, 21
- circuit board, dimensions, 44
- cleaning, 297, 325
- clock rate, 301
- clock speed
 - maximum, 261
- clocks
 - logic analyzer, 99, 135
 - qualification, 107, 142
 - qualified, and emulator, 248
 - slow, 256, 258, 294
- colors, 124
- comments, in GPA files, 279
- compilers, 163
- configuration
 - checklist, 21
 - logic analyzers, 78, 79, 102, 137
- configuration file floppy disk, 81
- configuration files
 - installing, 35
 - loading, 78, 79, 82
 - names of, 84

- configuration, emulation module
 - overview, 194
 - using debugger, 199
- connection
 - analysis probe, 41
 - analysis probe to logic analyzer, 51
 - analysis probe to target, 47
 - emulation module, 173, 174
 - high-density cables illustration, 53
 - high-density termination cables, 53
 - host workstation, 211
 - problems, LAN, 317
 - QFP target systems, 48
 - setup checklist, 21
- connectivity, 218
- connector
 - 10BASE2, 216
 - 10BASE-T, 216
 - debug port, 183
 - JTAG, 178
 - JTAG, levels, 308
- connector board, 327
- connectors
 - custom probing, 85
- custom probing
 - designing connectors, 85
- D**
- d prefix, 148
- DATA label, modifying, 102, 137
- data retry, 204
- data retry, configuring, 204
- data, displaying, 128
- debug mode, 327
- debug port
 - connecting to, 191
- debugger, 301
- debuggers
 - benefits of, 212
 - configuration, 199
 - Green Hills, 222
 - MRI, 230
 - SDS, 236
 - setting up, 215
 - writing, 305
- decoding
 - exception, 122
 - simplified mnemonic, 119
- delays, configuring, 201, 202
- development port
 - See debug port
- Diab Data
 - compiler, 164
- dimensions of analysis probe, 44
- direct probing, 87
- directories
 - configuration files, 80
 - software installation, 39
 - source code, 169
- disassembly, branch exception, 112
- display filtering, 124
- displaying on PC, 218, 219
- displaying over web, 218
- displaying state data, 128
- dmwrop, configuring, 206
- driver firmware error, 309
- DRTRY, 204
- E**
- elastomeric probe adapter
 - definition, 327
- electrical problem, 301
- Emulation Control Interface, 301
 - configuration, 196
 - debugger conflict, 215
 - introduction, 175
 - when to use, 242
- emulation module
 - built-in commands, 301
 - configuration, 199
 - connecting, 174, 190
 - definition, 327
 - description of, 3
 - HP 16600 installation, 188
 - HP 16700A installation, 186
 - port number, 217
 - product numbers, 4
 - target system design, 178
- emulation probe
 - definition, 327
- emulation solution
 - See solution
- enhanced inverse assembler
 - logic analyzer requirements, 29
- environmental characteristics
 - emulation module, 268
- equipment required
 - analysis probe, 26
 - emulation module, 31
- equipment supplied, 24
 - analysis probe, 24
 - emulation module, 30
 - ordering information, 4
 - overview, 4
- error messages
 - inverse assembler, 292
- even and odd pods, 53
- examples, measurement, 243
- exception decoding, 122
- exporting a display, 218, 219
- extended mnemonics, 152
- extender, 327
- F**
- files
 - loading vs. installing, 36
 - workstation setup, 215
- filtering, display, 124

firmware, 301
 emulation module, 193
 updating, 192, 193
 version, 193
flash ROM
 updating emulator, 192
flexible adapter
 definition, 327
floppy disk, installing software
 from, 81
floppy disks
 duplicating, 82
flowchart, setup, 21
format menu, 102, 137
full solution, 3
FUNCTIONS in GPA format, 277

G

General-Purpose ASCII format
 comments, 279
 FUNCTIONS, 277
 record format summary, 274
 SECTIONS, 276
 SOURCE LINES, 278
 START ADDRESS, 279
 VARIABLES, 277
general-purpose flexible adapter
 definition, 328
general-purpose probes, 87
Green Hills
 compiler, 165
 debugger, 222

H

high-density adapter cable
 definition, 328
high-density termination adapter
 definition, 328
host computer
 connecting to, 211
HRESET signal, 178

I

IEEE 802.3, 216
illegal opcode, 129
inwrop, configuring, 207
information sources, 33
init command, 309
installation, software, 35
instruction cache
 See cache
instruction decoding, 119, 152
intermodule measurement
 creating, 246
intermodule measurement
 problems, 291
 an event wasn't captured, 291
 analyzer doesn't stop, 248
inverse assembler
 configuration file names, 51, 52, 84
 definition, 328
 loading, 114
 loading files, 80, 82
 operating modes, 113
 output format, 129, 148
 overview, 101
 preferences, 114
 probing for "IA-only", 85
 requirements for, 109, 144
 requirements for enhanced, 29
inverse assembler floppy disk, 81
inverse assembler problems, 288
 failure to load or run, 290
 incorrect inverse assembly, 288
 no inverse assembly, 288
inverse assembly, 109, 144
 cache-on, 113
 displays, 243
 pods required, 27
 traditional, 109, 113
IP address, 216

