User's Guide

Publication number E2476-97007
June 2000

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

Logic Analysis Support for the Motorola PowerPC MPC8XX

Logic Analysis Support for the Motorola Embedded PowerPC MPC8XX—At a Glance

The Agilent Technologies E9603A inverse assembler, in conjunction with an Agilent Technologies logic analyzer, allows you to view PowerPC MPC8XX assembly instructions that are executing in your target system.

The inverse assembler/excution tracker model number is Agilent Technologies 9584A Option 001 when ordered alone. You can also order an analysis probe and inverse assembler/execution tracker, which provides the hardware for easy connection to a target system. The model number for the analysis probe and inverse assembler is Agilent Technologies 9584A Option 002.

The inverse assembler is identified as “Agilent Technologies E2477A” in the Setup Assistant. The analysis probe and inverse assembler is identified as “Agilent Technologies E2476B” in the Setup Assistant.

If You Purchased an emulation solution

The E9484A Emulation Solution lets you use an Agilent Technologies 16600/16700-series logic analysis system to debug and characterize PowerPC MPC8XX target systems. The emulation solution is a bundled product consisting of an inverse assembler and analysis probe (or an inverse assembler only and custom probing designed into the target system), an emulation module (and its cables and adapters), and the Agilent Technologies B4620B source correlation tool set.

For more information on an emulation solution

The *Emulation for the PowerPC MPC8XX User’s Guide* describes setting up and using the emulation probe and emulation module.

Information about using the logic analysis system with the emulation probe/module can be found in Chapter 10, “Coordinating Logic Analysis with Processor Execution”, beginning on page 185 of this manual.

Additional Equipment Included in an Emulation Solution

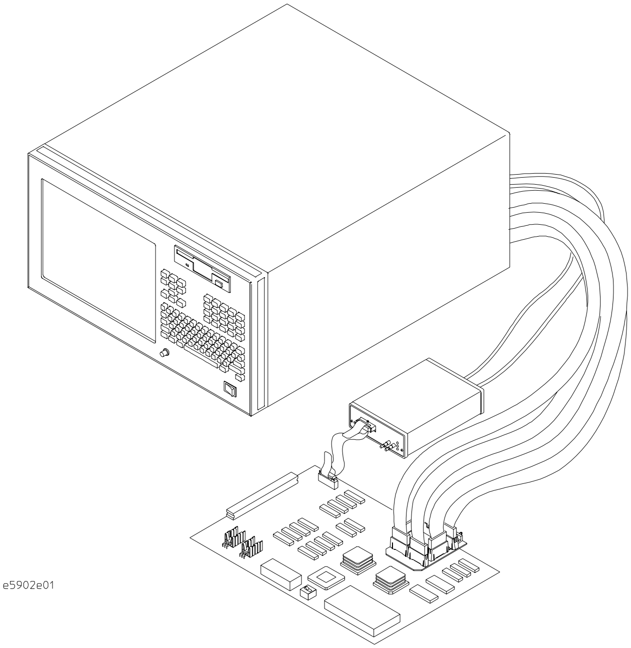
Emulation Module

The emulation module plugs into your Agilent Technologies 16600/700-series logic analyysis system frame, and the emulation probe connects to the emulation module and the JTAG port on your target system. The emulation module lets you use a microprocessor’s built-in debugging features including run control and access to registers and memory. A high-level source debugger can use the emulation probe/module to debug code running on the target system.

Source Correlation Tool Set

The Agilent Technologies B4620B Source Correlation Tool Set lets you set up logic analyzer triggers based on source code, and it lets you view the source code associated with signal values captured by the logic analyzer.

Emulation Solution



In This Book

This book documents the following products:

Processors supported	Product ordered	Includes
MPC860/855/821 Up to 50 MHz bus speed 357-pin BGA package	E9584A Option #002	E2476B BGA analysis probe and inverse assembler / execution tracker
MPC860/855/821 any package, custom probing	E9584A Option #001	E2477A inverse assembler / execution tracker

Related equipment

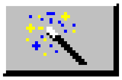
The following equipment is included in the PowerPC MPC8XX emulation solution.

Processors supported	Product ordered	Includes
MPC860/855/821 Up to 50 MHz bus speed 357-pin BGA package	E9484A Option #002	E2476B BGA analysis probe, inverse assembler / execution tracker, emulation probe, emulation module, B4620B Source Correlation Tool Set
MPC860/855/821 any package, custom probing	E9484A Option #001	E2477A inverse assembler / execution tracker, emulation probe, emulation module, B4620B Source Correlation Tool Set

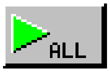
Tips To Save You Time

Use the Setup Assistant

Click here to connect the logic analyzer cables, and automatically load the correct configuration files. See page 20.

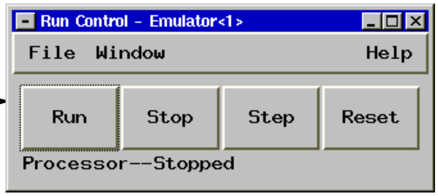


Use the appropriate Run button



Click here to start a measurement.

If your system includes an emulation probe/module, click here to run the target microprocessor.



Processor--Stopped

Additional Information Sources

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

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

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

http://www.agilent.com/find/logicanalyzer

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

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

Contents

1

Equipment and Requirements

This chapter describes:

- Setup Checklist
- Setup Assistant
- Equipment used with the analysis probe and inverse assembler
- List of compatible logic analyzers
- Emulation solution

Setup Checklist

Follow these steps to connect your equipment:

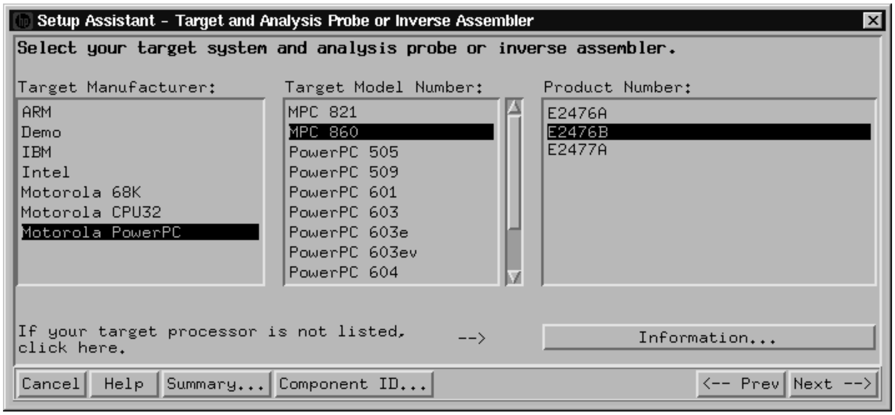
- Check that you received all of the necessary equipment. See page 24.
- If you need to install an emulation module in an Agilent Technologies 16600/700-series logic analysis system, see your emulation manual.
- Install the software. See page 59.
- Install the analysis probe, if ordered. See page 34. If you have an Agilent Technologies 16600/700-series logic analysis system, use the Setup Assistant to help you connect and configure your system. See page 22.

Setup Assistant

The Setup Assistant is the online tool for connecting and configuring your logic analysis system for microprocessor and bus analysis. The Setup Assistant is available on the Agilent Technologies 16600A and 16700-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 in the system window.



Setup Assistant - Target and Analysis Probe or Inverse Assembler

Select your target system and analysis probe or inverse assembler.

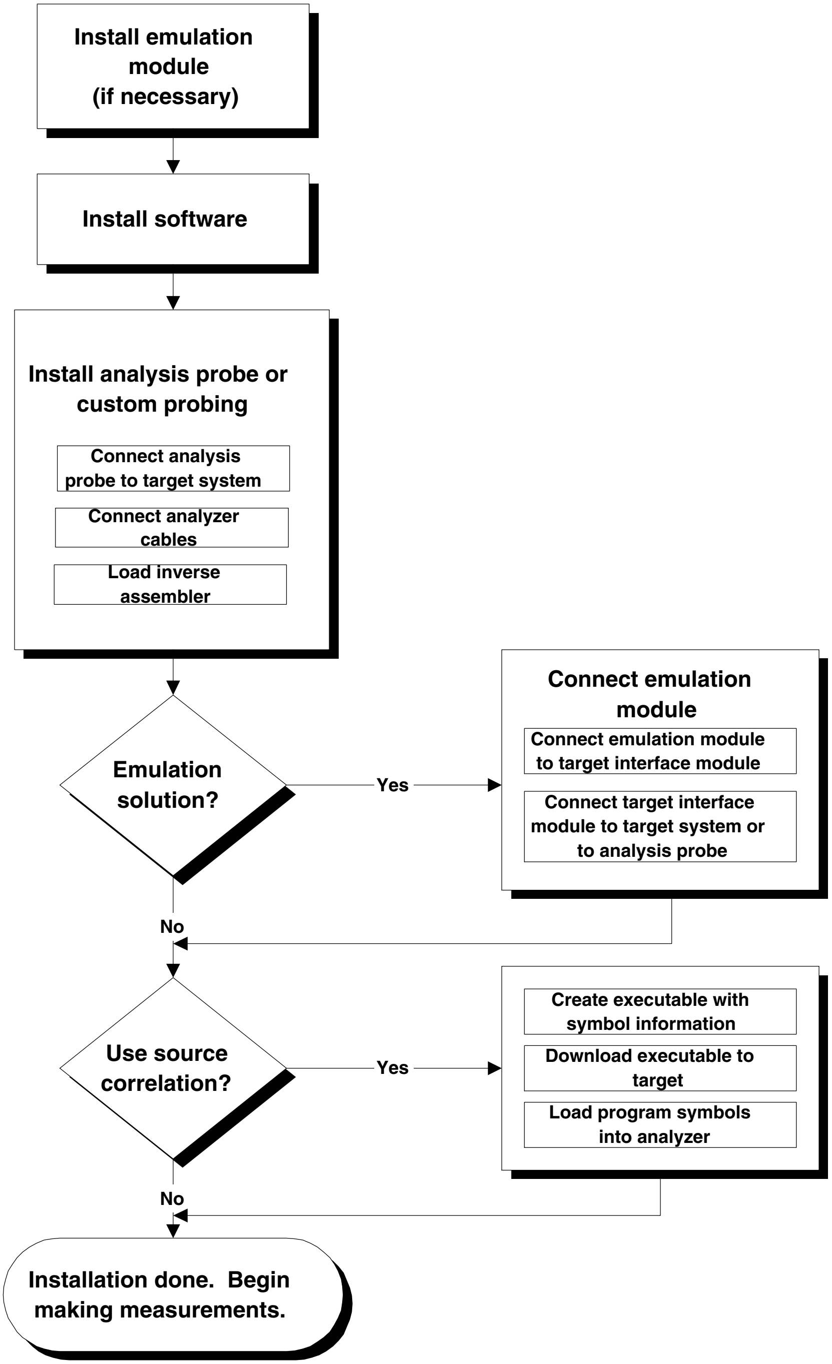
Target Manufacturer:
ARM
Demo
IBM
Intel
Motorola 68K
Motorola CPU32
Motorola PowerPC

Target Model Number:
MPC 821
MPC 860
PowerPC 505
PowerPC 509
PowerPC 601
PowerPC 603
PowerPC 603e
PowerPC 603ev
PowerPC 604

Product Number:
E2476A
E2476B
E2477A

If your target processor is not listed, click here. -->

If you ordered this product with your Agilent Technologies 16600/700-series logic analysis system, the logic analysis system has the latest software installed, including support for this product.


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

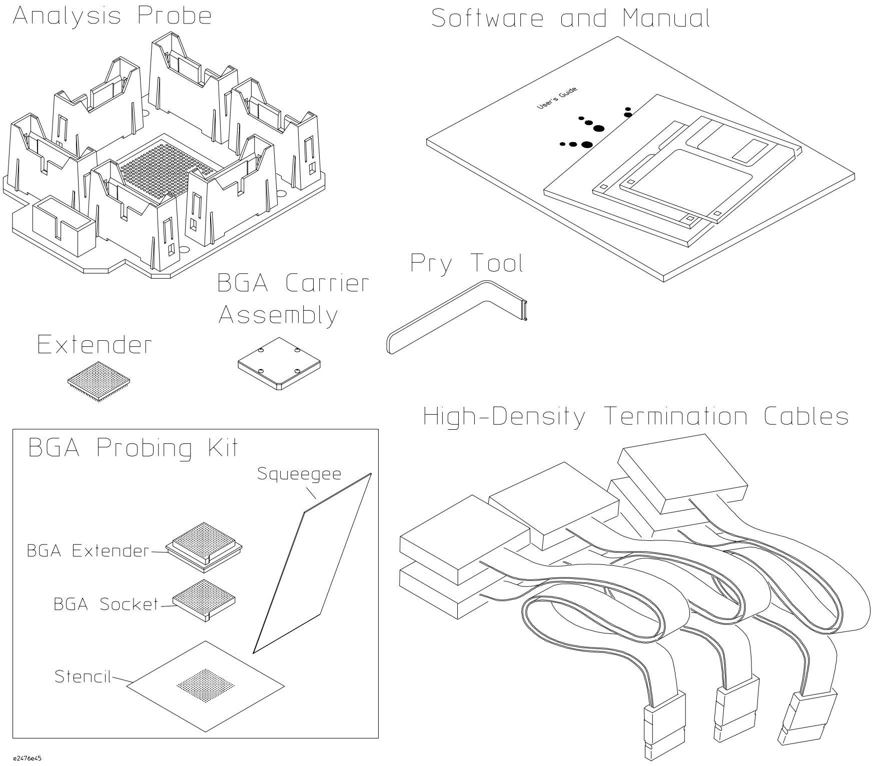
E2476F03.VSD

Equipment and Software Supplied

Listed below is the equipment and software supplied with:

- The Agilent Technologies E9584A Option 002 analysis probe and inverse assembler.
- The Agilent Technologies E9584A Option 001 inverse assembler.

Analysis Probe



Equipment Supplied with the Agilent Technologies E2476B Analysis Probe

The analysis probe (Agilent Technologies E2476B) includes:

- The Agilent Technologies E2476B analysis probe circuit board, which includes a pre-installed male-to-male header and a BGA carrier.
- An extender to protect target board and analysis probe sockets (one is pre-installed on the analysis probe socket; you can install another one on your target system).
- The Agilent Technologies E5355A BGA probing kit, which includes installation instructions.
- Three Agilent Technologies E5346A high-density termination cables.
- Logic analyzer configuration files, the inverse assembler with the cache-on trace reconstruction software on a CD ROM (for Agilent Technologies 16600/700-series logic analysis systems).
- Logic analyzer configuration files, the inverse assembler, and the cache-on execution tracker software on 3.5-inch disks (for other Agilent Technologies logic analyzers).
- This *User's Guide*.

Inverse Assembler (no Analysis Probe)

The inverse assembler (Agilent Technologies E9584A Option 001 when ordered separately) includes:

- Logic analyzer configuration files, the inverse assembler with the cache-on trace reconstruction software on a CD ROM (for Agilent Technologies 16600/700-series logic analysis systems).
- Logic analyzer configuration files, the inverse assembler, and the cache-on execution tracker software on 3.5-inch disks (for other Agilent Technologies logic analyzers).
- This *User's Guide*.

Additional equipment required

In addition to the items listed above, the following is required for state and timing analysis of an MPC8XX target system:

- A target system with an empty BGA socket, for the Agilent Technologies E5355A BGA Probing Kit. A BGA microprocessor is also required.
- One of the logic analyzers listed on page 27. The logic analyzer software version requirements are listed on page 28.
- For cache-on execution tracking, a supported logic analyzer with the required minimum software version as listed on page 28. The Agilent Technologies B4620B Source Correlation Tool Set is also highly recommended for correlating cache data with code execution.

If you are using the inverse assembler only (no analysis probe):

- Connector headers on your target system which supply the necessary signals to the logic analyzer. See Chapter 2, “Preparing the Target System,” beginning on page 31 for information on designing the appropriate connectors into the target system.
- Agilent Technologies termination adapter cables to attach your target system to a logic analyzer.

Additional equipment supported

Agilent Technologies B4620B Source Correlation Tool Set. The analysis probe and inverse assembler may be used with the Agilent Technologies B4620B Source Correlation Tool Set. The software is already installed on the Agilent Technologies 16600/16700-series logic analysis system’s disk. All you need is the entitlement certificate for licensing the source correlation tool set software. The CD-ROM is included in case you need to re-install the software.

Compatible Logic Analyzers

The following table lists the logic analyzers supported by the Agilent Technologies E2476B analysis probe and Agilent Technologies E2477A inverse assembler software. Logic analyzer software version requirements are shown on page 28.

The Agilent Technologies E2476B and E2477A require six logic analyzer pods (102 channels) for inverse assembly. The analysis probe contains six additional pods that you can monitor.

Logic Analyzers Supported

Agilent Technologies Logic Analyzer	Channel Count	State Speed	Timing Speed	Memory Depth
1660A/AS/C/CS/CP/E/ES/EP	136	100 MHz	250 MHz	4 k states
1661A/AS/C/CS/CP/E/ES/EP	102	100 MHz	250 MHz	4 k states
1670A	136	70 MHz	125 MHz	64 k or .5 M states
1670D	136	100 MHz	125 MHz	64 k or 1 M states
1671A	102	70 MHz	125 MHz	64 k or .5 M
1671D	102	100 MHz	125 MHz	64 k or 1 M
1670E	136	100 MHz	125 MHz	1 M
1671E	102	100 MHz	125 MHz	1 M
16550A (1 or 2 cards)	102/card	100 MHz	250 MHz	4 k states
16554A (2 or 3 cards)	68/card	70 MHz	125 MHz	512 k states
16555A (2 or 3 cards)	68/card	110 MHz	250 MHz	1 M states
16555D (2 or 3 cards)	68/card	110 MHz	250 MHz	2 M states
16556A (2 or 3 cards)	68/card	100 MHz	200 MHz	1 M states
16556D (2 or 3 cards)	68/card	100 MHz	200 MHz	2 M states
16557D (2 or 3 cards)	68/card	135 MHz	250 MHz	2 M states
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
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	333 MHz	333 MHz	2 M states
16718A (2 or 3 cards)	68/card	333 MHz	333 MHz	8 Mstates

Agilent Technologies Logic Analyzer	Channel Count	State Speed	Timing Speed	Memory Depth
16719A (2 or 3 cards)	68/card	333 MHz	333 MHz	32 M states
16750A (2 or 3 cards)	68/card	400 MHz	400 MHz	4/8 M states
16751A (2 or 3 cards)	68/card	400 MHz	400 MHz	16/32 M states
16752A (2 or 3 cards)	68/card	400 MHz	400 MHz	32/64 M states

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 E2476B/77A. You can obtain the latest software at the following web site:

http://www.agilent.com/find/logicanalyzer

If your software version is older than those listed, load new system software with the higher version numbers before loading the Agilent Technologies E2476B/77A software. See “Installing Software” on page 59 for instructions for loading software.

To use cache-on trace reconstruction or the cache-on execution tracker, you must have a 16600/700-series logic analysis system.

Logic Analyzer Software Version Requirements

Agilent Technologies Logic Analyzer	Minimum Logic Analyzer Software Version for use with E2476B/77A
16600A-series	The latest 16600A logic analyzer software version is on the CD-ROM shipped with this product.
1660-series and 1670-series	Software version A.02.01
Mainframes*	
16700-series	The latest 16700 logic analyzer software version is on the CD-ROM shipped with this product.
16500C Mainframe	Software version A.01.05
16500B Mainframe	Software version A.03.14

* The mainframes are used with logic analyzer modules such as the 16557D logic analyzer.

Emulation Solution

If you ordered an emulation solution, you received an emulation probe, and emulation module and accessories, which are described in the *Emulation for the MPC8XX User's Guide*.

The combination of an inverse assembler, an emulation module, and an Agilent Technologies 16600- or 16700-series logic analysis system lets you both view MPC8XX assembly instructions that are executing on your target system and use the target processor's built-in JTAG debugging features.

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. You can use the Agilent Technologies B4620B Source Correlation Tool Set to analyze high-level source using the logic analysis system.

2

Preparing the Target System

There are two ways to probe an MPC8XX target system:

- Using an analysis probe.
- Using logic analyzer connectors that have been designed into the target system.

This chapter describes:

- Target system design requirements including keep-out area and clearance to allow the analysis probe to be attached.
- Target system design considerations for logic analysis, which are the same whether you're using an analysis probe or designing connectors into your target system.
- How to attach the analysis probe to your target system.
- Design considerations for including logic analyzer connectors in your target system (when the analysis probe will not be used).

Preparing for Logic Analysis (and Inverse Assembly)

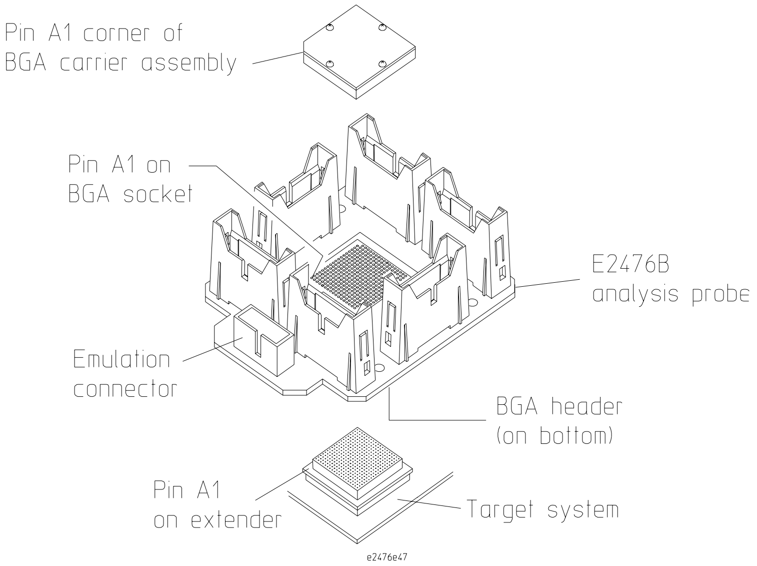
The MPC8XX inverse assembler requires a minimum of 102 logic analysis channels (six pods).

If optional signals are used, such as PCI analysis, additional logic analyzer pods are required.

You can either use the analysis probe or design high-density connectors into your target system for logic analyzer probe pods.

Connecting the Analysis Probe to the Target System

This section explains how to connect the Agilent Technologies E2476B analysis probe to the target system. If you are designing logic analyzer connectors into your target system and won't be using an analysis probe, skip this section and refer to "Designing Logic Analyzer Connectors into Your Target System" on page 43.



Connecting the 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 Agilent Technologies 16600/700-series logic analysis system. If you are using a 16600/700-series logic analysis system you will need to keep it powered-up so you can use the Setup Assistant to guide you through connecting and configuring your system.
- Assemble the microprocessor into the BGA carrier.
- Install the Agilent Technologies E5355A BGA probing kit on the target system.
- Test the target system with the BGA carrier assembly, without the analysis probe, then turn off the power again.
- Disconnect the BGA carrier assembly from the target system.
- Install the analysis probe onto the target system, and then install the BGA carrier assembly onto the analysis probe.

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

Protect Your Equipment

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

Analysis Probe Considerations

Some things to consider when using the analysis probe are:

- The component keep-out area on the target system.
- The clearance above the target system.
- The analysis probe’s dimensions.

Keep-out area on the target board

A 29.3 mm by 29.3 mm keep-out area is required on the target board where the analysis probe contacts the BGA socket.