J

J1-J6 connectors, 53
JTAG port
 connections, 183
jumper, definition, 328

K

keep-out area, 43

L

labels
 definition, 328
LAN
 emulation module, 216
 problems, 317
lights
 See status lights
listing
 incorrect, 288
Listing menu, 128, 146
listing windows, 243
Load menu, 114
loading configurations
 logic analyzer, 78, 79
loading configurations, vs.
 installing, 35
logic analyzers
 clocking, 107, 142
 configuration, 102, 137
 configuring, 80, 82
 HP 16550A 1-card 32-bit
 connections, 56
 HP 16550A 1-card no-data
 connections, 57
 HP 16550A 2-card 64-bit
 connections, 54
 HP 16554 1-card no-data
 connections, 61, 62, 75, 76
 HP 16554 2-card 64-bit
 connections, 58, 72
 HP 16554/55/56/57 three-card
 connections, 59, 73

- HP 16555 1-card no-data connections, 61, 62, 75, 76
 - HP 16555 2-card 64-bit connections, 58, 72
 - HP 16556 1-card no-data connections, 61, 62, 75, 76
 - HP 16556 2-card 64-bit connections, 58, 72
 - HP 16557 1-card no-data connections, 61, 62, 75, 76
 - HP 16557 2-card 64-bit connections, 58, 72
 - HP 1660 series 64-bit connections, 77
 - HP 16600A 32-bit data connections, 64
 - HP 16600A 64-bit data connections, 63
 - HP 16600A and HP 16700A-series, 23
 - HP 16600A no data connections, 65
 - HP 16601A 32-bit connections, 67
 - HP 16601A 64-bit connections, 66
 - HP 16601A no data connections, 68
 - HP 16602A 32-bit connections, 69
 - HP 16602A no data connections, 70
 - HP 16603A no data connections, 71
 - HP 1670 series 64-bit connections, 78
 - HP 16710 1-card 32-bit connections, 56
 - HP 16710 1-card no-data connections, 57
 - HP 16710 2-card 64-bit connections, 54
 - HP 16711 1-card 32-bit connections, 56
 - HP 16711 1-card no-data connections, 57
 - HP 16711 2-card 64-bit connections, 54
 - HP 16712 1-card 32-bit connections, 56
 - HP 16712 1-card no-data connections, 57
 - HP 16712 2-card 64-bit connections, 54
 - loading configuration files, 78, 79
 - software version requirements, 29
 - storage qualification, 107, 142
 - supported, 27
 - LSB, 102, 137, 146
- M**
- mainframe logic analyzer
 - definition, 328
 - male-to-male header
 - definition, 328
 - mask, subnet, 318
 - measurement examples, 243
 - memory
 - configuring delays, 201, 202
 - configuring parity, 202
 - configuring read, 205
 - configuring write, 206, 207
 - testing, 313
 - memory banks, 114
 - microprocessor aliases, 4
 - microprocessors supported, 4
 - Microtec Research Inc.
 - compiler, 165
 - debugger, 230
 - MMU support, 179, 181
 - mnemonics, 119, 152
- modes
 - analysis, 101
 - operating, 99, 135
- monitor, 199
- monitor, background debug, 327
- mrddop, configuring, 205
- MRI debugger, 230
- MSB, 102, 137, 146
- MULTI debugger, 222
- O**
- object module file symbols, 157
 - offset, address, 171
 - online configuration help, 23
 - operating characteristics
 - analysis probe, 261
 - emulation module, 268
 - operating modes, inverse
 - assembler, 113
 - Options menu, 114
- P**
- parity, configuring, 202
 - parity, support, 179
 - parts supplied, 24
 - passive probing, 85
 - path, source file, 169
 - PC (personal computer)
 - connecting to, 211
 - PC label, 168
 - performance verification test, 319
 - performance verification tests, 301
 - pod locations, 53
 - port number, emulation module,
 - 217, 303
 - power on/power off sequence, 45
 - PowerPC aliases, 4
 - ? prefix and suffix, 148
 - preprocessor
 - See* analysis probe
-