Clearance above the target board

The components on the target board must not interfere with installation of the analysis probe. The maximum height of components under the analysis probe can not exceed 14 mm. See the diagram on the next page for the dimensions of the analysis probe.

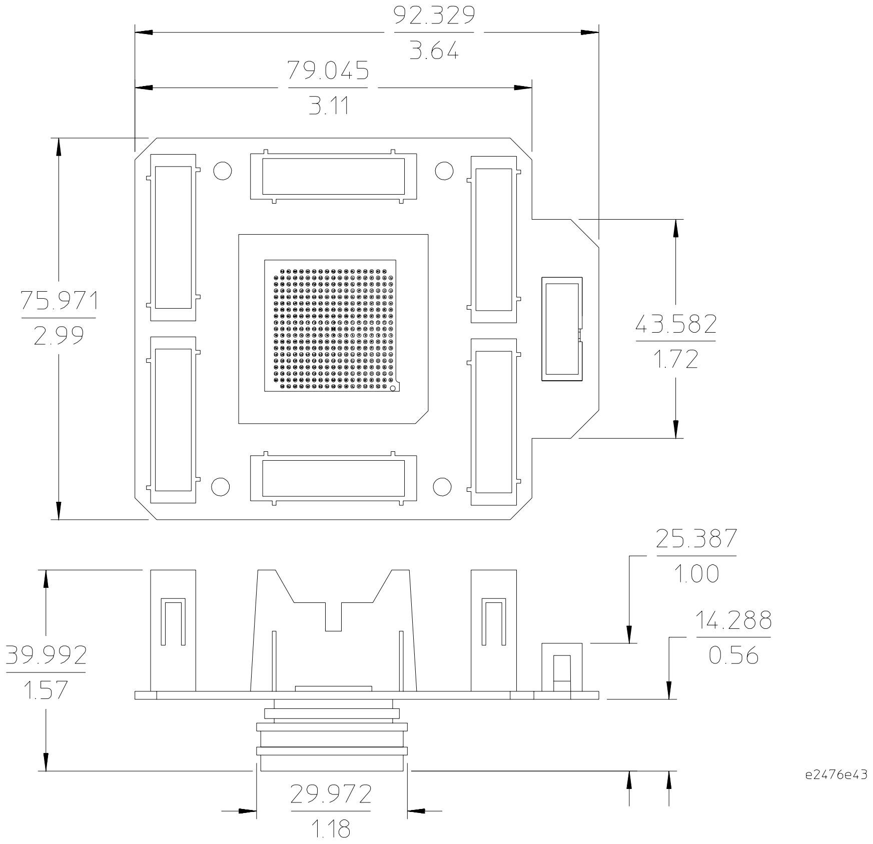
Remember to allow space above the analysis probe for the cables which plug into the top of the analysis probe.

See Also

The *Emulation and Analysis Solutions for Motorola MPC 8XX Microprocessors* data sheet has more detailed information and diagrams regarding the keep-out area and analysis probe dimensions. This data sheet is available from your Agilent Technologies representative.

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. Use this diagram to design your target system so that its components do not interfere with analysis probe installation.



Agilent Technologies E2476B Analysis Probe Circuit Board Dimension Diagram

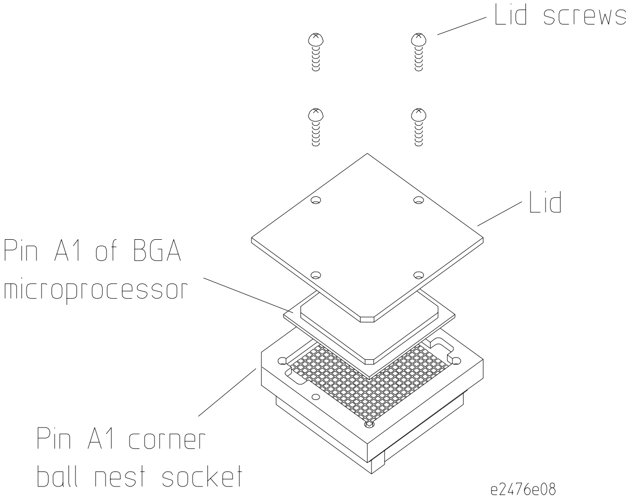
To assemble the microprocessor into the BGA carrier

The Agilent Technologies E2476B analysis probe has a BGA carrier for a 357-pin BGA microprocessor. Use the procedure below to install the BGA microprocessor into the BGA carrier.

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

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

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



Pin A1 Orientation of BGA Microprocessor and BGA Carrier

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

To install the Agilent Technologies E5355A BGA probing kit on the target system

The Agilent Technologies E5355A BGA probing kit requires a target system with an empty 357-pin BGA pad array. Connect the BGA probing kit as follows.

1 Ensure that your target system has a 357-pin BGA pad array with proper connections for your target microprocessor. This BGA pad array must be clean, unused, and have no solder on its pads.

2 Ensure that pin A1 of the BGA socket is properly aligned with pin A1 on the BGA pad array.

3 Install the socket onto the 357-pin BGA pad array, and solder it in place. Follow the soldering instructions in the process sheet that came with the Agilent Technologies E5355A BGA probing kit.

4 Install the extender into the socket. The extender protects the socket and your target board.

CAUTION:

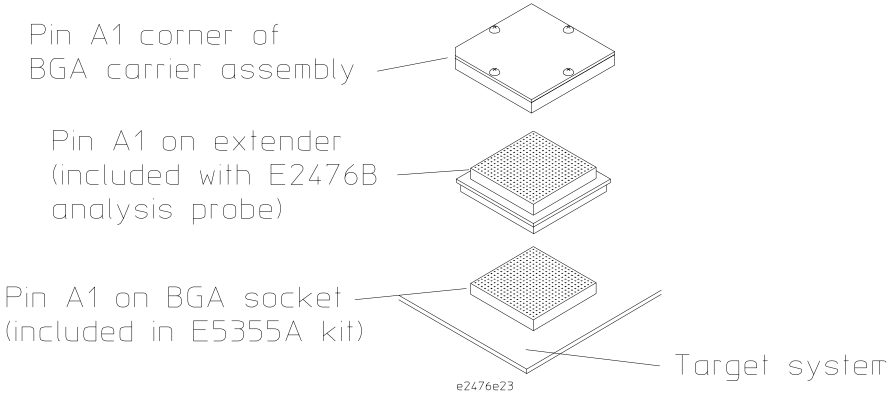
Target System Damage.
Once the extender is installed, do not remove it from the socket. The socket is held in place on your target system by solder between the socket pins and the BGA pads. If you remove the socket, one or more of the soldered pads may damage connections, traces, and BGA pads of your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

To test the target system with the BGA carrier assembly

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

1 Install the BGA carrier assembly into the extender.

2 Install the extender into the BGA socket on your target system.


Pin A1 corner of BGA carrier assembly
Pin A1 on extender (included with E2476B analysis probe)
Pin A1 on BGA socket (included in E5355A kit)
Target system
e2476e23

3 Turn on your target system and check operation.

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

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

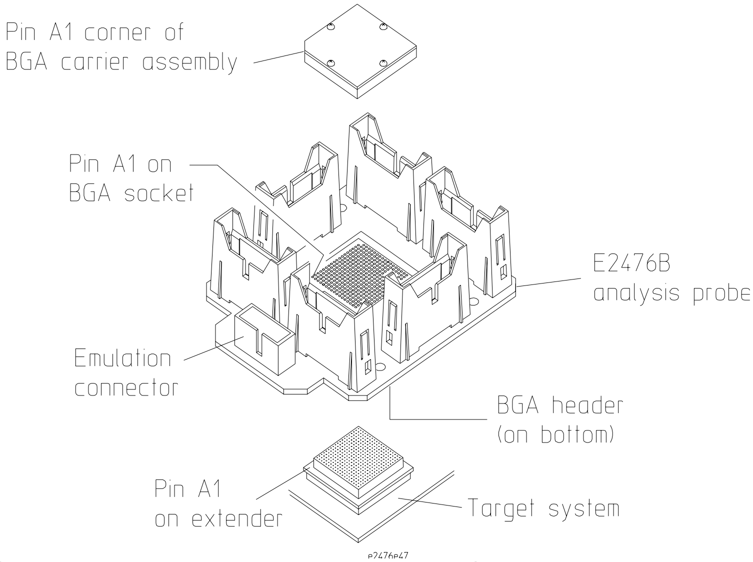
To install the analysis probe between the BGA carrier assembly and the target system

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

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

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

If the analysis probe interferes with components of the target system, or if a higher profile is required, additional BGA extenders can be used. BGA extenders can be ordered from Agilent Technologies using the Agilent part numbers listed in the Replaceable Parts table on page 183.



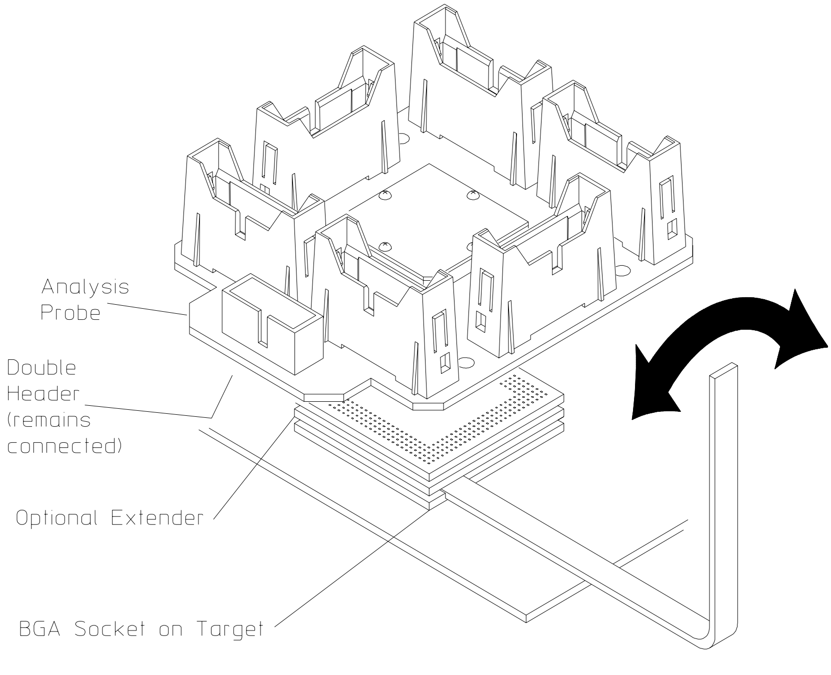
Connecting the Analysis Probe to the Target System

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

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

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

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



e2476e68

Designing Logic Analyzer Connectors into Your Target System

The logic analyzer can be connected directly to connectors on your target system. This section describes what kind of connectors to use, and how to connect the correct signals to the connectors.

If you are using an analysis probe, you do not need to include connectors on your target system.

Using High-Density Connectors

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

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

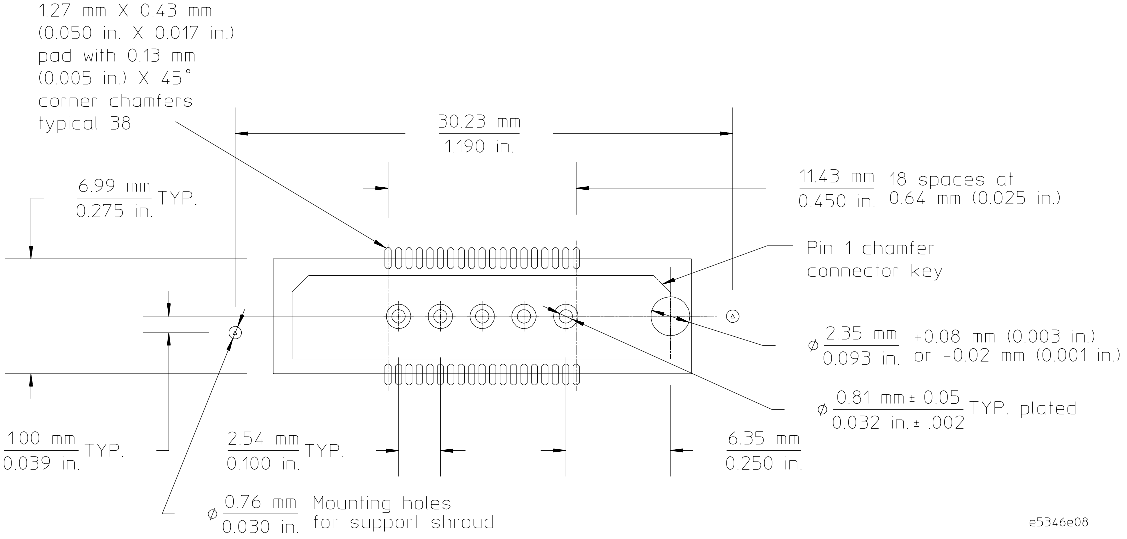
See Also

More information on this connector is available in the document *Passively Probing a Motorola MPC 860/821 BGA Target System with Agilent E5346A High-Density Termination Adapters*. This document is available in Portable Document Format (PDF) from the web site:

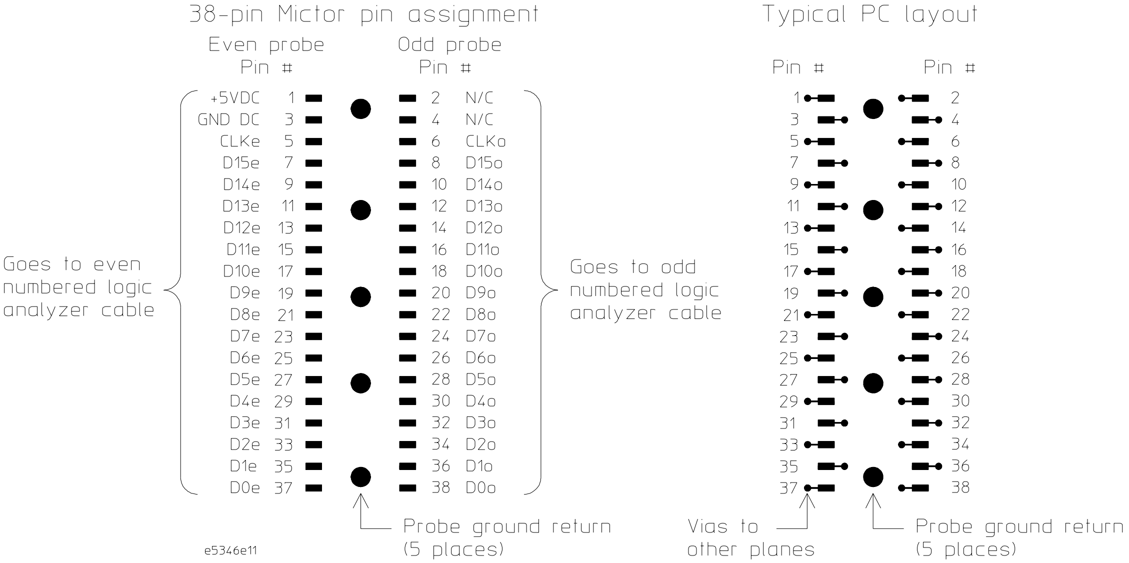
http://www.tm.agilent.com/tmo/datasheets/English/E5346A.html

High-Density Connector Mechanical Specifications

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



The high-density connector pin assignment and recommended circuit board routing are shown in the following figure.



38-pin Mictor pin assignment
Even probe
Odd probe
Goes to even numbered logic analyzer cable
Goes to odd numbered logic analyzer cable
Probe ground return (5 places)
e5346e11
Typical PC layout
Pin #
Pin #
Vias to other planes
Probe ground return (5 places)

Even probe		Odd probe	
	Pin #	Pin #	
+5VDC	1	2	N/C
GND DC	3	4	N/C
CLKe	5	6	CLKo
D15e	7	8	D15o
D14e	9	10	D14o
D13e	11	12	D13o
D12e	13	14	D12o
D11e	15	16	D11o
D10e	17	18	D10o
D9e	19	20	D9o
D8e	21	22	D8o
D7e	23	24	D7o
D6e	25	26	D6o
D5e	27	28	D5o
D4e	29	30	D4o
D3e	31	32	D3o
D2e	33	34	D2o
D1e	35	36	D1o
D0e	37	38	D0o

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

Recommended Connector Layout and Signal Routing

The advantages of the recommended configuration are:

- It is optimized for minimum trace lengths and electrical loading.

The disadvantages are:

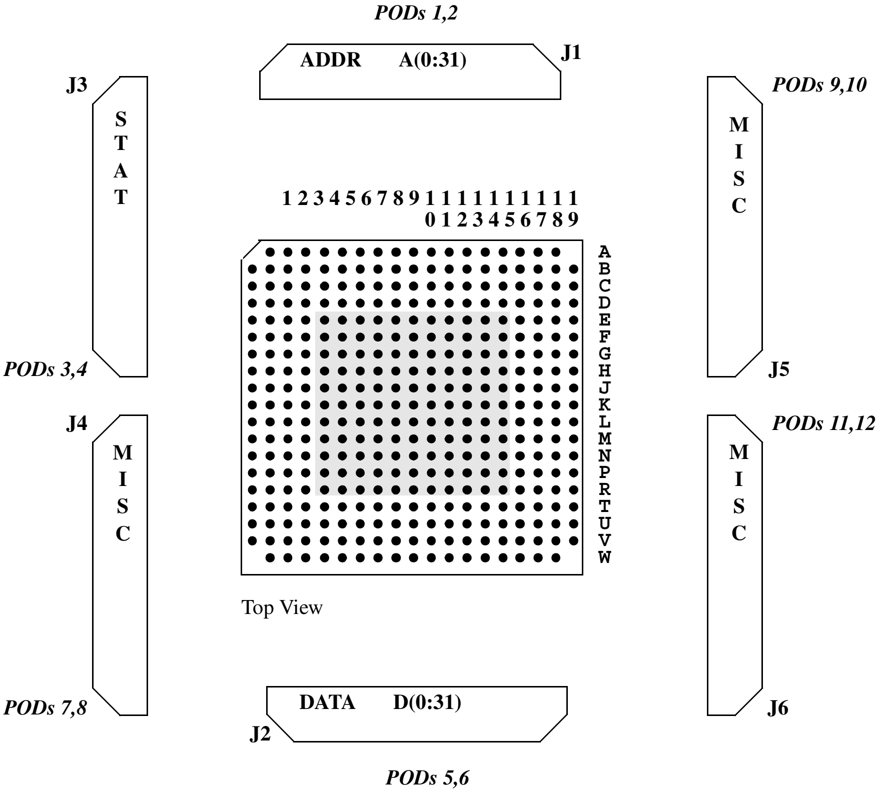
- It requires four high-density connectors for main memory disassembly in an 8-pod logic analyzer. This is required since pods 7 and 8 are split between two MICTOR connectors to minimize trace lengths.

Recommended Configuration Connection Notes

- 'nc' pins MUST be a true no-connect on the target. The signals are used for other functions unavailable to target probing.
- Five center inline pins on the connector are the signal ground returns and must be connected to ground.
- Any blank pins can be used for user defined signals.
- '#' or $\overline{\text{overscore}}$ denotes an active low signal.
- J1, J3: Required for inverse assembly with no data.
- J2: Required for inverse assembly with 32-bit data.
- J4-J6: Miscellaneous signals

Recommended Connector Layout

The following MICTOR placement is recommended to minimize trace lengths from the BGA to the connectors. Due to the high bus speeds, even small trace lengths can affect signal integrity.



Recommended Signal Routing

The tables on the following pages show MICTOR connector signal assignments for MPC8XX processor target systems. Observe recommended signal routing to minimize trace lengths and provide best signal fidelity.

MICTOR Connector J1					
J1 Pin	**BGA pin**	**MPC8XX Signal**	**J1 Pin**	**BGA Pin**	**MPC8XX Signal**
5	W3	CLKOUT	6	L1	$\overline{\text{STS}}$
7	B19	A0 (MSB)	8	C13	A16
9	B18	A1	10	B13	A17
11	A18	A2	12	D9	A18
13	C16	A3	14	D11	A19
15	B17	A4	16	C12	A20
17	A17	A5	18	B12	A21
19	B16	A6	20	B10	A22
21	A16	A7	22	B11	A23
23	D15	A8	24	C11	A24
25	C15	A9	26	D10	A25
27	B15	A10	28	C10	A26
29	A15	A11	30	A13	A27
31	C14	A12	32	A10	A28
33	B14	A13	34	A12	A29
35	A14	A14	36	A11	A30
37	D12	A15	38	A9	A31
Even Cable			**Odd Cable**		
Logic Analyzer Pod 2			**Logic Analyzer Pod 1**		

<table>
<tr><th colspan="6">MICTOR Connector J2</th></tr>
<tr><th>J2 Pin</th><th>BGA pin</th><th>MPC8XX Signal</th><th>J2 Pin</th><th>BGA Pin</th><th>MPC8XX Signal</th></tr>
<tr><td>5</td><td>B9</td><td>TSIZ0/$\overline{\text{REG}}$</td><td>6</td><td>C9</td><td>$\overline{\text{TSIZ1}}$</td></tr>
<tr><td>7</td><td>W14</td><td>D0 (MSB)</td><td>8</td><td>U10</td><td>D16</td></tr>
<tr><td>9</td><td>W12</td><td>D1</td><td>10</td><td>T12</td><td>D17</td></tr>
<tr><td>11</td><td>W11</td><td>D2</td><td>12</td><td>V9</td><td>D18</td></tr>
<tr><td>13</td><td>W10</td><td>D3</td><td>14</td><td>U9</td><td>D19</td></tr>
<tr><td>15</td><td>W13</td><td>D4</td><td>16</td><td>V8</td><td>D20</td></tr>
<tr><td>17</td><td>W9</td><td>D5</td><td>18</td><td>U8</td><td>D21</td></tr>
<tr><td>19</td><td>W7</td><td>D6</td><td>20</td><td>T9</td><td>D22</td></tr>
<tr><td>21</td><td>W6</td><td>D7</td><td>22</td><td>U12</td><td>D23</td></tr>
<tr><td>23</td><td>U13</td><td>D8</td><td>24</td><td>V7</td><td>D24</td></tr>
<tr><td>25</td><td>T11</td><td>D9</td><td>26</td><td>T8</td><td>D25</td></tr>
<tr><td>27</td><td>V11</td><td>D10</td><td>28</td><td>U7</td><td>D26</td></tr>
<tr><td>29</td><td>U11</td><td>D11</td><td>30</td><td>V12</td><td>D27</td></tr>
<tr><td>31</td><td>T13</td><td>D12</td><td>32</td><td>V6</td><td>D28</td></tr>
<tr><td>33</td><td>V13</td><td>D13</td><td>34</td><td>W5</td><td>D29</td></tr>
<tr><td>35</td><td>V10</td><td>D14</td><td>36</td><td>U6</td><td>D30</td></tr>
<tr><td>37</td><td>T10</td><td>D15</td><td>38</td><td>T7</td><td>D31 (LSB)</td></tr>
<tr><td colspan="3"><b>Even Cable</b></td><td colspan="3"><b>Odd Cable</b></td></tr>
<tr><td colspan="3"><b>Logic Analyzer Pod 4</b></td><td colspan="3"><b>Logic Analyzer Pod 3</b></td></tr>
</table>

MICTOR Connector J3					
J3 Pin	**BGA pin**	**MPC8XX Signal**	**J3 Pin**	**BGA Pin**	**MPC8XX Signal**
5	C2	LTA	6	D1	$\overline{\text{TEA}}$
7	H2	VFLS0/IP_B0/IWP0	8	C3	$\overline{\text{CS0}}$
9	J3	VFLS1/IP_B1/IWP1	10	A2	$\overline{\text{CS1}}$
11	J2	IPP_B2/$\overline{\text{I0IS16\_B}}$/AT2	12	D4	$\overline{\text{CS2}}$
13	G1	IPP_B3/IWP2/VF2	14	E4	$\overline{\text{CS3}}$
15	G2	IP_B4/LWP0/VF0	16	A4	$\overline{\text{CS4}}$
17	J4	IP_B5/LWP1/VF1	18	B4	$\overline{\text{CS5}}$
19	K3	IP_B6/DSDI/AT0	20	D5	$\overline{\text{CS6/CE1\_B}}$
21	H1	IP_B7//PTR/AT3	22	C4	$\overline{\text{CS7/CE2\_B}}$
23	E2	$\overline{\text{BG}}$	24	D8	$\overline{\text{BS\_A0}}$
25	E1	$\overline{\text{BB}}$	26	C8	$\overline{\text{BS\_A1}}$
27	G4	$\overline{\text{BR}}$	28	A7	$\overline{\text{BS\_A2}}$
29	E3	$\overline{\text{BI}}$	30	B8	$\overline{\text{BS\_A3}}$
31	D2	$\overline{\text{BDIP/GPLB5}}$	32	C7	$\overline{\text{WE0/BS\_B0/IORD}}$
33	F1	$\overline{\text{BURST}}$	34	A6	$\overline{\text{WE1/BS\_B1/IOWR}}$
35	B2	RD/$\overline{\text{WR}}$	36	B6	$\overline{\text{WE2/BS\_B2/PCOE}}$
37	F3	$\overline{\text{TS}}$	38	A5	$\overline{\text{WE3/BS\_B3/PCWE}}$
Even Cable			**Odd Cable**		
Logic Analyzer Pod 6			**Logic Analyzer Pod 5**		

MICTOR Connector J4					
J4Pin	**BGA pin**	**MPC8XX Signal**	**J4 Pin**	**BGA Pin**	**MPC8XX Signal**
5	N2	EXTCLK	6	K2	ALEA
7	-	-	8	H3	$\overline{\text{RSV/IRQ2}}$
9	J1	ALEB/DSCK/AT1	10	F2	$\overline{\text{CR/IRQ3}}$
11	N3	TEXP	12	V3	DP0/$\overline{\text{IRQ3}}$
13	L4	OP0	14	V5	DP1/$\overline{\text{IRQ4}}$
15	L2	OP1	16	W4	DP2/$\overline{\text{IRQ5}}$
17	L1	OP2/M0DCK1/$\overline{\text{STS}}$	18	V4	DP3/$\overline{\text{IRQ6}}$
19	M4	OP3/M0DCK2/DSD0	20	G3	FRZ/$\overline{\text{IRQ6}}$
21	R2	$\overline{\text{PORESET}}$	22	K1	$\overline{\text{KR}}$/$\overline{\text{IRQ4}}$/SPKROUT
23	P3	$\overline{\text{RSTCONF}}$	24	T5	IP_A0
25	N4	$\overline{\text{HRESET}}$	26	T4	IP_A1
27	P2	$\overline{\text{SRESET}}$	28	U3	IP_A2/$\overline{\text{IOIS16\_A}}$
29	R4	$\overline{\text{WAIT\_B}}$	30	W2	IP_A3
31	R3	$\overline{\text{WAIT\_A}}$	32	U4	IP_A4
33	C1	$\overline{\text{UPWAITA}}$/$\overline{\text{GPLA4}}$	34	U5	IP_A5
35	B1	$\overline{\text{UPPWAITB}}$/$\overline{\text{GPLB4}}$	36	T6	IP_A6
37	B3	$\overline{\text{CE1\_A}}$	38	T3	IP_A7
Even Cable			**Odd Cable**		
Logic Analyzer Pod 8			**Logic Analyzer Pod 7**		

MICTOR Connector J5					
J5 Pin	**BGA pin**	**MPC8XX Signal**	**J5 Pin**	**BGA Pin**	**MPC8XX Signal**
5	D7	GPLA0/GPLB00	6	A3	CE2_A
7	D3	GPLA5	8	U19	PA0/CLK8/TOUT4/L1TCLKB
9	C5	GPLA3/GPLB3/CS3	10	T19	PA1/CLK7/TIN4/BRGOUT4
11	B5	GPLA2/GPLB2/CS2	12	R18	PA2/CLK6/TOUT3/L1RCLKB/BRGCLK2
13	C6	OE/GPLA1/GPLB1	14	P17	PA3/CLK5/TIN3/BRGOUT3
15	T17	PC4/L1RSYNCA/CD4	16	P19	PA4/CLK4/TOUT2
17	T18	PC5/L1TSYNCA/SDACK1/CTS4	18	N18	PA5/CLK3/TIN2/L1TCLKA/BRGOUT2
19	R19	PC6/L1RSYNCB/CD3	20	M17	PA6/CLK2/TOUT1/BRGCLK1
21	M16	PC7/L1TSYNCB/SDACK2/CTS3	22	M19	PA7/CLK1/TIN1/L1RCLKA/BRGOYT1
23	M18	PC8/CD2/TGATE2	24	L17	PA8/L1RXDA
25	L18	PC9/CTS2	26	K18	PA9/L1TXDA
27	K19	PC10/CD1/TGATE1	28	J17	PA10/L1RXDB
29	J19	PC11/CTS1	30	G16	PA11/L1TXDB
31	F18	PC12//L1RQA/LIST4	32	F17	PA12/TXD2
33	E18	PC13/L1RQB/LIST3	34	E17	PA13/RXD2
35	D18	PC14/DREQ2/RTS2/LIST2	36	D17	PA14/TXD1
37	D16	PC15/DREQ1/RTS1/LIST1	38	C18	PA15/RXD1
Even Cable			**Odd Cable**		
Logic Analyzer Pod 10			**Logic Analyzer Pod 9**		

MICTOR Connector J6					
J6 Pin	**BGA pin**	**MPC8XX Signal**	**J6 Pin**	**BGA Pin**	**MPC8XX Signal**
5	U14	IRQ1	6	W15	IRQ7
7	V14	IRQ0	8	N16	PB16/L1RQA/LIST4
9	W16	PD3/SHIFT_CLK/$\overline{\mathrm{RRJECT4}}$	10	P18	PB17/L1RQB/LIST3
11	U16	PD4/LOAD_HSYNC/ $\overline{\mathrm{RRJECT3}}$	12	N17	PPB18/$\overline{\mathrm{RST2}}$/LIST2
13	U15	PD5/FRAME_VSYNC/ $\overline{\mathrm{RRJECT2}}$	14	N19	PB19/$\overline{\mathrm{RST1}}$/LIST1
15	V16	PD6/LCD_AC/LOE/$\overline{\mathrm{RTS4}}$	16	L16	PB20/SMRXD2/L1CLKOA
17	T15	PD7/LD0/$\overline{\mathrm{RTS4}}$	18	K16	PB21/SMTXD2/L1CLKOB
19	W17	PD8/LD1/TXD4	20	L19	PB22/$\overline{\mathrm{SMSYN2}}$/$\overline{\mathrm{SDACK2}}$
21	V17	PD9/LD2/RXD4	22	K17	PB23/$\overline{\mathrm{SMSYN1}}$/$\overline{\mathrm{SDACK1}}$
23	W18	PD10/LD3/TXD3	24	J18	PB24/SMRXD1
25	T16	PD11/LD4/RXDD3	26	J16	PB25/SMTXD1
27	R16	PD12/LD5/L1RSYNCB	28	F19	PB26/I2CSCL/BRGOUT2
29	V18	PD13/LD6/L1TSYNCB	30	E19	PPB27/I2CSDA/BRGOUT1
31	V19	PD14/LD7/L1RSYNCA	32	D19	PB28/SPIMISO/BRGOUT4
33	U17	PD15/LD8/L1TSYNCA	34	E16	PB29/SPIMOSI
35	U18	PB14/$\overline{\mathrm{RSTRT1}}$	36	C19	PB30/SPICLK
37	R17	PB15/BRGOUT3	38	C17	PB31/$\overline{\mathrm{SPISEL}}$/$\overline{\mathrm{RRJECT1}}$
Even Cable			**Odd Cable**		
Logic Analyzer Pod 12			**Logic Analyzer Pod 11**		

Designing a Debug Port Connector into Your Target System

You can use the analysis probe to make the debug port connection to your target system, or connect an emulation probe/module directly to a debug port on the target system. When using the MPC8XX emulation probe/module without an analysis probe, you need to consider how the emulation probe/module connects to the target system. Refer to the *Emulation for the PowerPC MPC8XX User's Guide* for information on designing a debug port.

3

Setting Up the Logic Analysis System

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 16600 and 16700-series logic analysis systems

Ensure the target system is powered off.

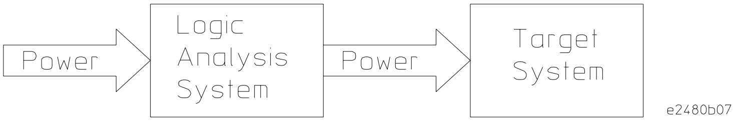
1. Turn on the logic analyzer. The Setup Assistant will guide you through the process of connecting and configuring the analysis probe.
2. When the analysis probe is connected to the target system and logic analyzer, and everything is configured, turn on your target system.

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.

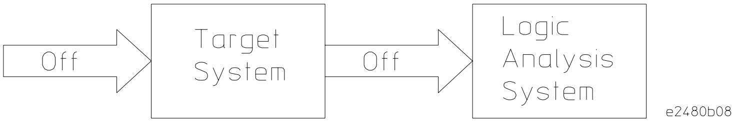


To power off

Turn off power to your system in the following order:

1 Turn off your target system.

2 Turn off your logic analysis system.



Installing Logic Analyzer Modules

You should install logic analyzer, oscilloscope, or pattern generator modules in your logic analysis system before you install an emulation module and software.

CAUTION: Electrostatic discharge (ESD) can damage electronic components. Use appropriate ESD equipment (grounded wrist strap, etc.) and ESD-safe procedures when you handle and install modules.

Refer to the Agilent Technologies 16600/16700-series logic analysis system's *Installation Guide* for instructions on installing modules.

Installing an emulation module

If you ordered an emulation module as part of your logic analysis system, it is already installed in the mainframe.

If you ordered an emulation module separtely, use the information provided in the *Emulation for the PowerPC MPC8XX User's Guide* to install your emulation module.

Installing Software

This section explains how to install the software you will need for your inverse assembler 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 (16600/700-series logic analysis systems)

If you ordered an inverse assembler 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 Agilent Technologies B4620B Source Correlation Tool Set is installed with the logic analysis system's operating system.

To install software from CD-ROM (16600/700-series logic analysis systems)

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 Agilent Technologies 16600/700-series logic analysis system’s 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 Select the **System Administration** icon.

4 Select the **Software Install** tab.

5 Select **Install...**.

Change the media type to “**CD-ROM**” if necessary.

6 Select **Apply**.

7 From the list of types of packages, double-click “**PROC-SUPPORT**.”

NOTE: For touch screen systems, double select the **“PROC-SUPPORT”** line by quickly touching it twice.

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

8 Select the “MPC8XX” package.

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

9 Select **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 /logic/configs/hp/mpc8xx. The inverse assemblers are stored in /logic/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.

4

Probing the Target System

Connecting the Logic Analyzer to the Target System

Each table on the following pages corresponds to a particular logic analyzer. The tables contain connection diagrams for the analysis probe for that logic analyzer. If you are using the inverse assembler only, and have used the recommended signal routing for the headers, these tables will apply to your target system also. You can also use the Setup Assistant to guide you through the connection process. See page 22.

CAUTION: Be sure to power down the target system before connecting or disconnecting cables. Otherwise, you may damage circuitry in the analyzer or target system.

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

- 1660A/AS/C/CS/CP/E/ES/EP logic analyzer - see page 67
- 1661A/AS/C/CS/CP/E/ES/EP logic analyzer - see page 68
- 1670A/D/E logic analyzer - see page 69
- 1671A/D/E logic analyzer - see page 70
- 16550A logic analyzers (1 or 2 cards) - see page 80, 71
- 16554/55/56/57 logic analyzer (2 or 3 cards) - see page 73 - 74
- 16600A logic analyzer - see page 76
- 16601A logic analyzer - see page 78
- 16602A logic analyzer - see page 79
- 16710/11/12A logic analyzers (1 or 2 cards) - see page 80 - 81
- 16715/16/17/18/19A logic analyzers (2 or 3 cards) - see page 83 - 84
- 16750/51/52A logic analyzers (2 or 3 cards) - see page 83 - 84

Number of Pods Used/Required

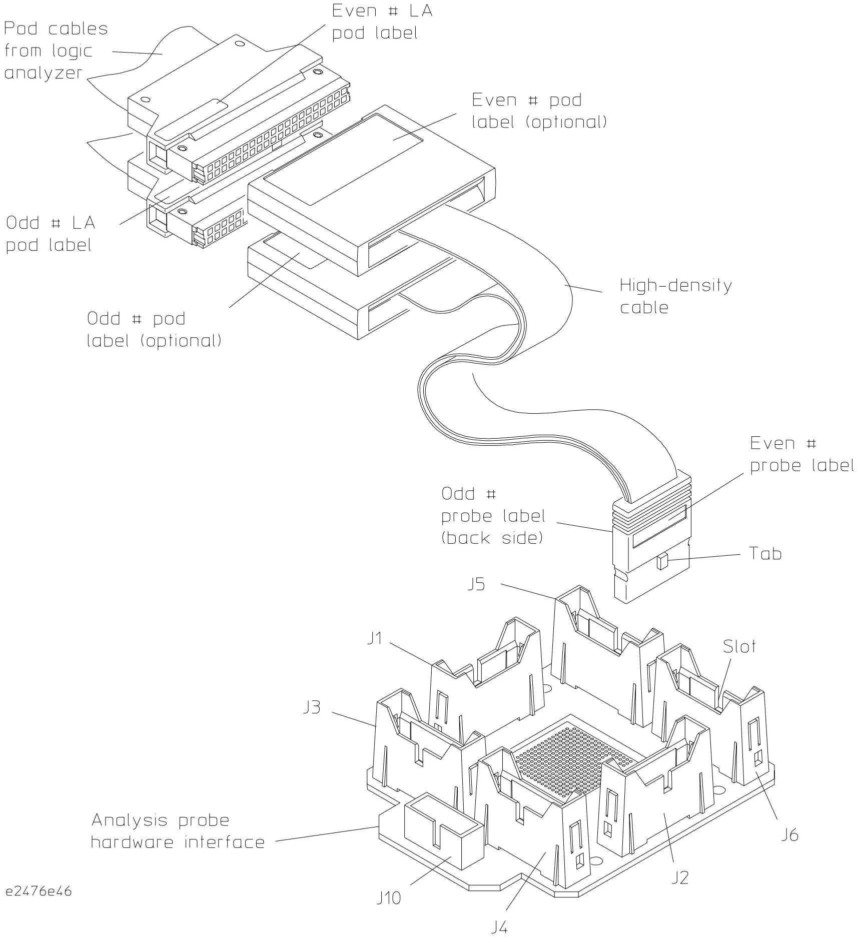
The figure on the following page shows the connectors on the analysis probe. Note that only J1, J2, and J3 (six pods) are required for inverse assembly. If fewer than 6 pods are used, only configuration file definitions for the available pods will be used.

J4, J5, and J6 contain additional signals you might want to monitor; however, they must be connected according to the diagrams on the following pages.

NOTE: If you have a 16600A/700-series logic analysis system with a logic analyzer card not listed here, use the Setup Assistant to connect and configure your logic analyzer.

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

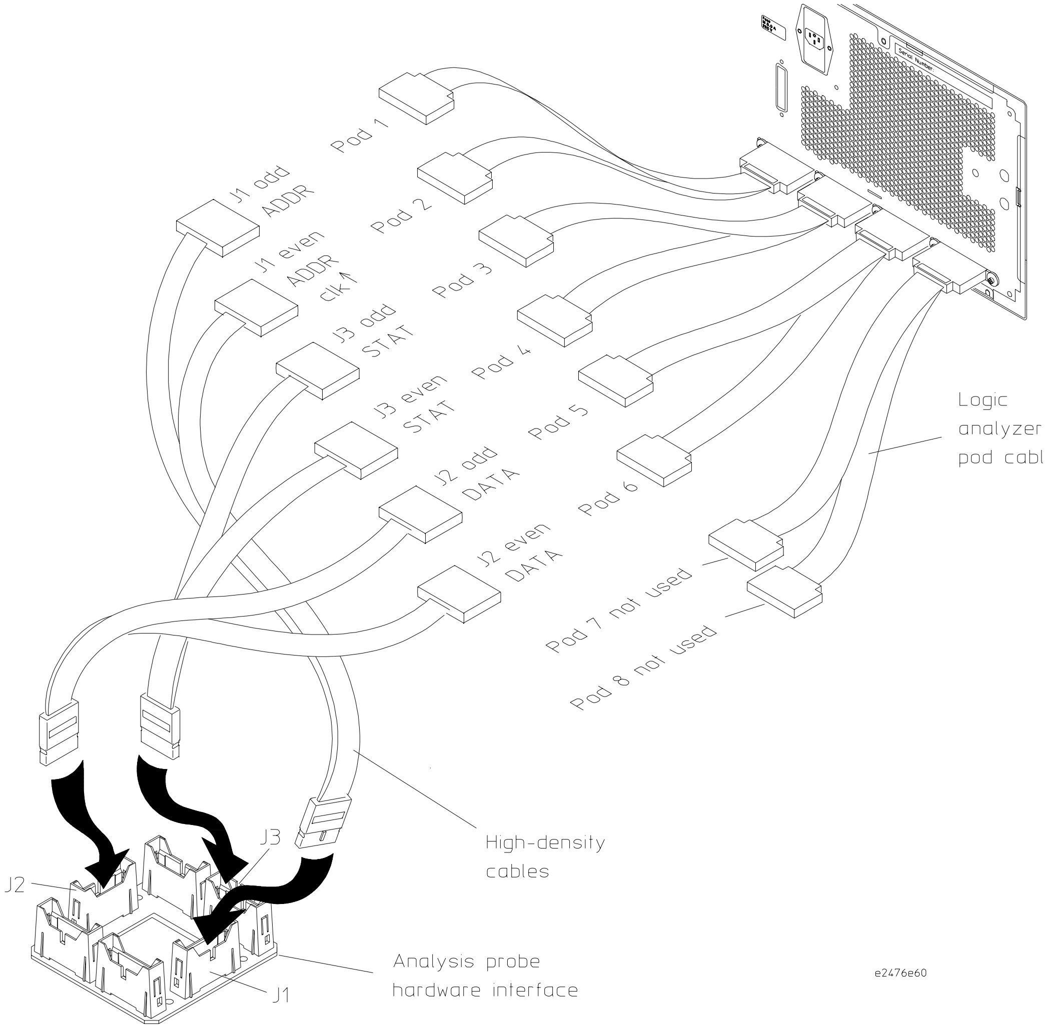
Three Agilent Technologies E5346A high-density termination cables, and labels to identify them, are included with the E2476B. Connect the cables to the connectors on the analysis probe as shown in the illustration below. Attach the labels to the cables after connecting the cables to the logic analyzer.


Pod cables from logic analyzer
Even # LA pod label
Even # pod label (optional)
Odd # LA pod label
High-density cable
Odd # pod label (optional)
Even # probe label
Odd # probe label (back side)
Tab
J5
J1
Slot
J3
Analysis probe hardware interface
J6
J2
e2476e46
J10
J4

Connecting High-Density Cables to the Analysis Probe

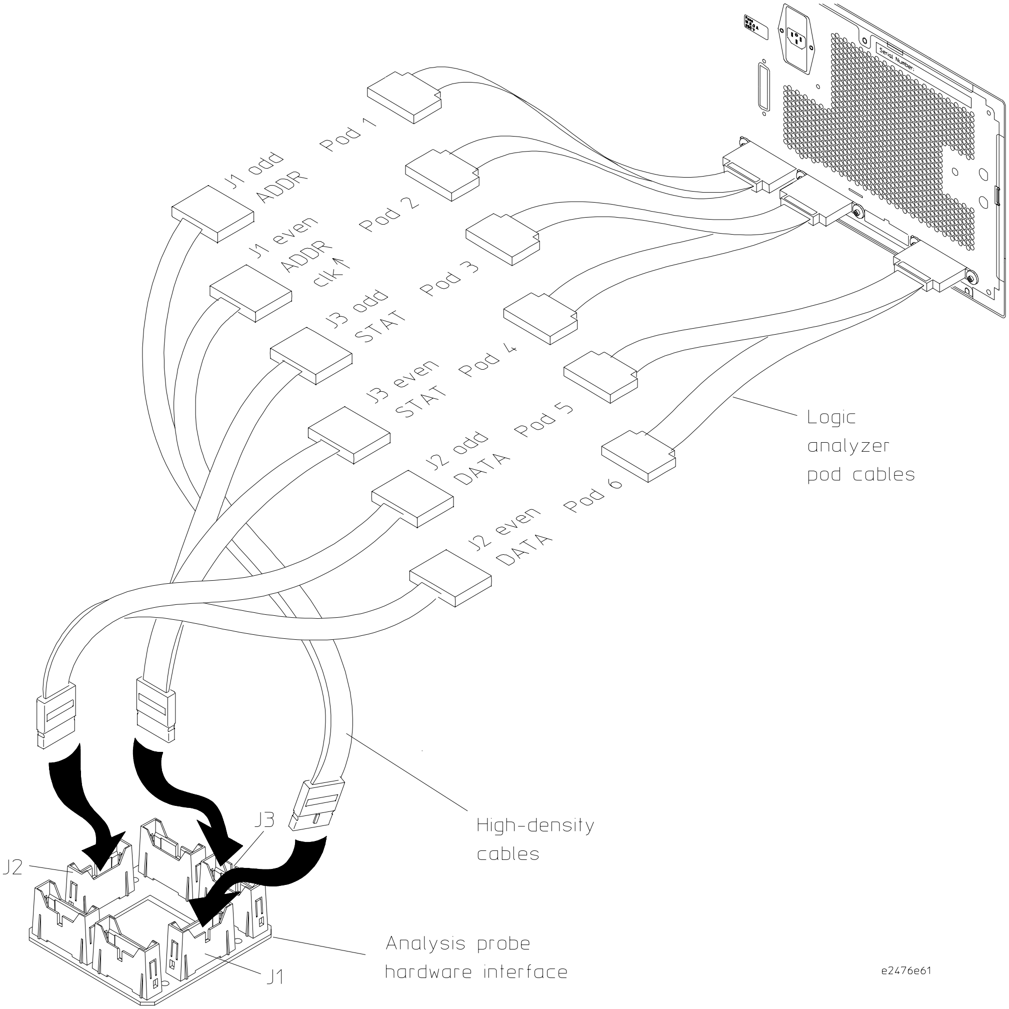
To connect to the 1660A/AS/C/CS/CP/E/ES/EP logic analyzers