- problems
 - analysis probe, 281
 - emulation module, 299
- processor support package, 38
- processors supported, 4
- program flow analysis
 - Pods required, 27
- program symbols, 157
- prompts, 305
 - list of, 305
 - troubleshooting, 309
- prototype analyzer
 - definition, 329
- PV
 - See* performance verification
- Q**
- QACK pin, 178
- QFP target systems, 48
- question mark, 148
- R**
- R/W status bit, 103, 138
- real-time runs, 301
- real-time runs, configuring, 199
- record format, General-Purpose
 - ASCII, 274
- references, 33
- register commands, 306
- registers
 - listing format, 129, 148
- remote operation of logic analyzer,
 - 218
- repair
 - analysis probe, 296
 - emulation module, 323
- replacement parts, 324
- requirements
 - target system, 43, 178
- RESET
 - light, 302
- reset
 - configuring, 201
 - effects of, 315
 - troubleshooting, 310
- RESET signals, 179
- restrict to real-time runs, 301
- run control tool
 - See* emulation control interface
- S**
- SDS debugger, 236
- SECTIONS in GPA format, 276
- service, how to obtain, 296, 323
- setup
 - See* configuration
- Setup Assistant, 23
- setup assistant, 23
 - definition, 329
- setup checklist, 21
- show cycles
 - disassembly, 101
- signals
 - debug port, 183
 - logic analyzer, 262
- signals, expected levels, 308
- SingleStep debugger, 236
- skid, reducing, 247
- slow clock, 301
- slow clock message, 256, 258
- slow or missing clock, 301
- socket, BGA, 43
- software
 - installing, 35
 - list of installed, 39
 - requirements, 29
- software addresses, 168
- software analyzer, 26
- software probe
 - See* emulation module
 - See* emulation probe
- software requirements, 29
- software, emulation module
 - firmware, 193
- solution
 - at a glance, 2
 - definition, 329
- solutions
 - description of, 2
 - equipment required, 32
 - product numbers, 4
- source code, 155
 - displays, 243
- source correlation
 - data display, 131, 147
 - in analyzer, 155
 - using, 166
- source correlation tool set, 131
- source file search path, 169
- SOURCE LINES in GPA format,
 - 278
- specifications
 - See* characteristics
- SRESET signal, 178
- START ADDRESS in GPA format,
 - 279
- STAT
 - encoding, 103, 138
 - label, 103, 138
 - modifying, 102, 137
- state-per-ack, mode of operation,
 - 99, 135
- state-per-clock mode, 107
- state-per-transfer, mode of operation,
 - 99, 135
- status bits, 103, 138
- status encoding, 103, 138
- status lights, 302
- storage qualification, 107, 142
- STS status bit, 103, 138
- subnet mask, 318
- ? prefix and suffix, 148
- SW_ADDR label, 129

- symbols
 - definition, 329
 - in analyzer, 155
 - object file, 157
 - object module file, 157
 - predefined, 105, 140, 157
 - program, 157
 - user-defined, 158
- synchronous mode, 99, 135
- T**
- TA status bit, 103, 138
- target control port, 329
- target interface module (TIM)
 - connecting, 191
 - definition, 329
- target system
 - boot failure, 286
 - connecting to, 174
 - designing for HP E2477A
 - software, 86
 - power sequence, 45
 - problems with, 306
 - QFP connections, 48
 - requirements for analysis probe, 43
 - requirements for emulation, 178
- TEA status bit, 103, 138
- telnet, 217, 303
- terminal interface, 217, 301
 - See also* built-in commands
- tests, emulation module, 319
- 32bitmode, configuring, 203
- timing, mode of operation, 100, 136
- TLBISYNC pin, 203
- trace
 - erratic, 287
 - missing display, 285
- trace reconstruction, cache-on, 109
- transition board
 - definition, 330
- trigger
 - dialog, 107
 - emulation module, 244
 - on address, 107
 - on break, 251
 - source code, 170
 - specification, 107, 142
 - unwanted, 284
- Trigger menu, 107, 142
- troubleshooting, 301
 - analysis probe, 281
 - emulation module, 299
- TS status bit, 103, 138
- TSIZ status bit, 103, 138
- U**
- Undefined Opcode, 148
- unknown opcode, 129
- USER light, 302
- V**
- VARIABLES in GPA format, 277
- versions
 - emulation module firmware, 193
 - logic analyzer software, 29
- voltage
 - emulation module, 269
- W**
- web enabled logic analyzer, 218
- web sites
 - HP logic analyzers, 33
 - See also* under debugger names
- wizard
 - See* setup assistant
- X**
- X windows, 218, 219
- XRAY debugger, 230

© Copyright Hewlett-Packard Company 1994-1999
All Rights Reserved.

Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

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 or 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.

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 PowerPC 603 User's Guide*.

Publication number
E2455-97011, September, 1999
Printed in USA.

The information in this manual previously appeared in:
E2455-97010 February 1999
E2455-97009 December 1998
E2455-97008 July 1996
E2455-97007 December 1996
E3452-97001 August 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

Reflection 1 is a U.S. trademark of Walker, Richer & Quinn, Inc.

UNIX is a registered trademark of the Open Group.

Windows and MS Windows are U.S. registered trademarks of Microsoft Corp.

X/Open is a registered trademark, and the X device is a trademark of X/Open Company Ltd. in the UK and other countries.

PowerPC 603 and Embedded PowerPC microprocessors are products of Motorola, Inc.