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



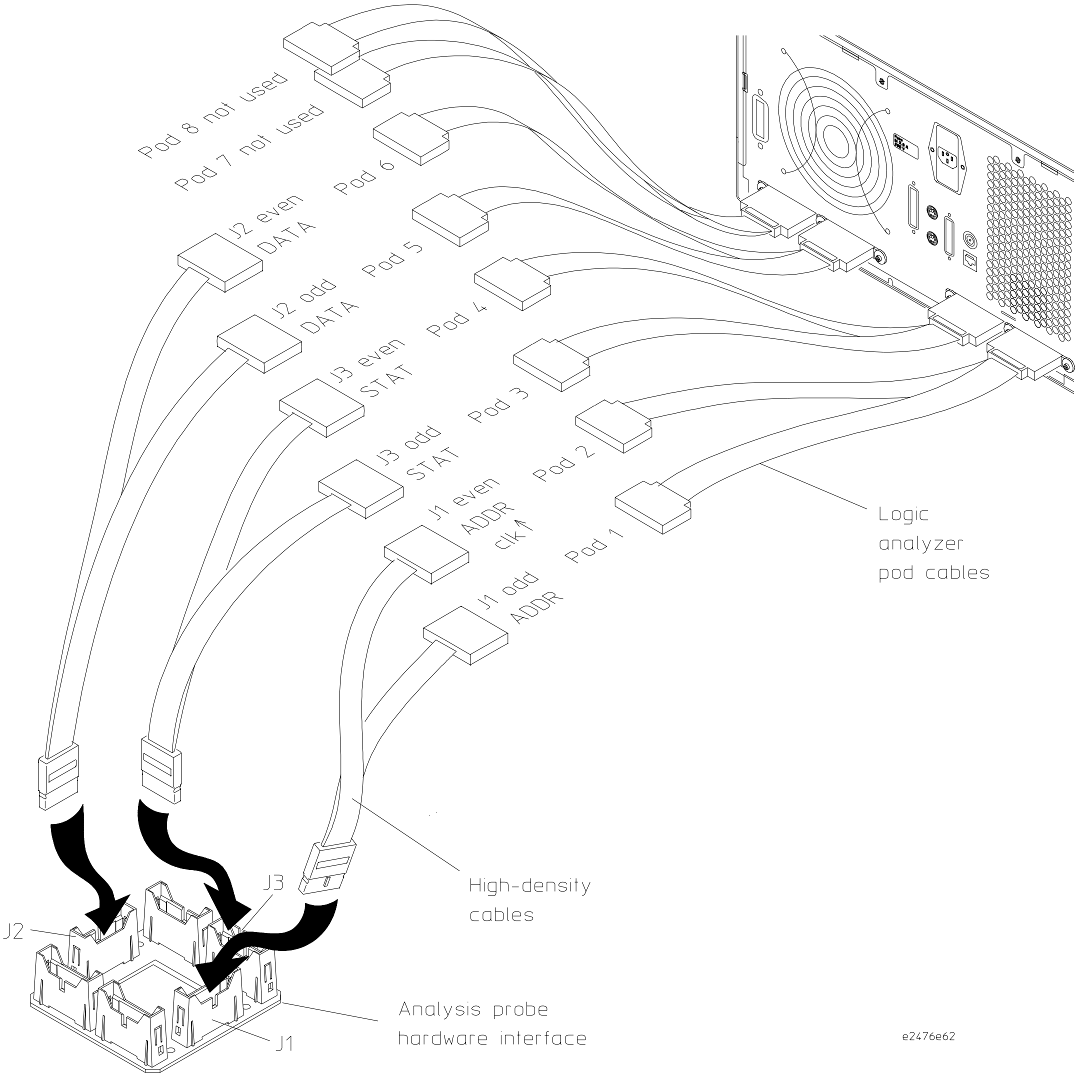
To connect to the 1661A/AS/C/CS/CP/E/ES/EP logic analyzers

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



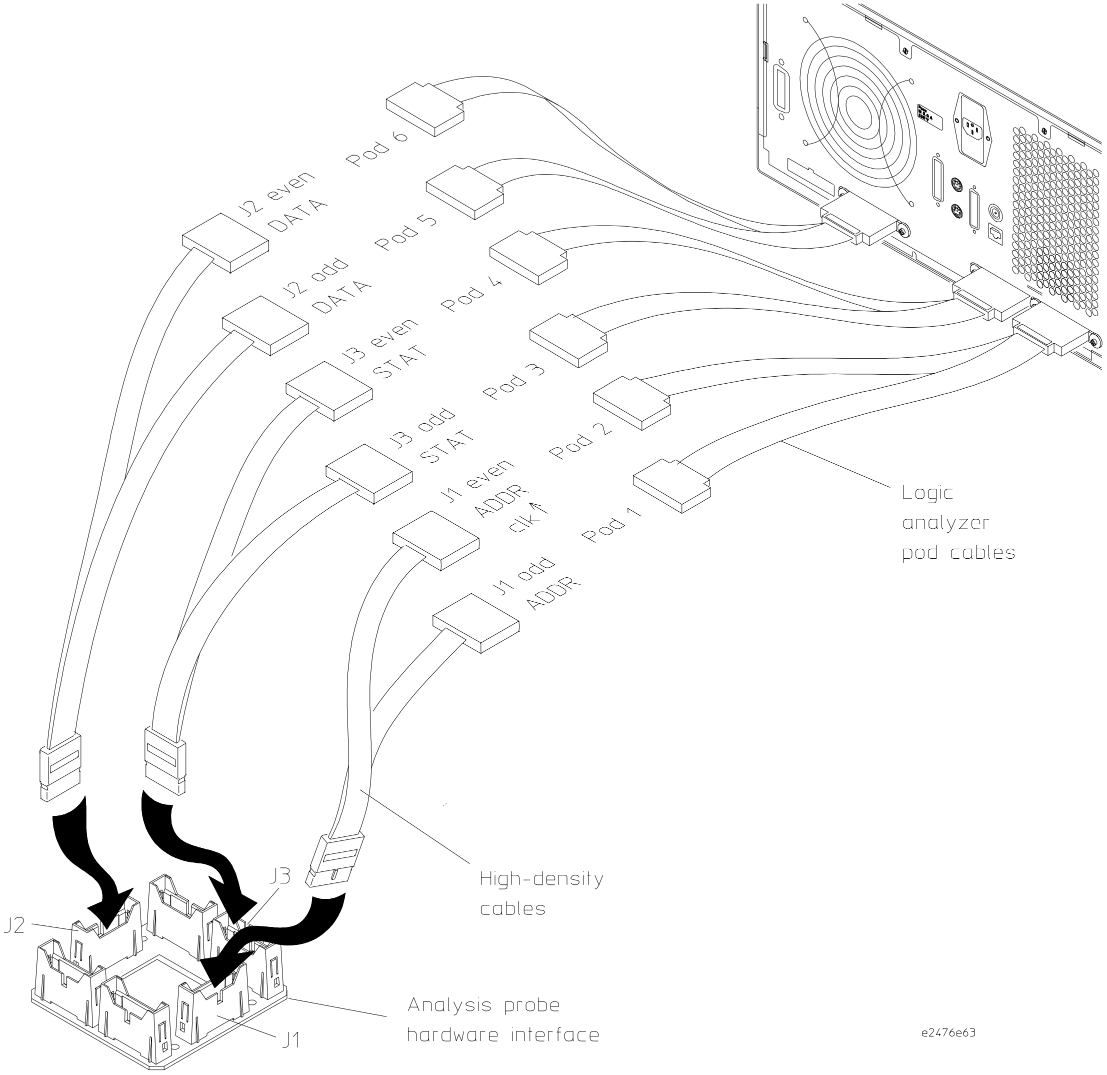
To connect to the 1670A/D/E logic analyzer

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



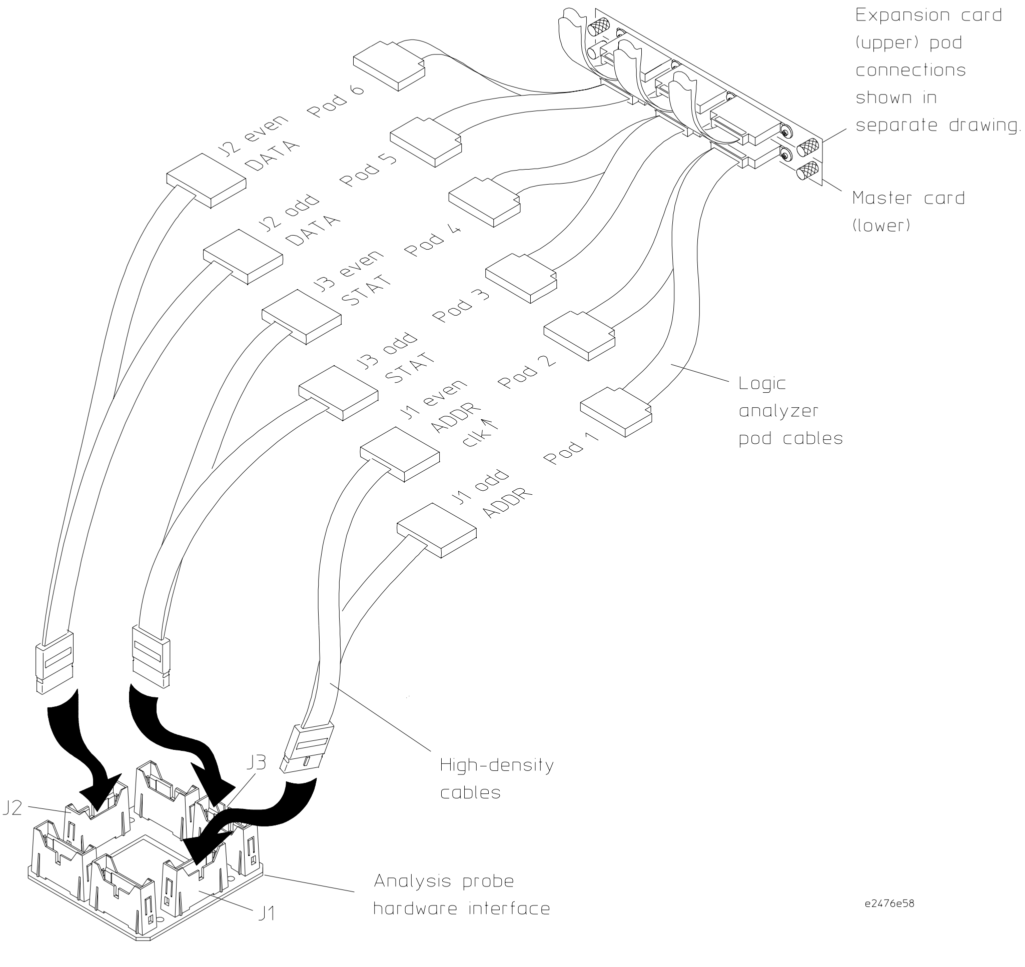
To connect to the 1671A/D/E logic analyzer

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

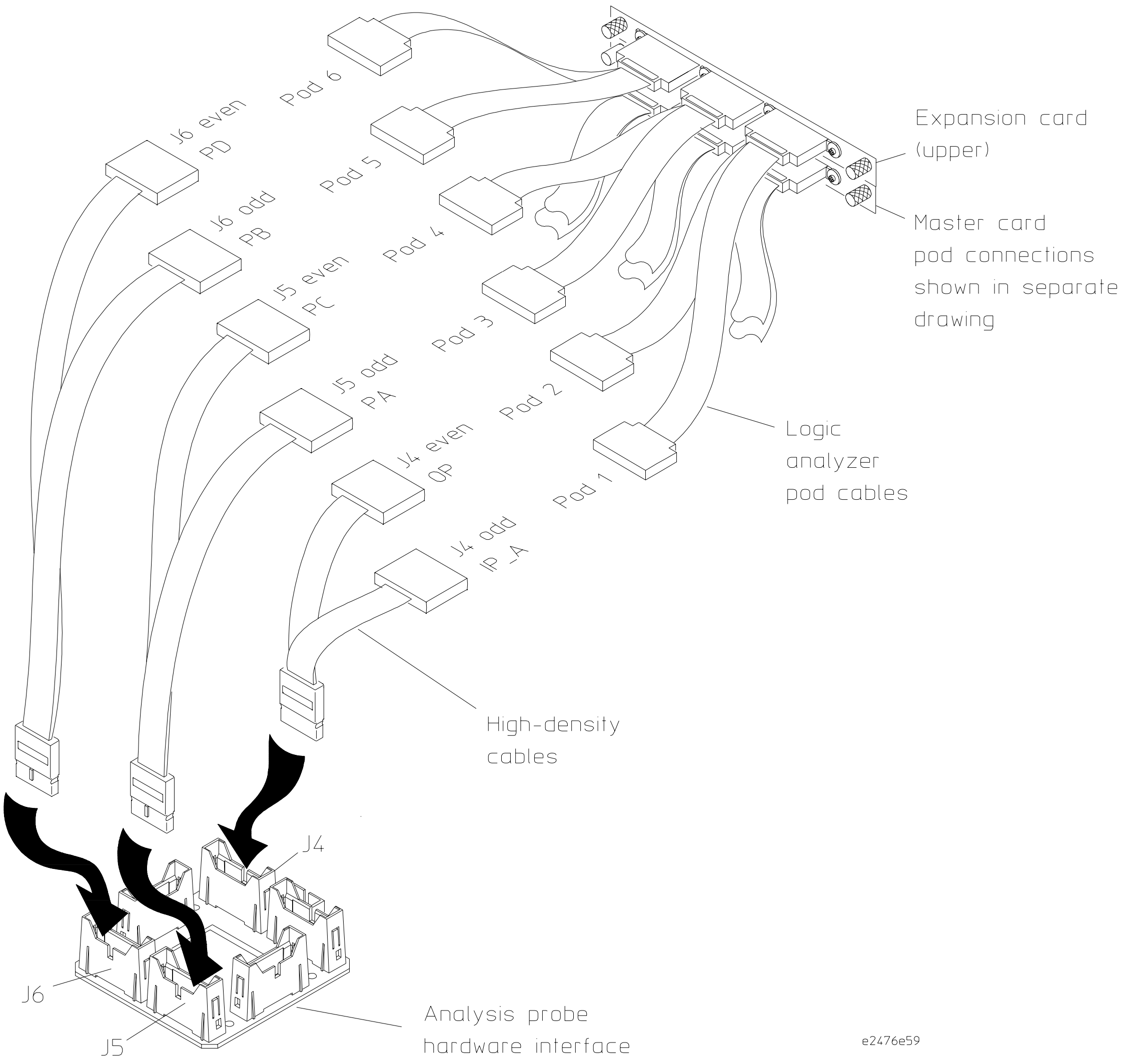


To connect to the 16550A analyzer (two-card)

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



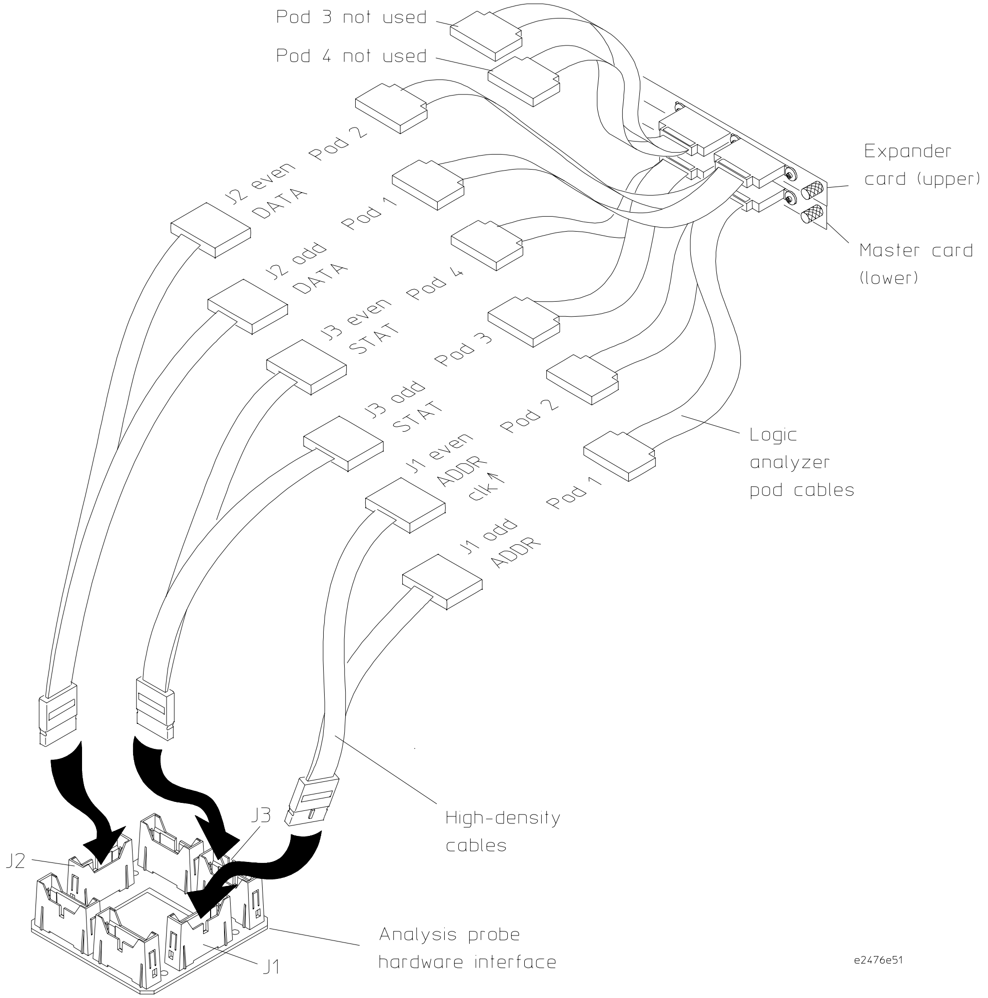
16550A analyzer two-card connections, part 1



16550A analyzer two-card connections, part 2

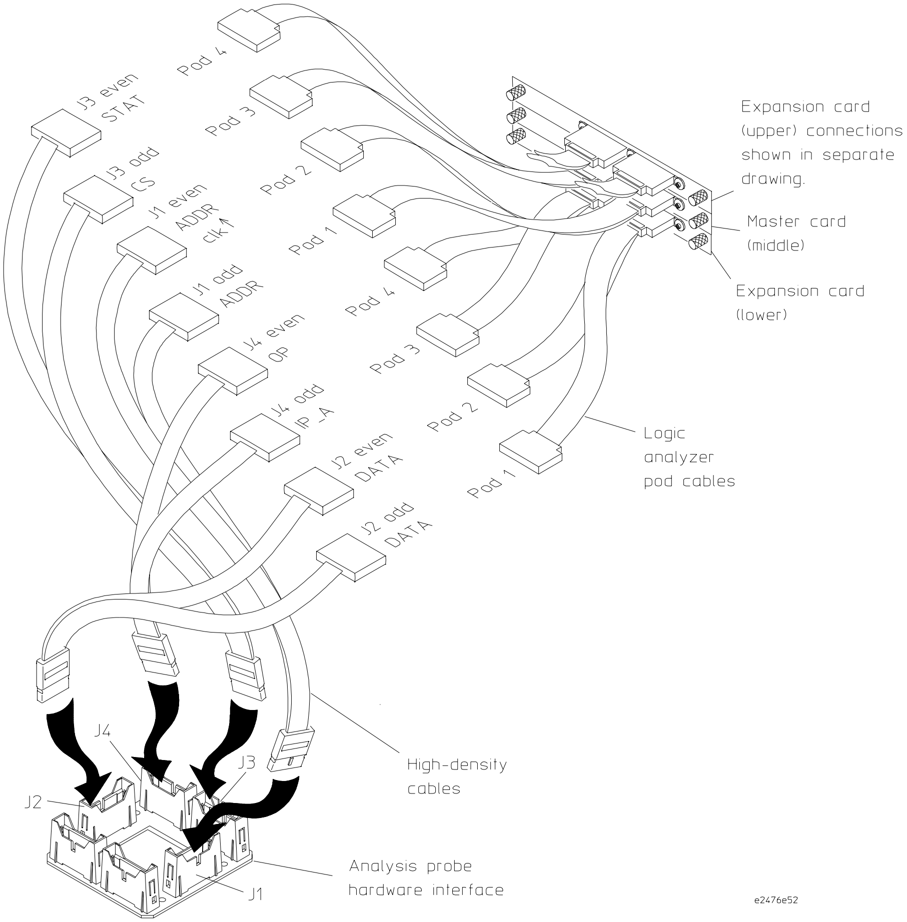
To connect to the 16554/55/56/57 analyzer (two-card)

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

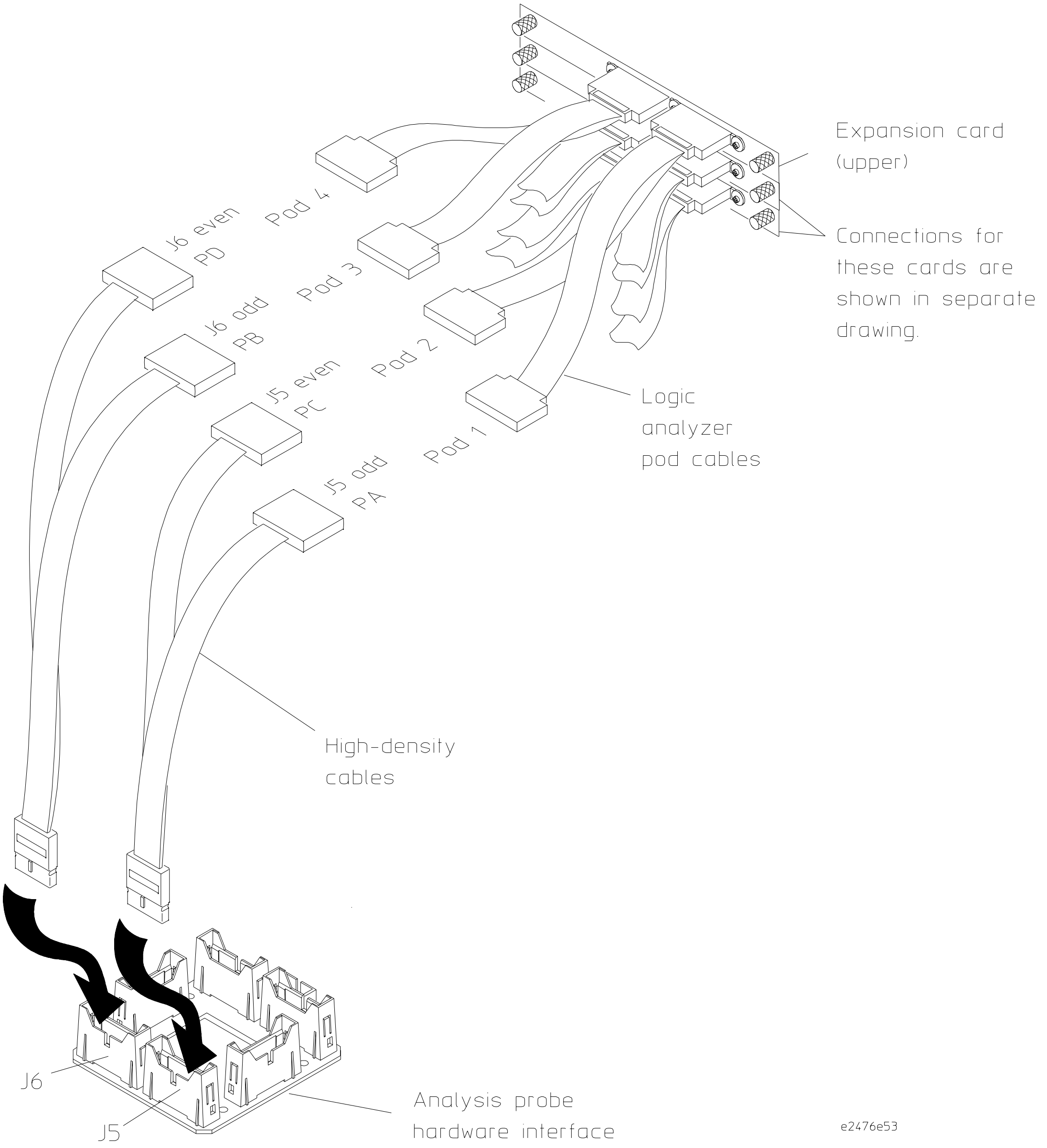


To connect to the 16554/55/56/57 analyzer (three-card)

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



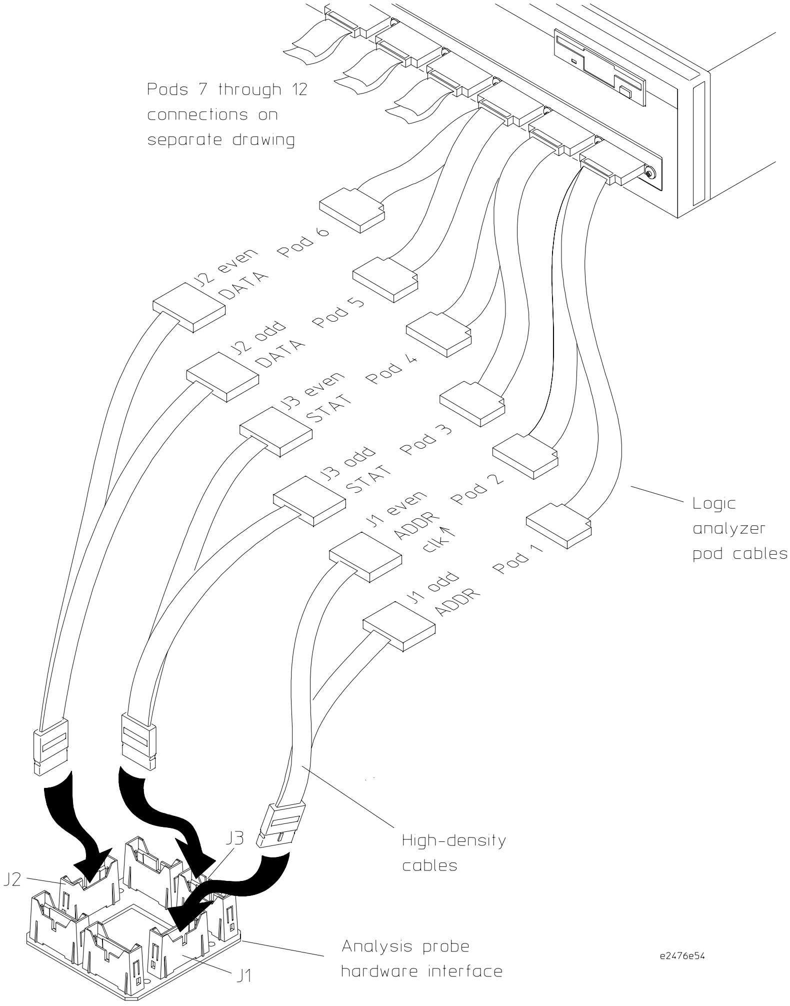
16554/55/56/57 analyzer three-card analyzer connections, part 1



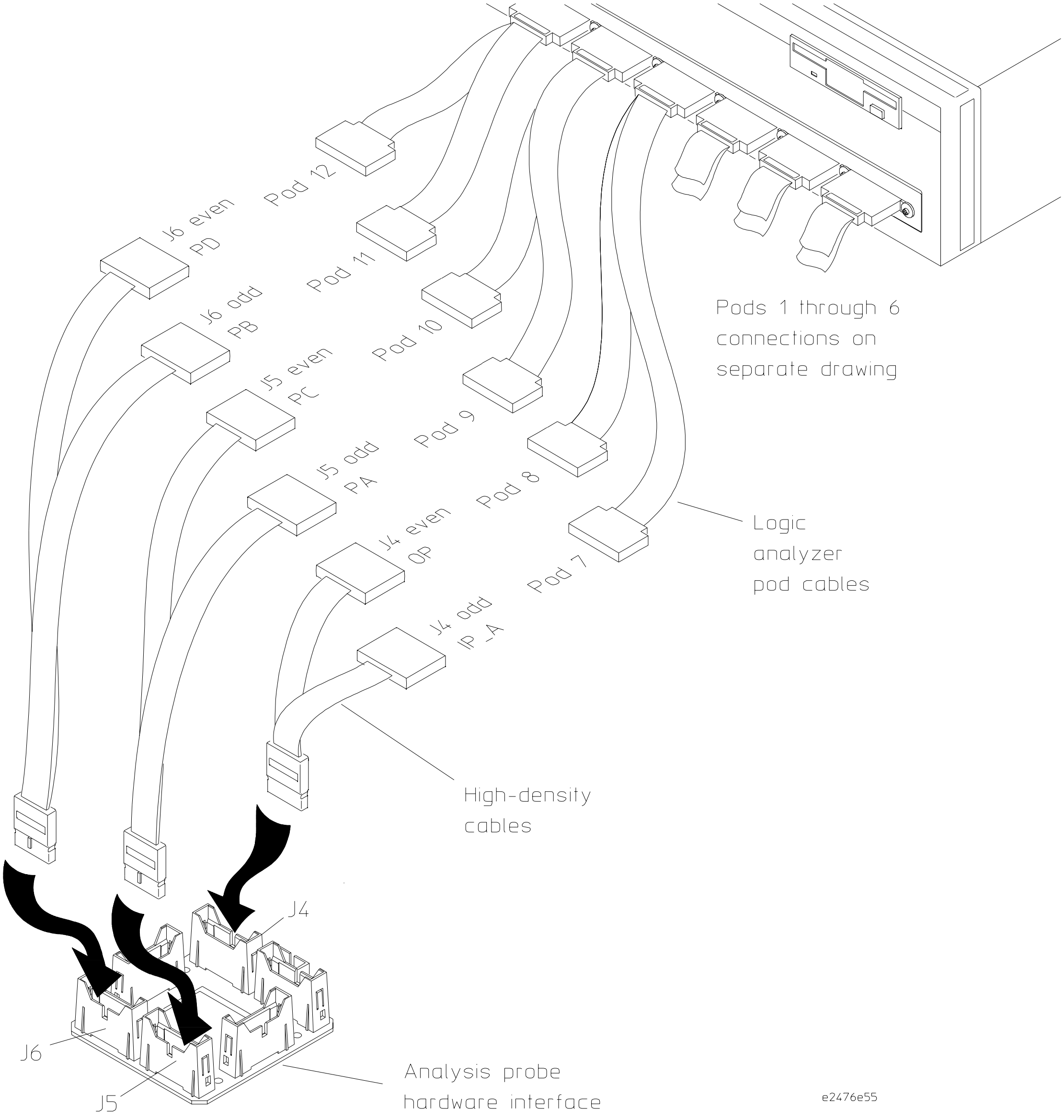
Three-card 16554/55/56/57 analyzer three-card connections, part 2

To connect to the 16600A logic analyzer

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



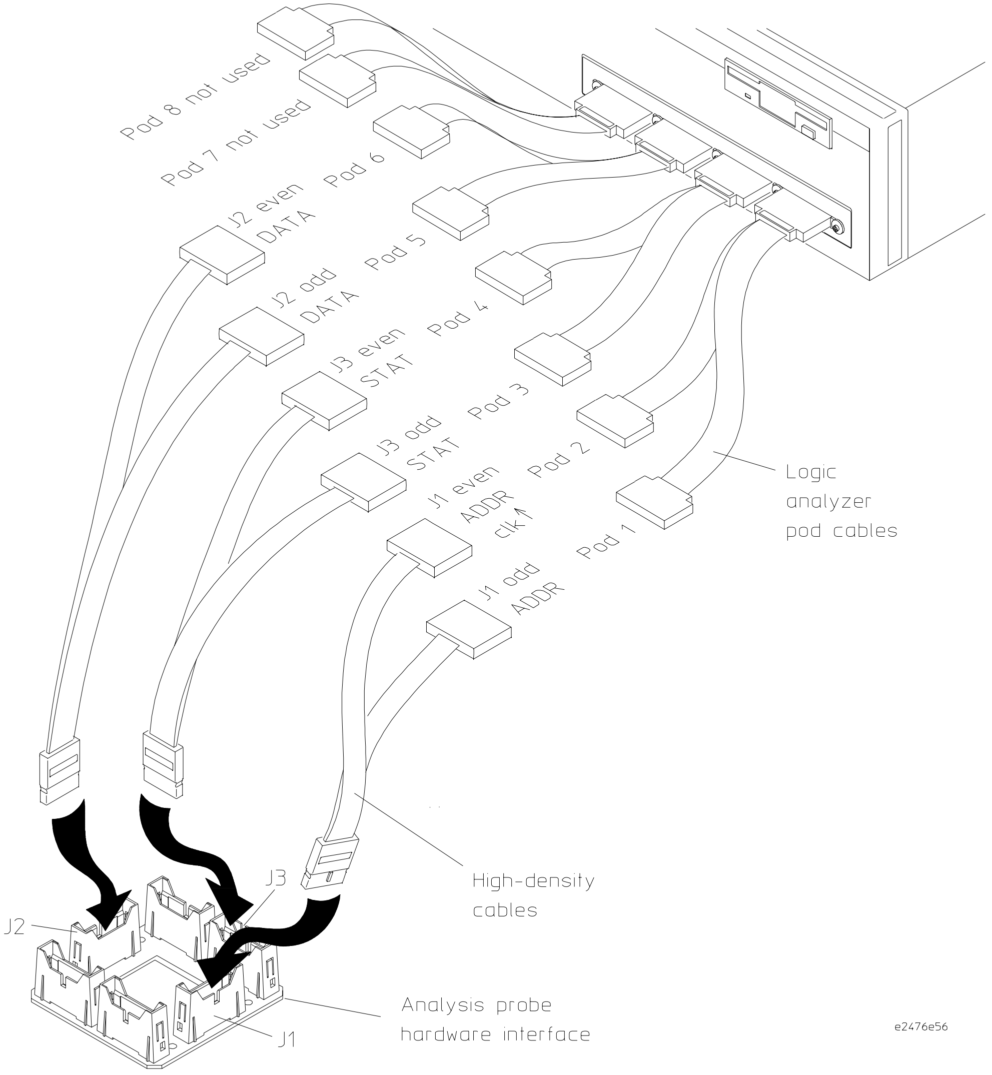
16600A analyzer connections, part 1



16600A analyzer connections, part 2

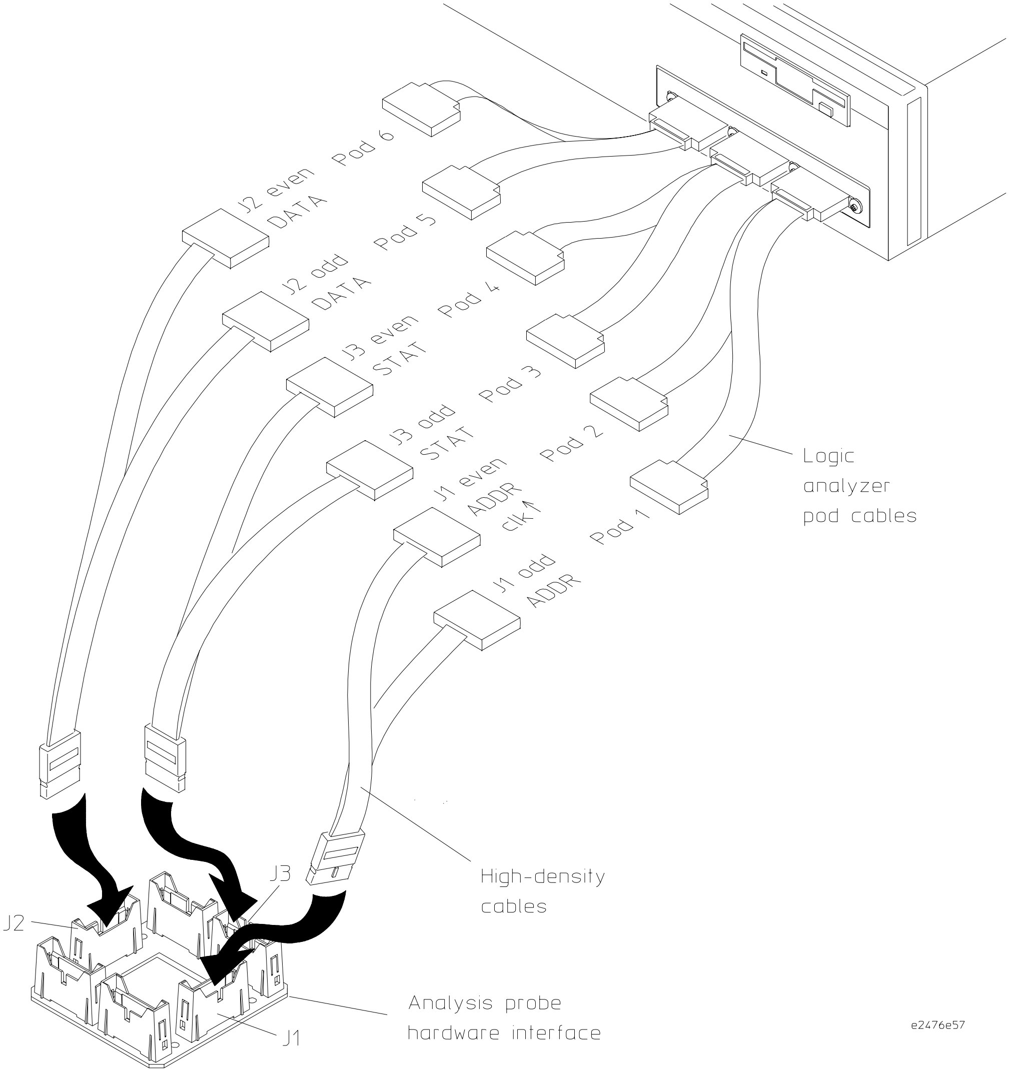
To connect to the 16601A logic analyzer

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



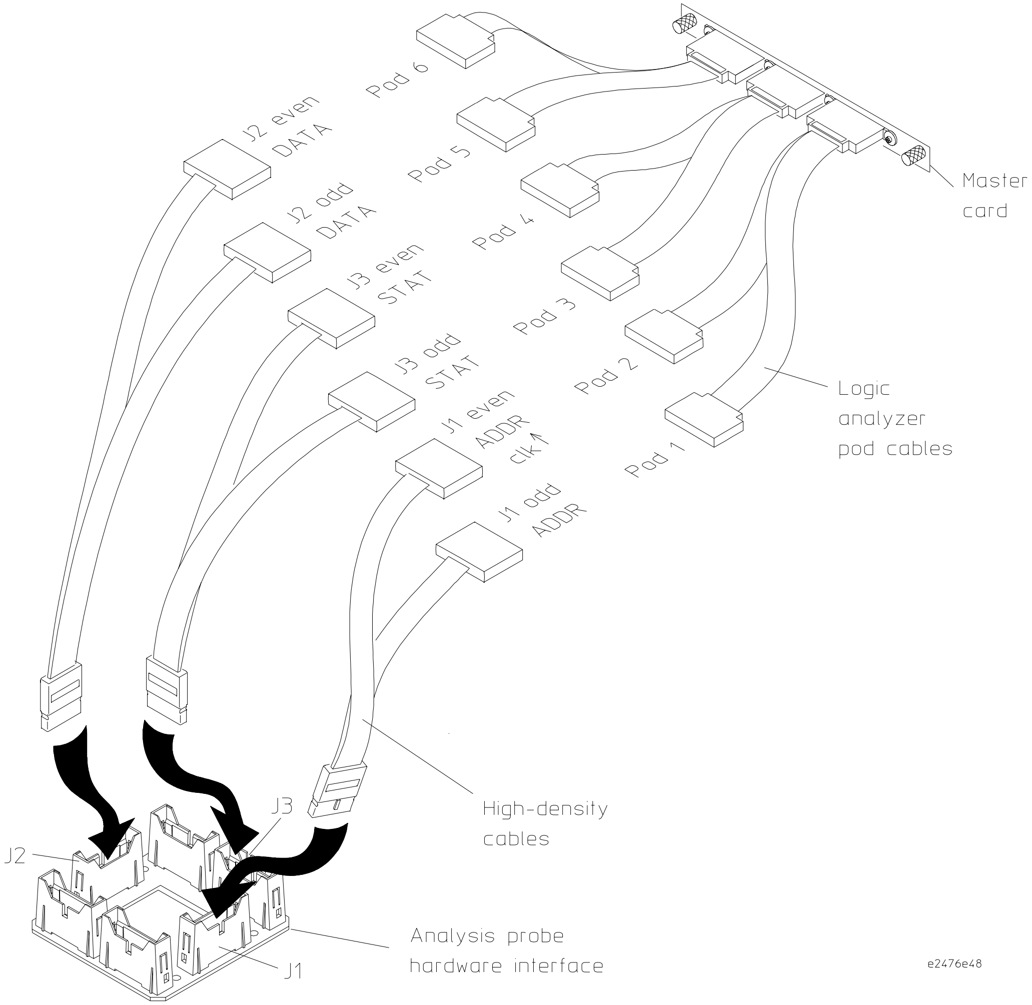
To connect to the 16602A logic analyzer

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



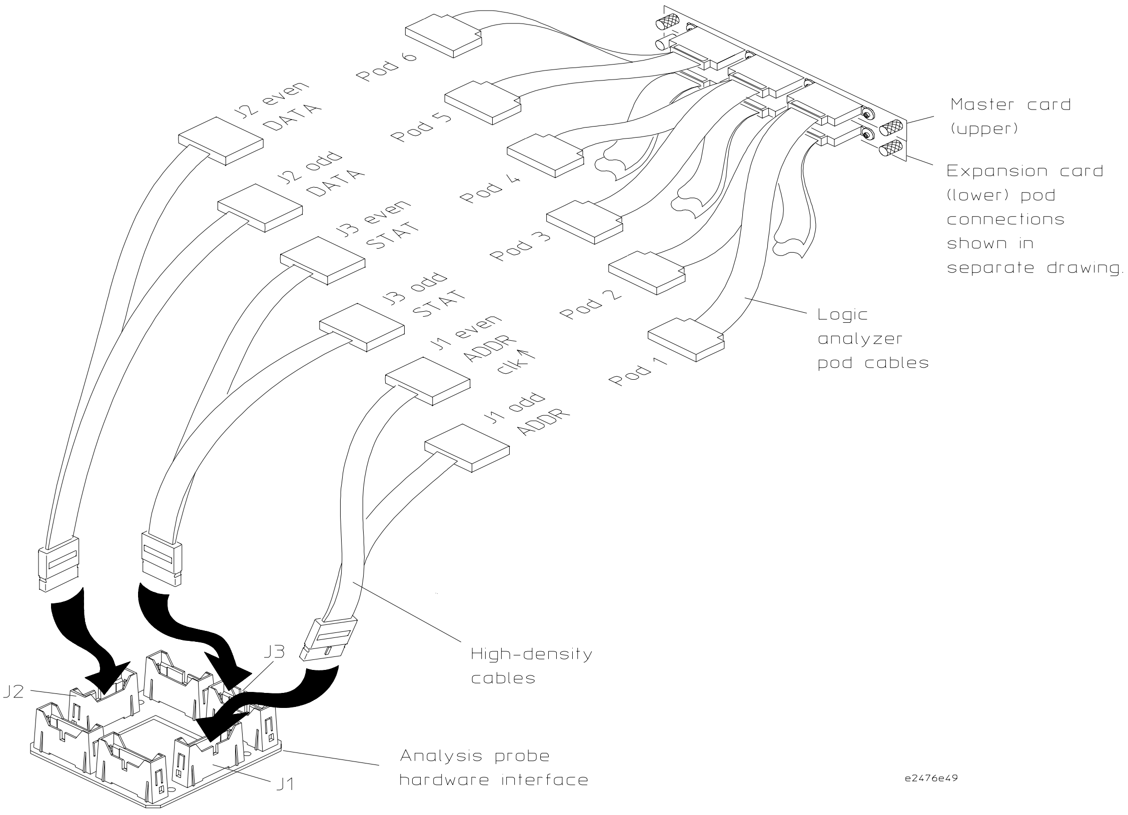
To connect to the 16710/11/12A or 16550A analyzer (one-card)

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

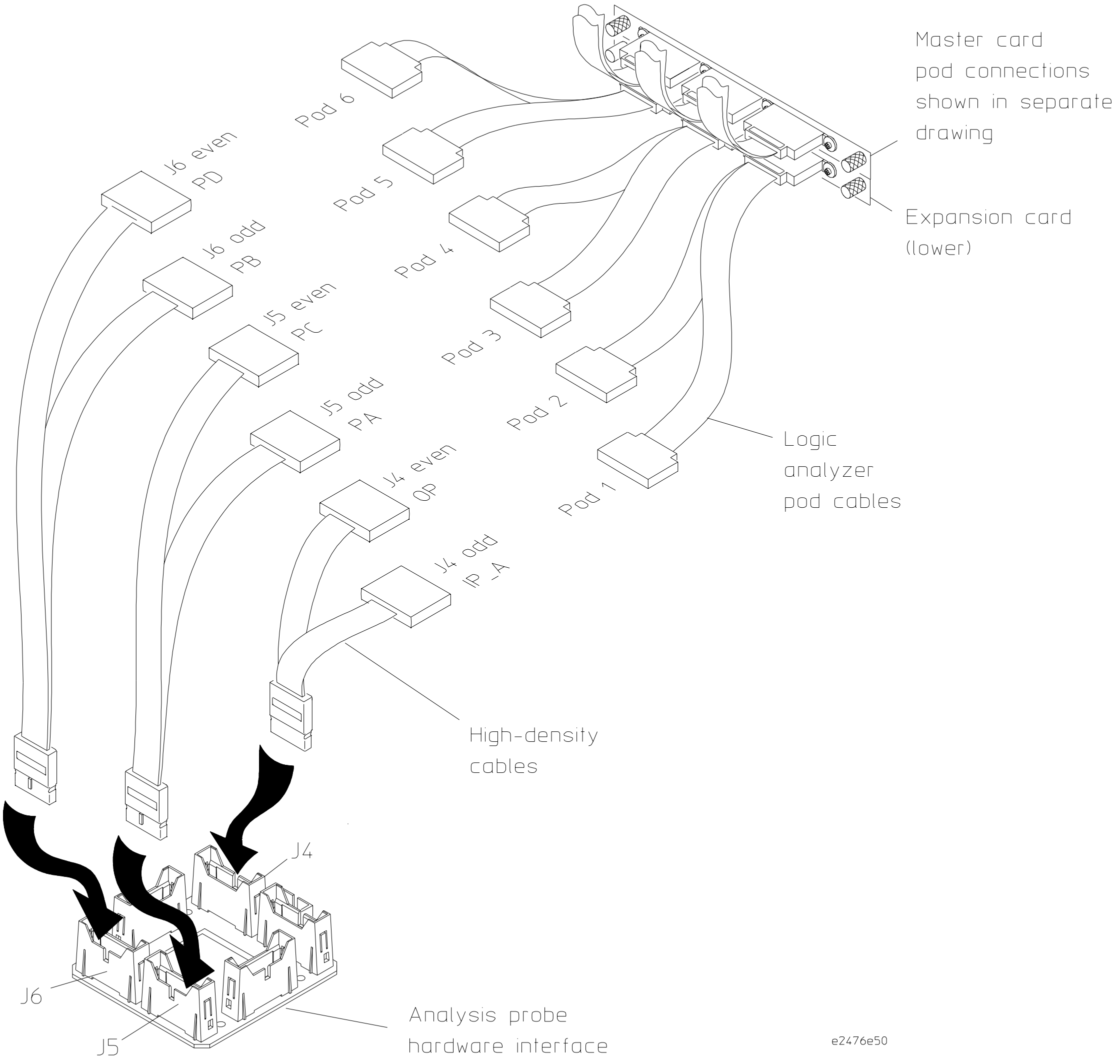


To connect to the 16710/11/12A analyzer (two-card)

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



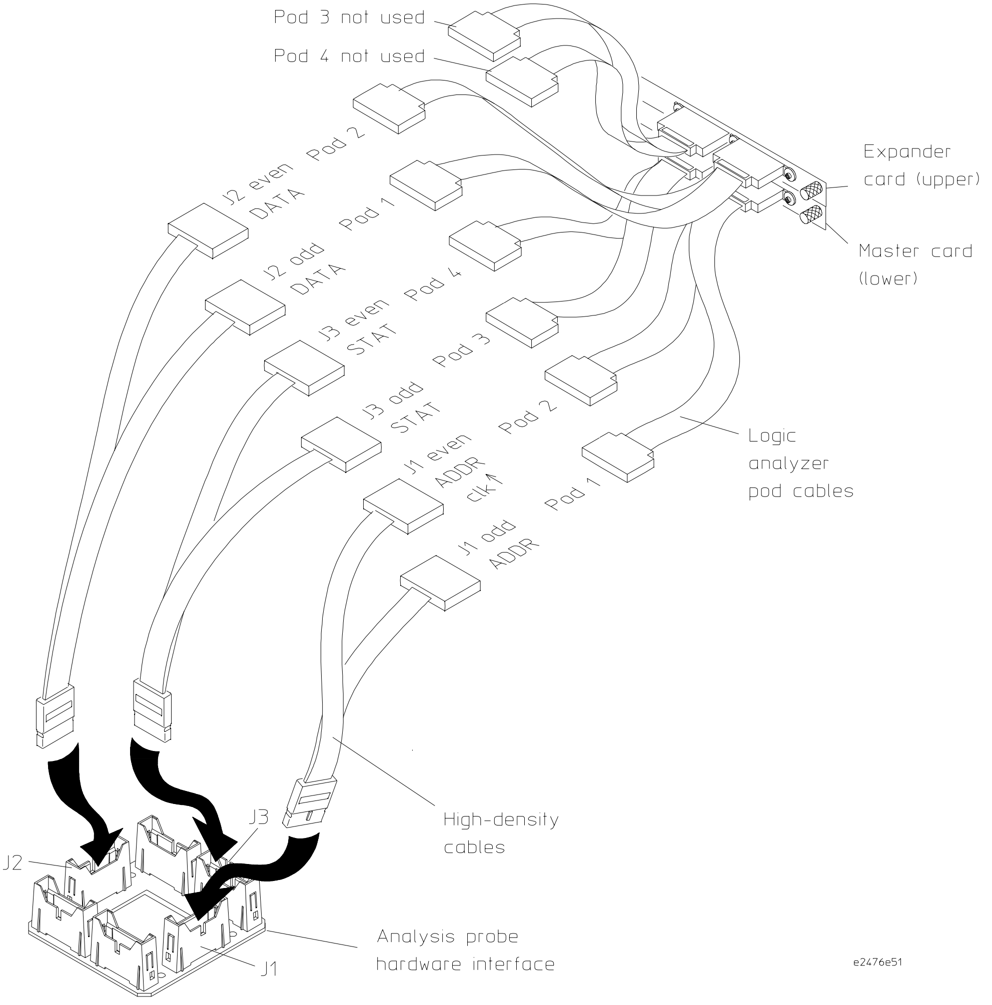
16710/11/12A analyzer two-card connections, part 1



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

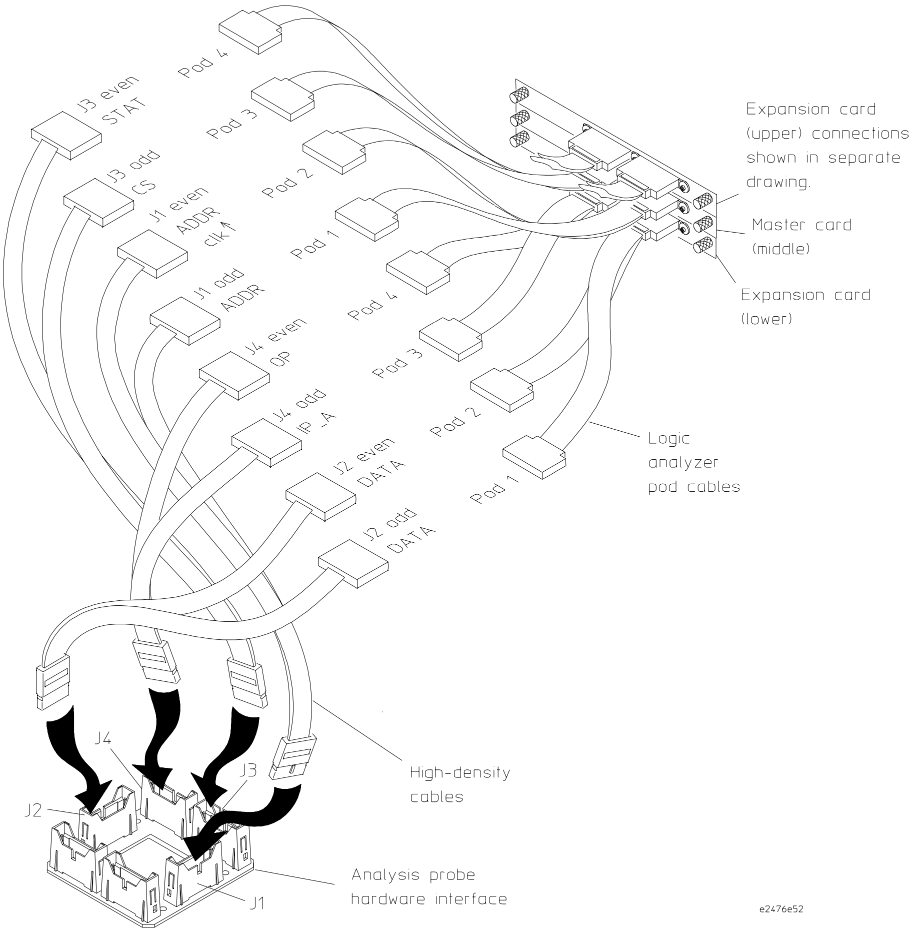
To connect to the 16715/16/17/18/19A or 16750/51/52 analyzers (two-card)

Use the figure below (continued on next page) to connect the analysis probe to the two-card 16715/16/17A or 16750/51/52 logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

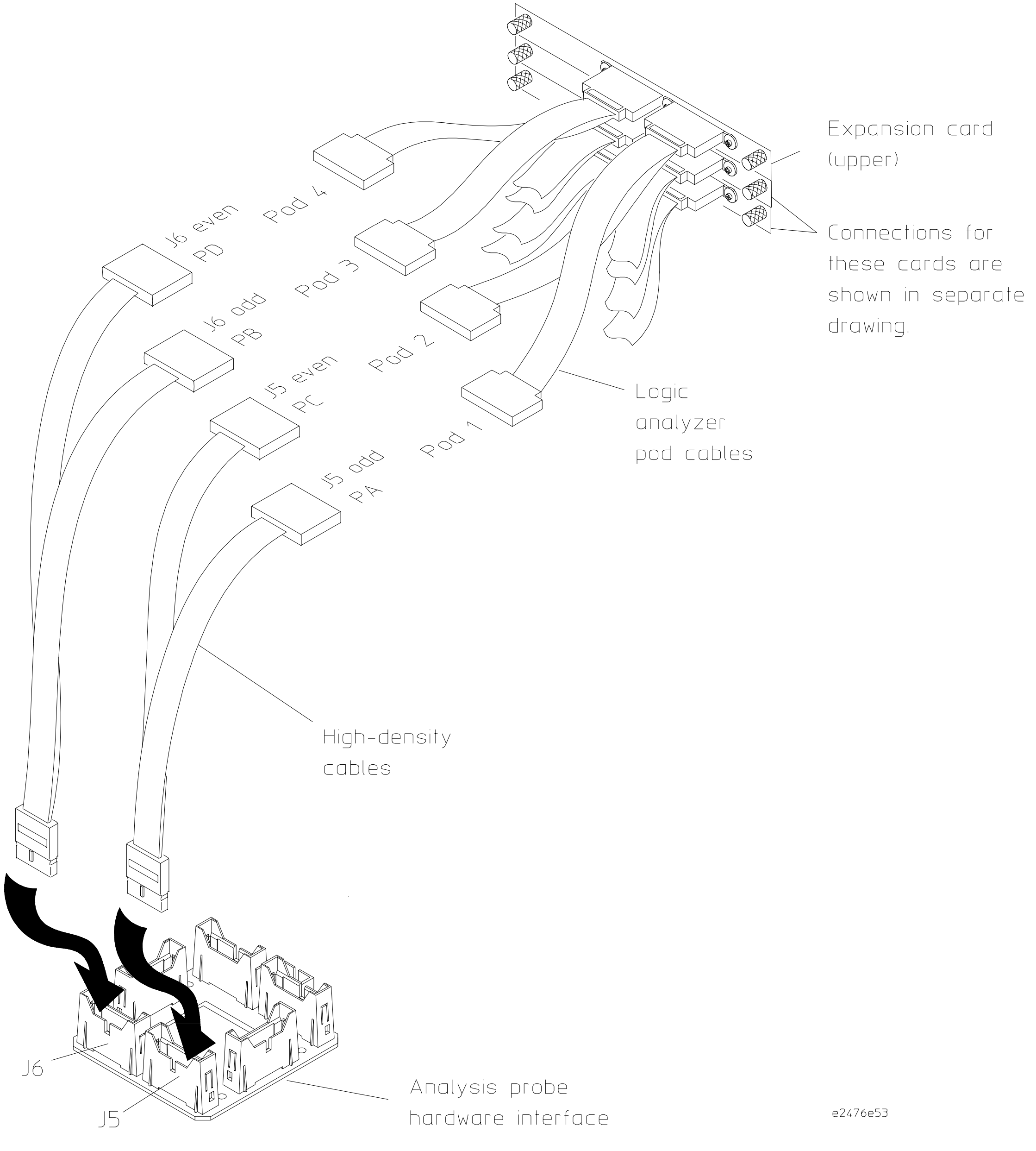


To connect to the 16715/16/17/18/19A or 16750/51/52 analyzers (three-card)

Use the figure below (continued on next page) to connect the analysis probe to the three-card 16715/16/17A or 16750/51/52 logic analyzers. Remove the analysis probe from the target system before connecting the high-density cables. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



16715/16/17/18/19A analyzer three-card connections, part 1


Expansion card (upper)
Connections for these cards are shown in separate drawing.
J6 even PD
Pod 4
J6 odd PB
Pod 3
J5 even PC
Pod 2
J5 odd PA
Pod 1
Logic analyzer pod cables
High-density cables
J6
J5
Analysis probe hardware interface
e2476e53

16715/16/17/18/19A analyzer three-card connections, part 2

5

Configuring the 16600/700-Series Logic Analyzer

The sections of this chapter describe setting up and using the MPC8XX inverse assembler and execution tracker using 16600/700-series logic analysis systems. If you are using 1660/1670/16500B/C-series logic analyzers, see Chapter 6, “Configuring the 1660/1670/16500B/C-Series Logic Analyzer,” beginning on page 123.

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

- Loading the configuration file and the inverse assembler
- Tables showing configuration file names
- Inverse assembler modes of analysis
- Inverse assembler modes of operation
- Changing the acquisition mode
- Using the Invasm menu
- Setting inverse assembler preferences
- Symbols
- Labels
- Disabling the instruction cache for traditional inverse assembly

Configuring 16600/700-series Logic Analysis Systems

You configure the logic analyzer by loading a configuration file. Normally this is done using the Setup Assistant (see page 22). If you did not use the Setup Assistant, you can load the configuration and inverse assembler files from the logic analysis system hard disk.

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, and whether you are performing state or timing analysis.

To load configuration files (and the inverse assembler) from hard disk

If you use Setup Assistant, it will load configuration files and the inverse assembler for you. This is the preferred method. If you did not use Setup Assistant, you can load the configuration and inverse assembler files from the logic analysis system hard disk.

1 Select the File Manager icon. Use File Manager to ensure that the subdirectory /logic/configs/hp/mpc8xx/ exists.

   If the above directory does not exist, you need to install the MPC8XX Processor Support Package. Close File Manager, then use the procedure on the CD-ROM jacket to install the MPC8XX Processor Support Package before you continue. See “Installing Software” on page 59 for details.

2 Using File Manager, select the configuration file you want to load in the /logic/configs/hp/mpc8xx/ directory, then select **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 MPC8XX analysis by loading the appropriate MPC8XX configuration file. Loading the indicated file also automatically loads the inverse assembler with cache-on trace reconstruction. The configuration file names are shown in the table on page 93.

3 Close File Manager.

To load configuration files (and the inverse assembler) from floppy disk

If you use Setup Assistant, it will load configuration files and the inverse assembler for you. This is the preferred method. If you did not use Setup Assistant, you can load the configuration and inverse assembler files from the logic analysis system hard disk or floppy disk; however, the preferred method is to install this functionality from the CD-ROM onto the hard disk and load from the hard disk.

To install a configuration and inverse assembler file from a floppy disk:

1 Insert the floppy disk in the floppy drive on the Agilent 16600/16700-series logic analysis system mainframe.

2 In the logic analysis System window, select 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.
- Select 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 Select **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 Select the Workspace window icon to see the arrangement of analysis tools in your configuration.

6 Select 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.

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

This procedure restores the configuration that was in effect when the configuration file was saved. Because the file was not saved using your system, you may receive error messages about loading the enhanced inverse assembler or about pods that were truncated. Select 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 selecting the **Save** tab in the File Manager. For details about how to save configuration files, open the Help window.

To list software packages that are installed

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

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

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I8XXE)
16550A (1 card)	100 MHz STATE 250 MHz TIMING	c8XXF_1
16550A (2 card)	100 MHz STATE 250 MHz TIMING	c8XXF_2
16554A (2 card)	0.5 M SAMPLE 70/250 MHz LA	c8XXM_2
16555A/D (2 card)	1.0 M SAMPLE 110/250 MHz LA	c8XXM_2
16556A/D (2 card)	1.0 M SAMPLE 100/400 MHz LA	c8XXM_2
16557D (2 card)	2.0 M SAMPLE 135/250 MHz LA	c8XXM_2
16554A (3 card)	0.5 M SAMPLE 70/250 MHz LA	c8XXM_3
16555A/D (3 card)	1.0 M SAMPLE 110/250 MHz LA	c8XXM_3
16556A/D (3 card)	1.0 M SAMPLE 100/400 MHz LA	c8XXM_3
16557D (3 card)	2.0 M SAMPLE 135/250 MHz LA	c8XXM_3
16600A	na	c8XXF_2
16601A	na	c8XXF_2
16602A	na	c8XXF_1

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I8XXE)
16710/11/12A (1 card)	100 MHz STATE 250 MHz TIMING	c8XXF_1
16710/11/12A (2 card)	100 MHz STATE 250 MHz TIMING	c8XXF_2
16715/16/17/18/19A (2 card)	167 MHz STATE 333/667 MHz TIMING	c8XXM_2
16715/16/17/18/19A (3 card)	167 MHz STATE 333/667 MHz TIMING	c8XXM_3
16750/51/52 (2 card)	167 MHz STATE 2 GHz TIMING	c8XXM_2
16750/51/52 (3 card)	167 MHz STATE 2 GHz TIMING	c8XXM_3

Inverse Assembler Modes of Analysis

The inverse assembler offers two modes of analysis for MPC8XX 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 (see page 105).

Traditional inverse assembly

The inverse assembler lets you obtain displays of MPC8XX 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 Agilent Technologies 16600/700-series logic analyzers. It provides much more information when used together with the Agilent Technologies B4620B Source Correlation Tool Set. Source correlation performs a correlation of the addresses from cache with the high-level code execution.

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.

Cache-on trace reconstruction

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

The inverse assembler uses show cycles, the $\overline{\text{STS}}$/VF/VFLS signals, and program image files to decode captured MPC8XX execution into complete program trace. When compared to data that comes straight from the logic analyzer, the data from the inverse assembler:

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

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

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

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

NOTE: Cache-on trace reconstruction requires that all processor cycles are stored, so storage qualification can not be used when cache-on trace reconstruction is enabled. See “Using Storage Qualification” on page 152.

Inverse Assembler Modes of Operation

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

Inverse Assembler Modes of Operation

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

NOTE:

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

Changing the Acquisition Mode

NOTE: It is strongly recommended that you do not change the setup related to the MPC8XX sampling, format, pod assignment or configuration dialogs. The Setup Assistant will configure the logic analyzer for making measurements of the MPC8XX.

The Agilent Technologies E2476B analysis probe can be used in three different operating modes: State-per-ack, State-per-clock, or Timing. The Agilent Technologies E2477A inverse assembler software can be used for State-per-ack and State-per-clock analysis. The following sections describe these operating modes and how to configure the logic analyzer for each mode.

State-per-ack mode

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

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

To change to state-per-clock mode

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

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

To change to timing mode

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

1. Open the logic analyzer's **Setup** window.
2. Select the **Sampling** tab.
3. Change the type option from **State Mode** to **Timing Mode**.

To use the Invasm menu

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

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

Loading the Inverse Assembler

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

Setting the Inverse Assembler Preferences

Why the configuration is necessary

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

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

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

Finding memory bank information using a debugger

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

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

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

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

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

Finding memory bank information at compile time

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

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

Example

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

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

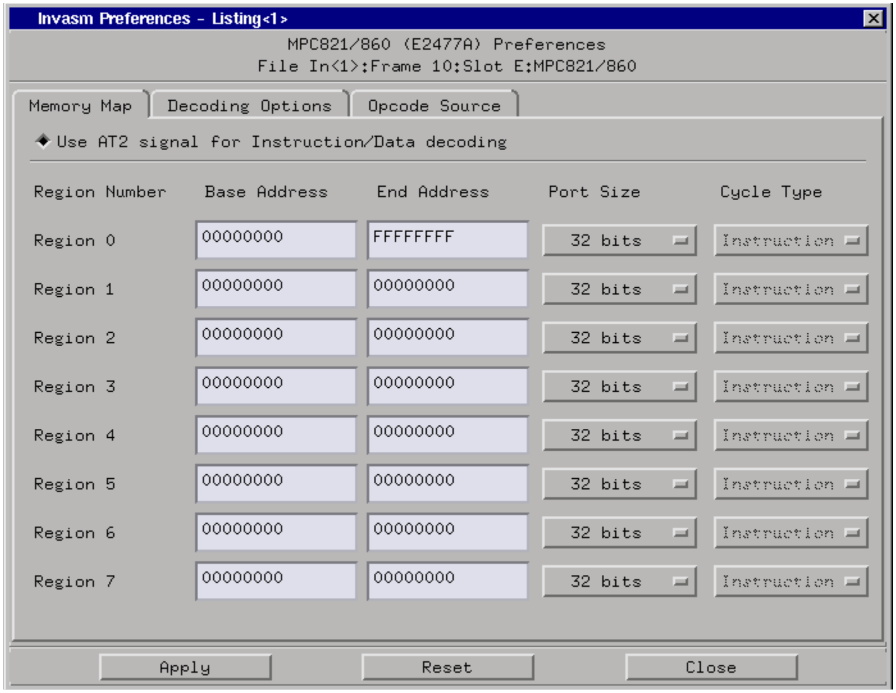
Region	Base Address	End Address	Port Size	Cycle Type
Region 0	00020000	0002FFFF	32 bit	instruction
Region 1	00100000	001FFFFF	32 bit	data

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

To set the memory map preferences

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

Inverse Assembler Preferences Dialog



Invasm Preferences - Listing<1>

MPC821/860 (E2477A) Preferences
File In<1>:Frame 10:Slot E:MPC821/860

◆ Use AT2 signal for Instruction/Data decoding

Region Number	Base Address	End Address	Port Size	Cycle Type
Region 0	00000000	FFFFFFFF	32 bits	Instruction
Region 1	00000000	00000000	32 bits	Instruction
Region 2	00000000	00000000	32 bits	Instruction
Region 3	00000000	00000000	32 bits	Instruction
Region 4	00000000	00000000	32 bits	Instruction
Region 5	00000000	00000000	32 bits	Instruction
Region 6	00000000	00000000	32 bits	Instruction
Region 7	00000000	00000000	32 bits	Instruction

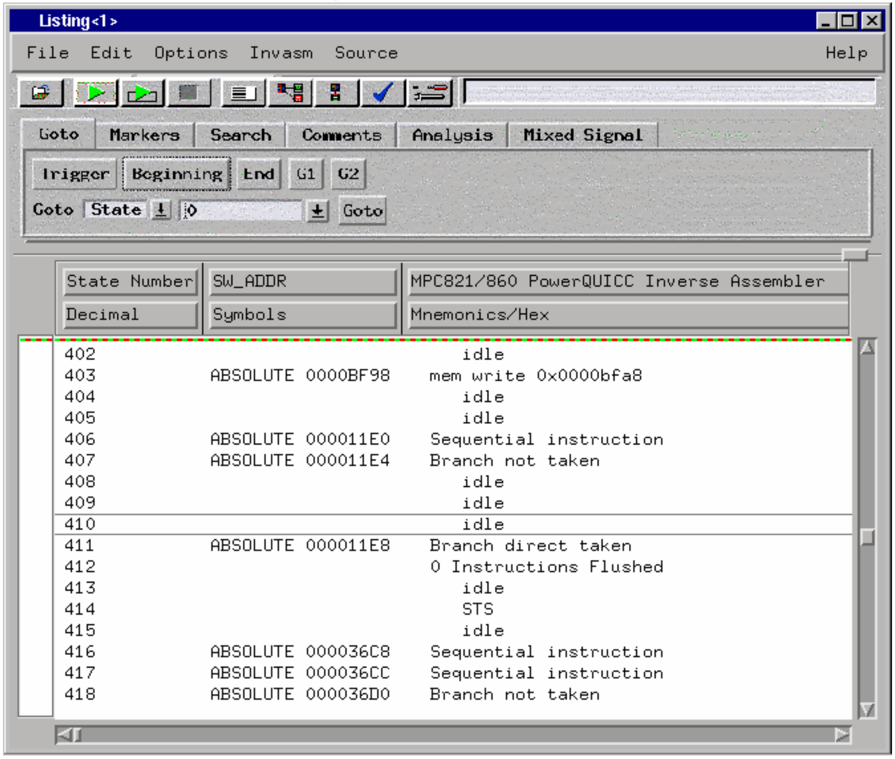
Select **Apply** when you have finished configuring the memory map.

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

Enabling show cycle disassembly

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

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

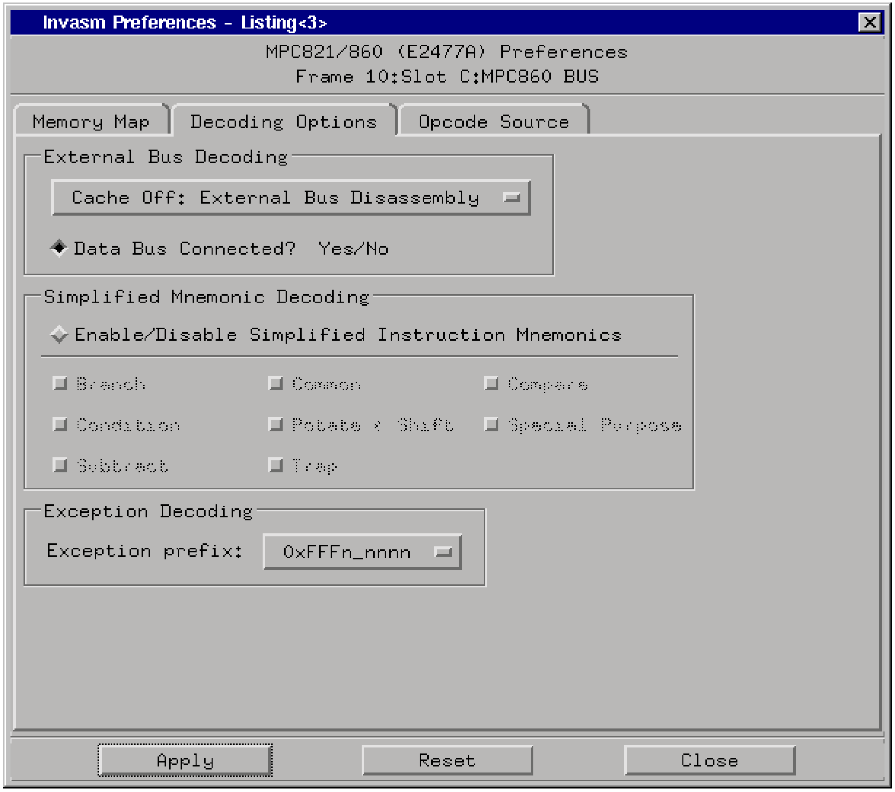


```
State Number   SW_ADDR                MPC821/860 PowerQUICC Inverse Assembler
Decimal        Symbols                Mnemonics/Hex
402                                      idle
403            ABSOLUTE 0000BF98      mem write 0x0000bfa8
404                                      idle
405                                      idle
406            ABSOLUTE 000011E0      Sequential instruction
407            ABSOLUTE 000011E4      Branch not taken
408                                      idle
409                                      idle
410                                      idle
411            ABSOLUTE 000011E8      Branch direct taken
412                                   0 Instructions Flushed
413                                      idle
414                                      STS
415                                      idle
416            ABSOLUTE 000036C8      Sequential instruction
417            ABSOLUTE 000036CC      Sequential instruction
418            ABSOLUTE 000036D0      Branch not taken
```

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

To set the decoding options preferences

Inverse Assembler Decoding Options Dialog



Invasm Preferences - Listing<3>

MPC821/860 (E2477A) Preferences
Frame 10:Slot C:MPC860 BUS

Memory Map | Decoding Options | Opcode Source

External Bus Decoding

Cache Off: External Bus Disassembly

◆ Data Bus Connected? Yes/No

Simplified Mnemonic Decoding

◇ Enable/Disable Simplified Instruction Mnemonics

□ Branch	□ Common	□ Compare
□ Condition	□ Rotate & Shift	□ Special Purpose
□ Subtract	□ Trap	

Exception Decoding

Exception prefix: 0xFFFn_nnnn

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

Data Bus Connected. Read and write states are always indicated regardless of whether the data bus is connected. However, when the data bus is connected, read/write data will also be displayed. See "Inverse Assembler Modes of Operation" on page 97

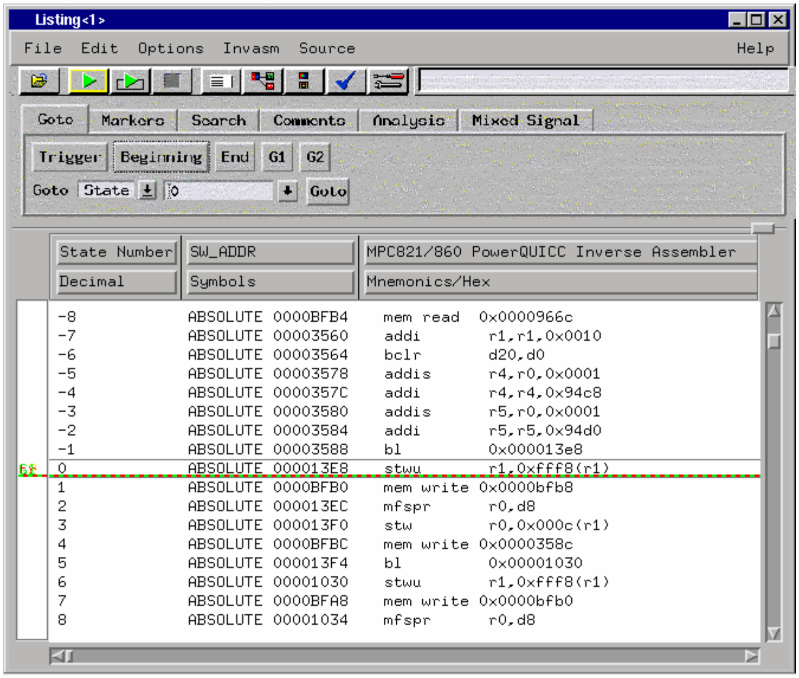
Simplified Mnemonic Decoding. PowerPC assemblers support a number of simplified mnemonics for some popular assembly language instructions, as described in Appendix F of the *PowerPC Microprocessor*

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. Select the options for the simplified mnemonics you desire.

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

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

Listing Window Showing Trace with Data Bus Connected: Cache Off

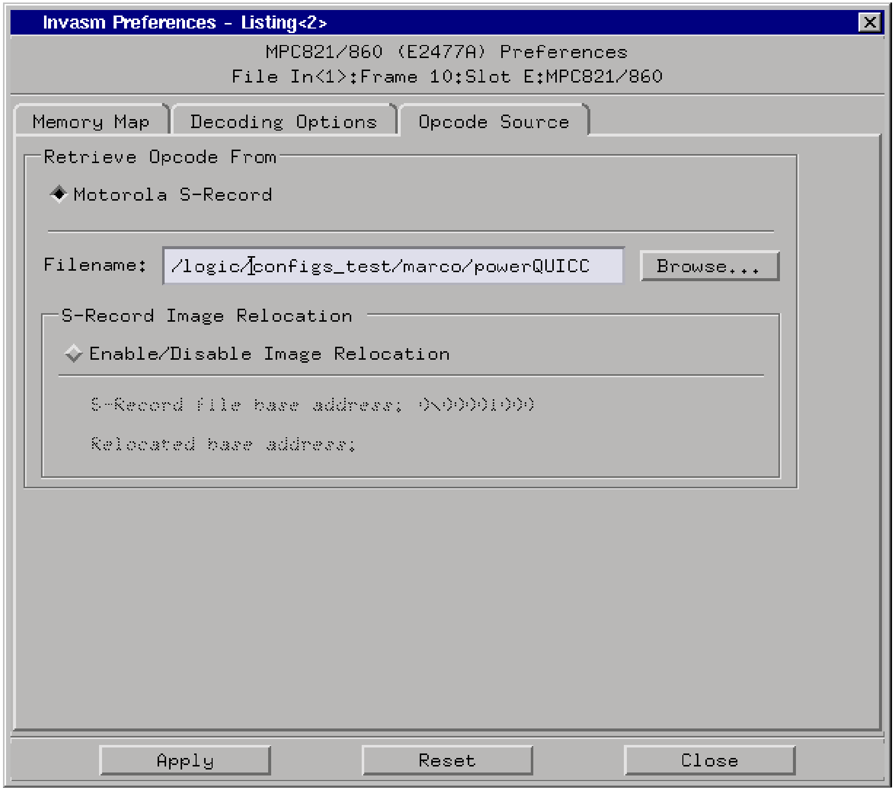


```
State Number   SW_ADDR                MPC821/860 PowerQUICC Inverse Assembler
Decimal        Symbols                Mnemonics/Hex
-8             ABSOLUTE  0000BFB4     mem read   0x0000966c
-7             ABSOLUTE  00003560     addi        r1,r1,0x0010
-6             ABSOLUTE  00003564     bclr        d20,d0
-5             ABSOLUTE  00003578     addis       r4,r0,0x0001
-4             ABSOLUTE  0000357C     addi        r4,r4,0x94c8
-3             ABSOLUTE  00003580     addis       r5,r0,0x0001
-2             ABSOLUTE  00003584     addi        r5,r5,0x94d0
-1             ABSOLUTE  00003588     bl          0x000013e8
0              ABSOLUTE  000013E8     stwu        r1,0xfff8(r1)
1              ABSOLUTE  0000BFB0     mem write  0x0000bfb8
2              ABSOLUTE  000013EC     mfspr       r0,d8
3              ABSOLUTE  000013F0     stw         r0,0x000c(r1)
4              ABSOLUTE  0000BFBC     mem write  0x0000358c
5              ABSOLUTE  000013F4     bl          0x00001030
6              ABSOLUTE  00001030     stwu        r1,0xfff8(r1)
7              ABSOLUTE  0000BFA8     mem write  0x0000bfb0
8              ABSOLUTE  00001034     mfspr       r0,d8
```

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

To set the opcode source preferences

Inverse Assembler Preferences Opcode Source Dialog



MPC821/860 (E2477A) Preferences
File In<1>:Frame 10:Slot E:MPC821/860

Retrieve Opcode From

● Motorola S-Record

Filename: /logic/configs_test/marco/powerQUICC

S-Record Image Relocation

○ Enable/Disable Image Relocation

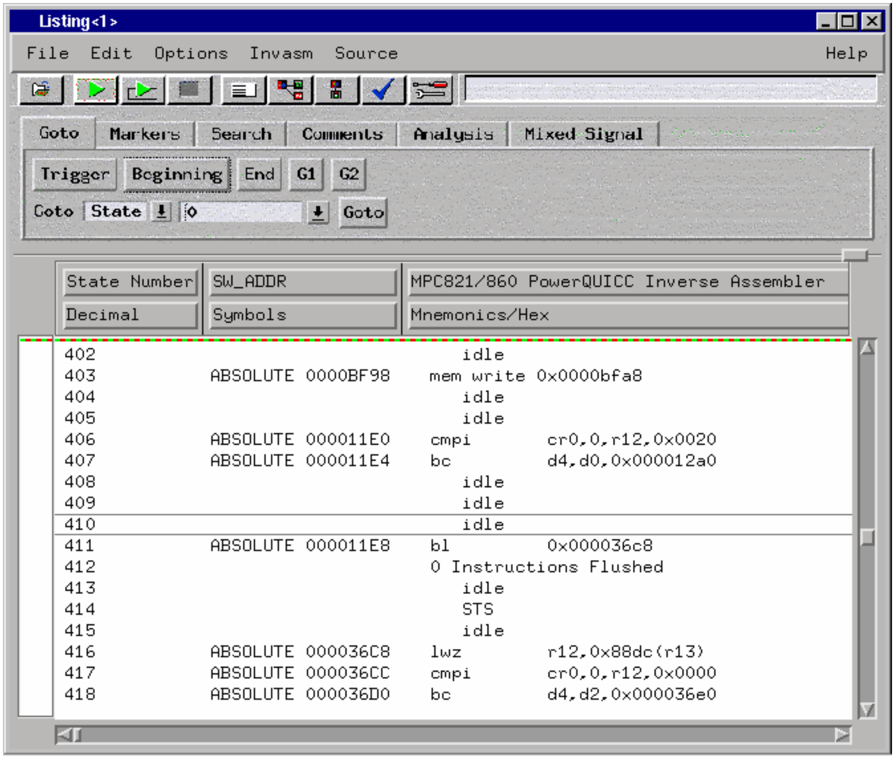
S-Record file base address: 0x00001000

Relocated base address:

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

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

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



```
State Number   SW_ADDR                MPC821/860 PowerQUICC Inverse Assembler
Decimal        Symbols                Mnemonics/Hex
402                                        idle
403            ABSOLUTE 0000BF98      mem write 0x0000bfa8
404                                        idle
405                                        idle
406            ABSOLUTE 000011E0      cmpi        cr0,0,r12,0x0020
407            ABSOLUTE 000011E4      bc          d4,d0,0x000012a0
408                                        idle
409                                        idle
410                                        idle
411            ABSOLUTE 000011E8      bl          0x000036c8
412                                   0 Instructions Flushed
413                                        idle
414                                        STS
415                                        idle
416            ABSOLUTE 000036C8      lwz         r12,0x88dc(r13)
417            ABSOLUTE 000036CC      cmpi        cr0,0,r12,0x0000
418            ABSOLUTE 000036D0      bc          d4,d2,0x000036e0
```

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

Symbols

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

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

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

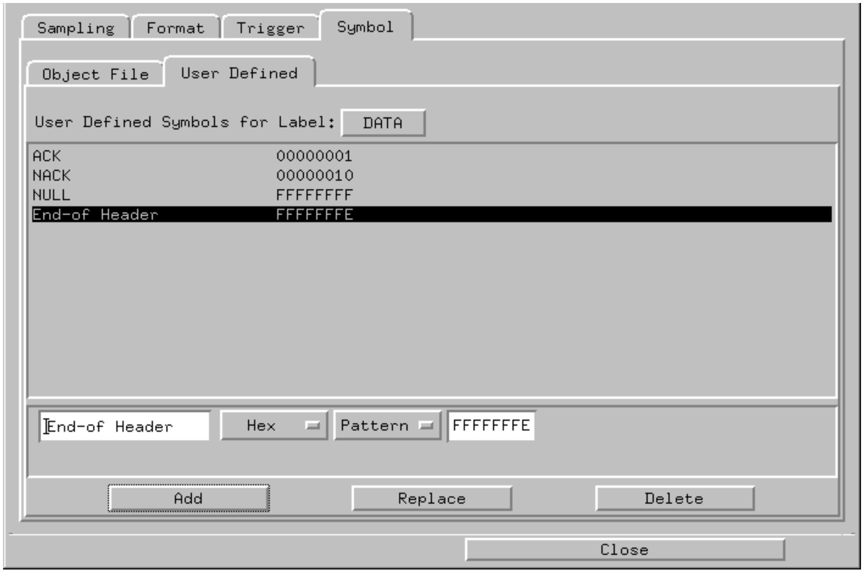
When source file line number symbols are downloaded to the logic analyzer, you can set up triggers on source lines. The B4620B Source Correlation Tool Set also lets you display the high-level source code associated with captured data.

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

The **User Defined** dialog is shown in the next figure. Use this dialog to create your own symbols.

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

User Defined Dialog



User Defined Symbols for Label: DATA

ACK	00000001
NACK	00000010
NULL	FFFFFFFF
End-of Header	FFFFFFFE

End-of Header | Hex | Pattern | FFFFFFFE

Predefined MPC8XX Symbols

If you are using an analysis probe for the MPC8XX microprocessor, the logic analyzer configuration files include predefined symbols. These symbols appear along with other user-defined symbols. Some of these symbols make several of the STAT fields easier to interpret. The following table lists the symbol descriptions.

Symbol Description

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

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

Note: The VF symbols display the proper order of these pins. These symbols compensate for the MPC8XX multiplexing scheme.

Object File Symbols

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

Requirements

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

An accurate bus trace

Typically, an Agilent Technologies analysis probe is used to capture MPC8XX microprocessor data. However, it is also possible to design connections for the logic analyzer into a prototype target system. Refer to the previous chapters on analysis probes and designing connections for custom probing.

An inverse assembler

Typically, the inverse assembler software is included with Agilent Technologies analysis probes, but it can also be purchased separately when custom probing connections are designed into a prototype target system. The MPC8XX 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 MPC8XX inverse assembly.

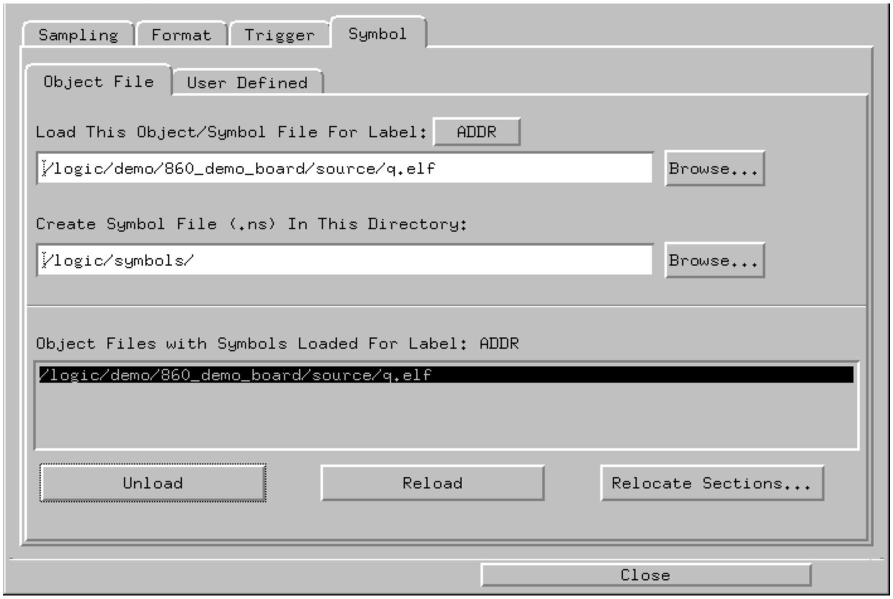
A symbol file

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 203).

To use object file symbols in the 16600/700

To load symbols in the 16600/700-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.

Symbol Dialog



Load This Object/Symbol File For Label: ADDR

/logic/demo/860_demo_board/source/q.elf

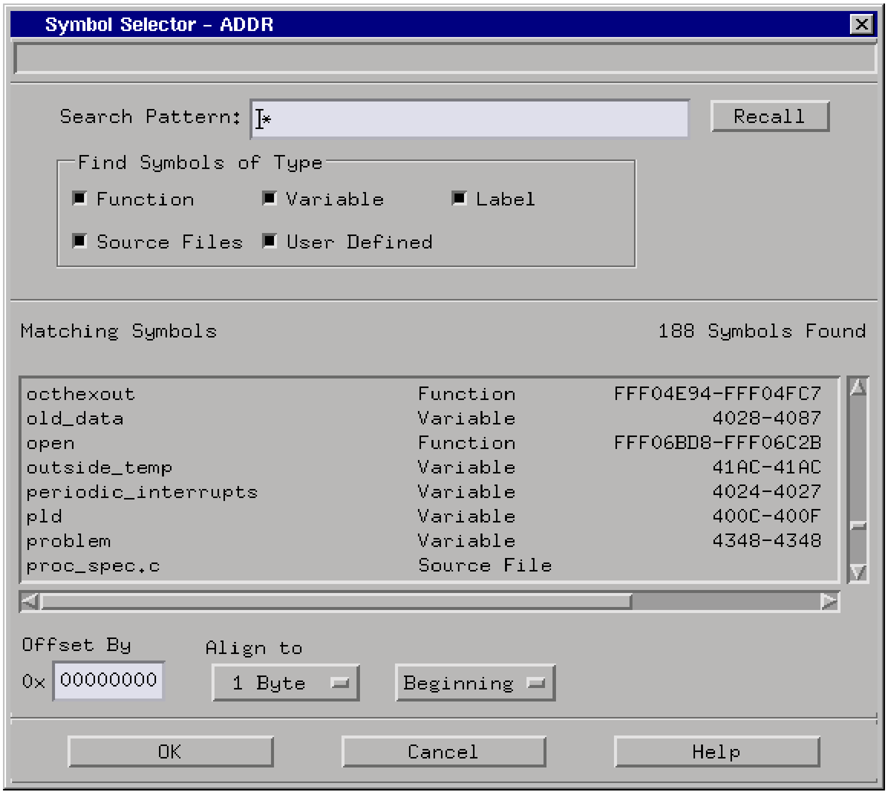
Create Symbol File (.ns) In This Directory:

/logic/symbols/

Object Files with Symbols Loaded For Label: ADDR

/logic/demo/860_demo_board/source/q.elf

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



Symbol Selector - ADDR

Search Pattern: *

Find Symbols of Type

■ Function ■ Variable ■ Label

■ Source Files ■ User Defined

Matching Symbols 188 Symbols Found

octhexout	Function	FFF04E94-FFF04FC7
old_data	Variable	4028-4087
open	Function	FFF06BD8-FFF06C2B
outside_temp	Variable	41AC-41AC
periodic_interrupts	Variable	4024-4027
pld	Variable	400C-400F
problem	Variable	4348-4348
proc_spec.c	Source File	

Offset By 0x 00000000

Align to 1 Byte Beginning

If your language tool is not one of those listed on page 115, you can create a symbol file in the General-Purpose ASCII (GPA) file format (refer to the “General-Purpose ASCII (GPA) File Format” chapter).

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

Compilers for the MPC8XX

The following MPC8XX compilers and their ELF/DWARF format object files can be used with Agilent Technologies logic analyzers and the Agilent Technologies 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 Agilent Technologies sales engineer to find out if there are other compilers for the MPC8XX microprocessor that can be used with Agilent Technologies logic analyzers.

Diab Data Compiler Options

The following options should be used:

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

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

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

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

Green Hills Compiler Options

The following options should be used:

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

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

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

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

Microtec Research Inc. Compiler Options

The following options should be used:

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

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

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

Using Labels

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

Labeling Conventions

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

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

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

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

Status bit labels

Each bit of the STAT label is described in the table below. The inverse assembler uses STAT bits BR, BG, BB, TS, TA, TEA, R/W, BURST, BI, and TSIZ. The signal-to-connector tables beginning on page 48 list all the MPC8XX signals probed and their corresponding analyzer channels.

Status Bit Description

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

To enable/disable the instruction cache on the MPC8XX

NOTE: It is not necessary to disable the instruction cache when using the cache-on trace reconstruction feature of the inverse assembler.

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

To disable the cache with the emulation module:

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

Register values for controlling the cache

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

To disable the cache with code:

- Disable the instruction cache with the following code:

```
mfspr    r3, hid0
rlwinm   r3, r3, 0, 17, 15   # clear bit 16 (ICE)
mtspr    hid0, r3
isync
```

- To also disable the data cache use:

```
mfspr    r3, hid0
rlwinm   r3, r3, 0, 18, 15   # clear ICE and DCE
mtspr    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
```

To enable the cache with code:

- Enable the instruction cache with the following code:

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

6

Configuring the 1660/1670/16500B/C-Series Logic Analyzer

Configuring 1660/1670/16500B/C-Series Logic Analysis Systems

The information in this chapter is specific to systems using 1660/1670 or 16500B/C-series logic analyzers. For systems using 16600/700-series logic analysis systems, see Chapter 5, “Configuring the 16600/700-Series Logic Analyzer,” beginning on page 87.

To load configuration files and the inverse assembler—1660/1670/16500B/C-series logic analysis systems

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

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

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

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

   Your configuration file choice depends on which analyzer you are using. See table on following page.
6. Execute the load operation to load the file into the logic analyzer.

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

To install software on other logic analyzers

Consult the documentation for your logic analyzer for details.

Checking pin assignments

When using 1660/1670/16500B/C-series logic analyzers with the E2476B analysis probe, it may be necessary to adjust the configuration. The channel assignments for four pins were changed when the E2476B analysis probe superceeded the E2476A analysis probe. These are listed in the table below. Check (and change if necessary) the configuration using your logic analyzer's format menu.

<table>
<tr><th colspan="6">MICTOR Connector J1</th></tr>
<tr><th>J1 Pin</th><th>BGA pin</th><th>MPC860 Signal</th><th>J1 Pin</th><th>BGA Pin</th><th>MPC860 Signal</th></tr>
<tr><td>5</td><td>B9</td><td>TSIZ0/$\overline{\text{REG}}$</td><td>6</td><td>C9</td><td>TSIZ1</td></tr>
<tr><th colspan="3">Even Cable</th><th colspan="3">Odd Cable</th></tr>
<tr><th colspan="3">Logic Analyzer Pod 2</th><th colspan="3">Logic Analyzer Pod 1</th></tr>
</table>

<table>
<tr><th colspan="6">MICTOR Connector J2</th></tr>
<tr><th>J2 Pin</th><th>BGA pin</th><th>MPC860 Signal</th><th>J2 Pin</th><th>BGA Pin</th><th>MPC860 Signal</th></tr>
<tr><td>5</td><td>W3</td><td>CLKOUT</td><td>6</td><td>L1</td><td>$\overline{\text{STS}}$</td></tr>
<tr><th colspan="3">Even Cable</th><th colspan="3">Odd Cable</th></tr>
<tr><th colspan="3">Logic Analyzer Pod 4</th><th colspan="3">Logic Analyzer Pod 3</th></tr>
</table>

Logic Analyzer Configuration Files for 16500/1660/1670-Series Logic Analyzers

Analyzer Model	Analyzer Description (modules only)	Configuration File for Inverse Assembly (I860E)	Configuration File for Cache-on Execution Tracking (I860ET)
16550A (1 card)	100 MHz STATE 250 MHz TIMING	CM860F6	CM860ET1
16550A (2 card)	100 MHz STATE 250 MHz TIMING	CM860F12	CM860ET2
16554A (2 card)	0.5 M SAMPLE 70/250 MHz LA	CM860M8	CM860ET3
16555A/D (2 card)	1.0 M SAMPLE 110/250 MHz LA	CM860M8	CM860ET3
16556A/D (2 card)	1.0 M SAMPLE 100/400 MHz LA	CM860M8	CM860ET3
16557D (2 card)	2.0 M SAMPLE 135/250 MHz LA	CM860M8	CM860ET3
16554A (3 card)	0.5 M SAMPLE 70/250 MHz LA	CM860M12	CM860ET4
16555A/D (3 card)	1.0 M SAMPLE 110/250 MHz LA	CM860M12	CM860ET4
16556A/D (3 card)	1.0 M SAMPLE 100/400 MHz LA	CM860M12	CM860ET4
16557D (3 card)	2.0 M SAMPLE 135/250 MHz LA	CM860M12	CM860ET4
1660A/AS/C/CS/E/ES/EP	na	CM860M8	na
1661A/AS/C/CS/E/ES/EP	na	CM860F6	na
1670A/D/E	na	CM860M8	na
1671A/D/E	na	CM860F6	na

Modes of Analysis

The inverse assembler offers two modes of analysis for MPC8XX microprocessors: inverse assembly, and cache-on execution tracking. These two modes cannot operate at the same time. To change from one analysis mode to the other, load the appropriate configuration file.

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

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

Inverse assembly analysis

The inverse assembler lets you obtain displays of MPC8XX operations in MPC8XX mnemonics. Information that is processed in cache is not visible to the inverse assembler, and cannot be decoded.

Cache-on execution tracker

The cache-on execution tracker lets you track instructions executed in the cache. However, the data is not inverse assembled into MPC8XX mnemonics. The logic analyzer displays the data in instruction-type format.

The cache-on execution tracker requires the Agilent Technologies 16600A/700A-series logic analyzers, or the Agilent Technologies 16505A prototype analyzer for the Agilent Technologies 16500B/C mainframe. See pages page 27 and page 28 for the logic analyzers and logic analyzer software version requirements for using the cache-on execution tracker.

The cache-on execution tracker provides much more information when used together with the Agilent Technologies B4620B Source Correlation Tool Set. Source correlation performs a correlation of the addresses from cache with the high-level code execution.

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

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

Changing the Acquisition Mode

The E2476B analysis probe can be used in three different operating modes: State-per-ack, State-per-clock, or Timing. The E2477A inverse assembler/execution tracker software can be used for State-per-ack and State-per-clock analysis. The following sections describe these operating modes and how to configure the logic analyzer for each mode.

State-per-ack mode

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

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

State-per-clock mode

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

Timing mode

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

Configuring Signal Assignments

Bus signal conventions

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

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

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

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

Format menu

This section describes the organization of MPC8XX signals in the logic analyzer's Format menu.

The configuration software sets up the analyzer format menu to display either six, eight, or twelve pods of data, depending on the analyzer. The figure on the following page shows the **Format** menu for the MPC8XX.

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.

Status Encoding

Each of the bits of the STAT label is described in the table below. The inverse assembler uses STAT bits BR, BG, BB, TS, TA, TEA, R/W, BURST, BI, and TSIZ. The signal-to-connector tables in the “Hardware Reference” chapter list all the MPC8XX signals probed and their corresponding analyzer channels.

Status Bit Description

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

Predefined Logic Analyzer Symbols

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

Symbol Description

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

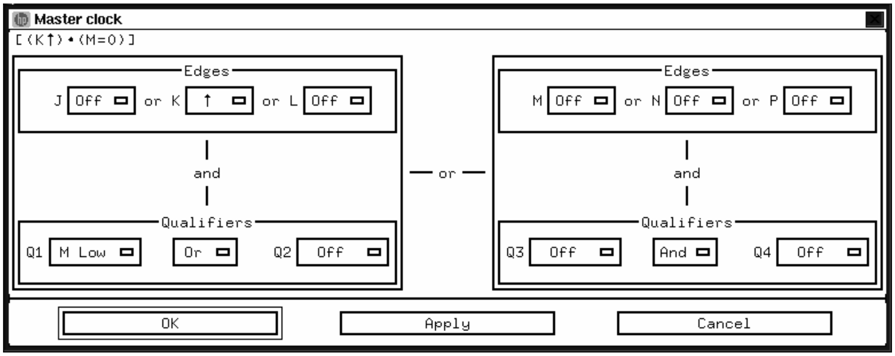
Label	Symbol	Encoding
BB	(blank)	1
	bb	0
BDIP	(blank)	1
	bdip	0
STS	(blank)	1
	sts	0
AT0	CPM	1
	CPU	0
AT2	data	1
	instr	0
AT3	trace	1
	resrv	0
TS	(blank)	1
	ts	0
VF	000	000
(see note)	010	001
	100	010
	110	011
	001	100
	011	101
	101	110
	111	111

Note: The VF symbols display the proper order of these pins. These symbols compensate for the MPC8XX multiplexing scheme.

To qualify stored data

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

Storage Qualification



Master clock

[(K↑)•(M=0)]

Edges: J Off or K ↑ or L Off

and

Qualifiers: Q1 M Low Or Q2 Off

or

Edges: M Off or N Off or P Off

and

Qualifiers: Q3 Off And Q4 Off

The memory controller in the MPC8XX will allow data to be latched on the falling edge of the system clock (K falling). If your data appears incorrect, check the memory controller registers for this type of configuration.

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

Burst Mode

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

Using the Inverse Assembler

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

Before the inverse assembler will correctly disassemble information captured on the MPC8XX address bus, you must:

- If you are not using the cache-on execution tracker, make sure the target system's cache has been disabled.
- Set up the inverse assembler preferences to give the inverse assembler information about the MPC8XX memory controller and memory bank setup.

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

To disable the instruction cache on the MPC8XX

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

To disable the cache with the emulation module:

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

Register values for controlling the cache

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

To disable the cache with code:

- Disable the cache with the following code:

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

- To also disable the data cache use:

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

- To invalidate and disable the caches use:

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

To use the Invasm menu

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

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

Load

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

Filter

See “Display filtering” on page 160 for information on filtering a listing.

Inverse Assembler Preferences

The MPC8XX does not always provide a signal to distinguish instruction reads from data reads. Also, the width of the memory being accessed (port size) is not indicated on external signals. For accurate inverse assembly the inverse assembler must be configured to match the memory controller configuration on the microprocessor.

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

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

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

Why the configuration is necessary

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

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

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

Finding memory bank information using a debugger

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

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

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

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

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

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

Finding memory bank information at compile time

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

Example

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

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

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

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

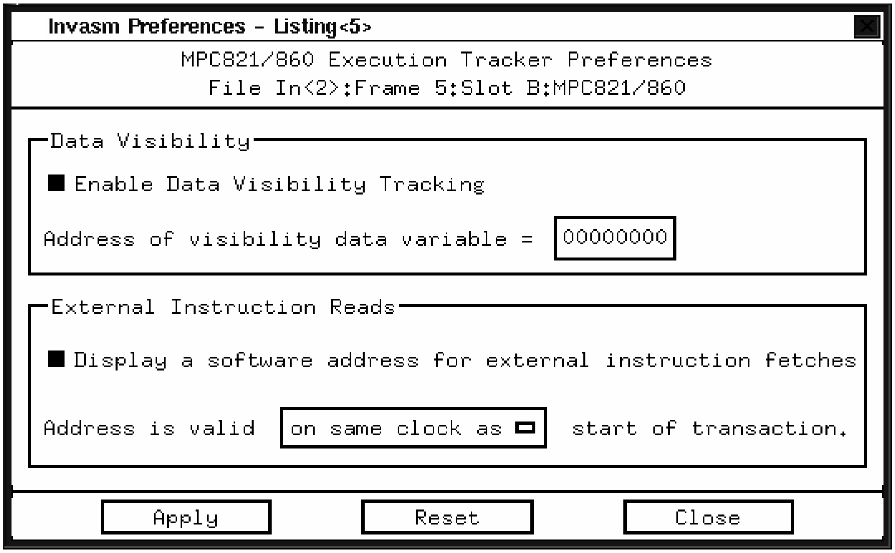
Options

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

Execution Tracker Preferences

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

Execution Tracker Preferences Menu



Invasm Preferences - Listing<5>

MPC821/860 Execution Tracker Preferences
File In<2>:Frame 5:Slot B:MPC821/860

Data Visibility

■ Enable Data Visibility Tracking

Address of visibility data variable = 00000000

External Instruction Reads

■ Display a software address for external instruction fetches

Address is valid on same clock as start of transaction.

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

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

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

To enable Program Trace cycles

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

- Enable the Program Trace cycles with the following code:

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

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

Symbols

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

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

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

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

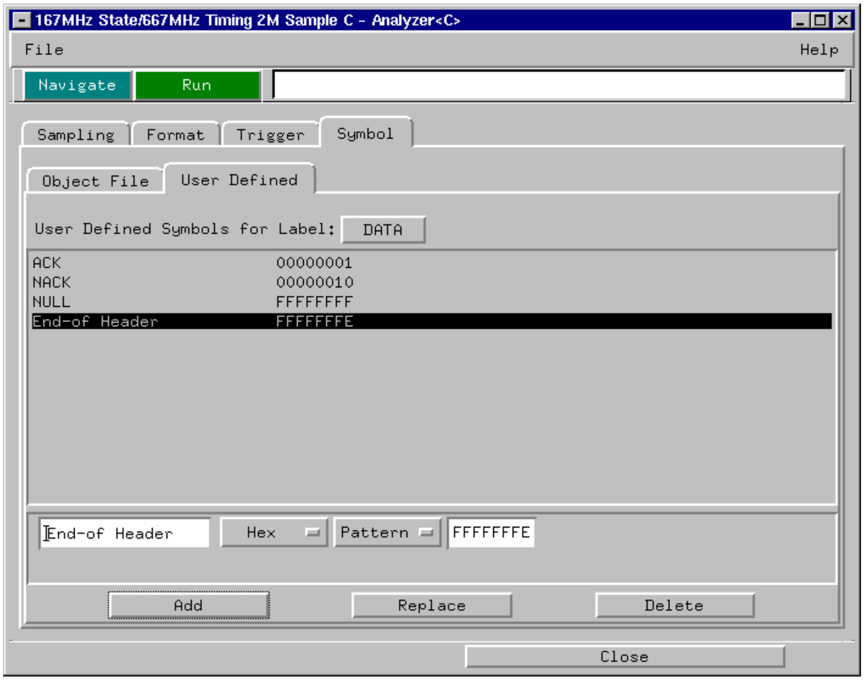
After describing user-defined symbols, the rest of this chapter describes the requirements and considerations for displaying object file symbols and source code for MPC8XX address values captured by a logic analyzer.

User-Defined Symbols

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

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

User-Defined Symbols Dialog



User Defined Symbols for Label: DATA

ACK	00000001
NACK	00000010
NULL	FFFFFFFF
End-of Header	FFFFFFFE

End-of Header | Hex | Pattern | FFFFFFFE

Predefined MPC8XX Symbols

If you are using an analysis probe for the MPC8XX microprocessor, the logic analyzer configuration files include predefined symbols.

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

The predefined MPC8XX symbols are listed on page 134.

To use object file symbols in the 1660/1670/ 16500B/C

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

Requirements

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

An accurate bus trace

Typically, an Agilent Technologies analysis probe is used to capture MPC8XX microprocessor data. However, it is also possible to design connections for the logic analyzer into a prototype target system. Refer to the previous chapters on analysis probes and designing connections for custom probing.

An inverse assembler

Typically, the inverse assembler software is included with Agilent Technologies analysis probes, but it can also be purchased separately when custom probing connections are designed into a prototype target system. The MPC8XX 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 MPC8XX 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 Chapter 11, “General-Purpose ASCII (GPA) Symbol File Format,” beginning on page 203).

See Also

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

7

Capturing Processor Execution

The normal steps in using the logic analyzer are:

1. Configure the logic analyzer.
2. Format labels for the logic analyzer channels (that is, mapping logic analyzer channels to target system signal names).
3. Load symbols from the program's object file.
4. Set up the trigger, and run the measurement.
5. Display the captured data.

With the MPC8XX inverse assembler, the logic analyzer is configured, and labels are created (formatted) for the logic analysis channels when configuration files are loaded. See "Configuring the 16600/700-Series Logic Analyzer" on page 87 or "Configuring the 1660/1670/16500B/C-Series Logic Analyzer" on page 123, depending on which analyzer you are using.

You can load program object file symbols into the logic analyzer when configuring it (see "To use object file symbols in the 16600/700" on page 113 or "To use object file symbols in the 1660/1670/16500B/C" on page 146).

This chapter describes setting up logic analyzer triggers when using the Agilent Technologies E9584A Option 001 inverse assembler and the Agilent Technologies B4620B source correlation tool set.

See Chapter 8, "Displaying Captured Processor Execution," beginning on page 155 for information on displaying captured data.

Trigger sequence

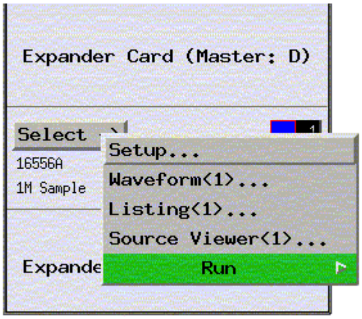
The Trigger sequence is set up by the software to store all states.

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

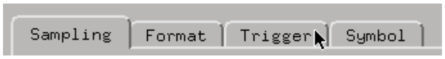
Setting Up Logic Analyzer Triggers

To set up logic analyzer triggers

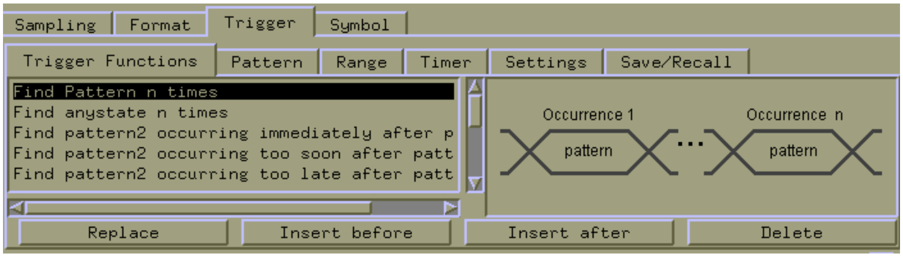
1 Open the logic analyzer's Setup window.



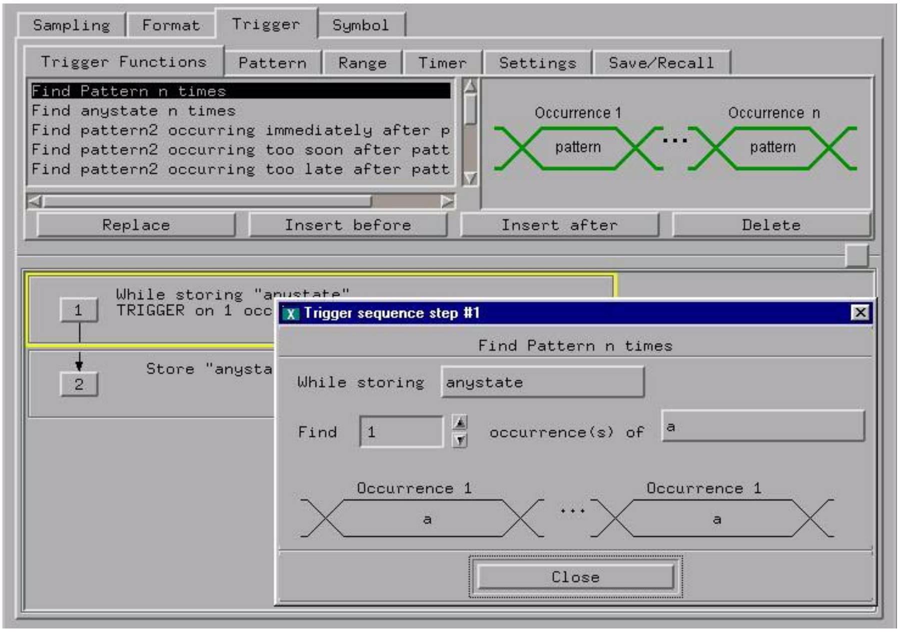
2 Select the Trigger tab.



3 Select the trigger function that will be used in the logic analysis measurement.



4 Set up the trigger sequence.



Find Pattern n times
Find anystate n times
Find pattern2 occurring immediately after p
Find pattern2 occurring too soon after patt
Find pattern2 occurring too late after patt

1 While storing "anystate"
TRIGGER on 1 occ

2 Store "anysta

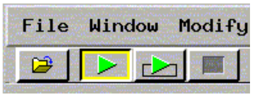
Find Pattern n times

While storing anystate

Find 1 occurrence(s) of a

Occurrence 1 a ... Occurrence 1 a

5 Run the measurement.



See Also The Agilent Technologies 16600A/700-series logic analysis system's on-line help for more information on setting up logic analyzer triggers.

Triggering on Symbols and Source Code

When setting up trigger specifications to capture MPC8XX execution:

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

Using the Address Offset

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

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

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

Using Storage Qualification

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

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

NOTE: Storage qualification can not be used when cache-on trace reconstruction or cache-on execution tracking are enabled. These features require that all processor cycles are stored. See "Cache-on trace reconstruction" on page 95 or "Cache-on execution tracker" on page 128.

To qualify stored data

The configuration file sets up the logic analyzer clock.

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

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

The memory controller in the MPC8XX will allow data to be latched on the falling edge of the system clock (K falling). If your data appears incorrect, check the processor’s memory controller registers to ensure that the clocking configurations of the processor and the logic analyzer match.

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

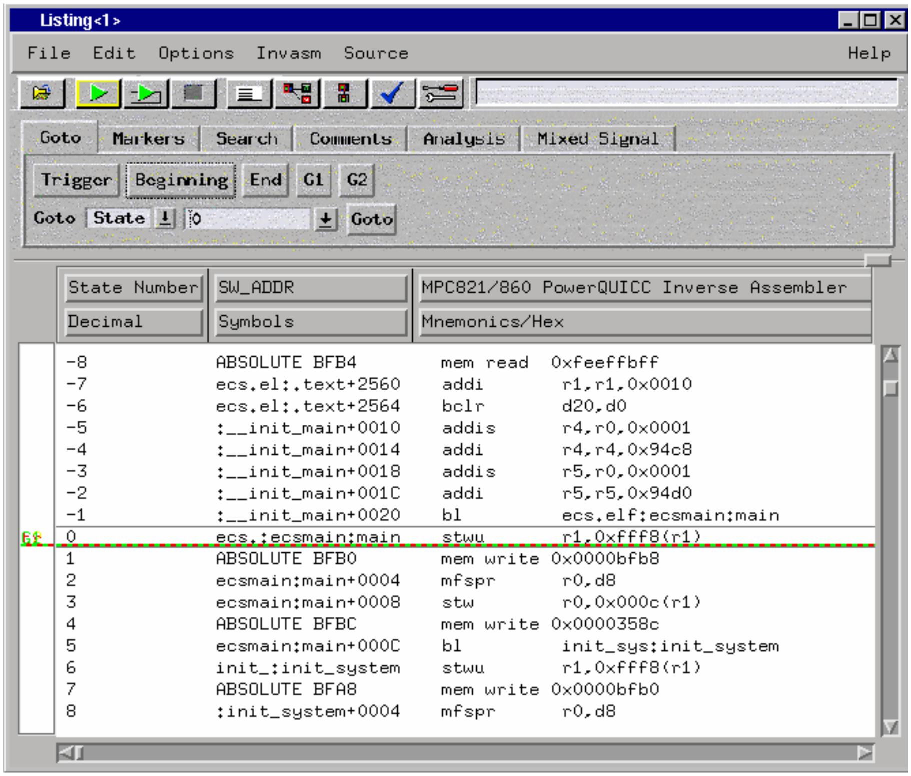
8

Displaying Captured Processor Execution

To display captured state data

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

Listing window



```
State Number   SW_ADDR              MPC821/860 PowerQUICC Inverse Assembler
Decimal        Symbols              Mnemonics/Hex
-8             ABSOLUTE BFB4         mem read  0xfeeffbff
-7             ecs.el:.text+2560     addi       r1,r1,0x0010
-6             ecs.el:.text+2564     bclr       d20,d0
-5             :__init_main+0010     addis      r4,r0,0x0001
-4             :__init_main+0014     addi       r4,r4,0x94c8
-3             :__init_main+0018     addis      r5,r0,0x0001
-2             :__init_main+001C     addi       r5,r5,0x94d0
-1             :__init_main+0020     bl         ecs.elf:ecsmain:main
0              ecs.:ecsmain:main     stwu       r1,0xfff8(r1)
1              ABSOLUTE BFB0         mem write 0x0000bfb8
2              ecsmain:main+0004     mfspr      r0,d8
3              ecsmain:main+0008     stw        r0,0x000c(r1)
4              ABSOLUTE BFBC         mem write 0x0000358c
5              ecsmain:main+000C     bl         init_sys:init_system
6              init_:init_system     stwu       r1,0xfff8(r1)
7              ABSOLUTE BFA8         mem write 0x0000bfb0
8              :init_system+0004     mfspr      r0,d8
```

The column on the left of the inverse assembly data display is the least significant hexadecimal digit of the current address. On the Agilent Technologies 16600A/700-series logic analysis systems, the entire synthesized address appears under the label “SW_ADDR”. You can observe the actual address bits presented by the MPC8XX 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 MPC8XX. 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.

Extended Mnemonics

PowerPC assemblers support a number of extended mnemonics for some popular assembly language instructions as described in the MPC8XX User's Manual. The disassembler supports the following extensions:

- Conditional traps and branches decode the condition mnemonically when possible. For some conditions which have no conventional mnemonics (for example, “signed less than or unsigned greater than”), the condition field is displayed in binary.
- The L bit is omitted as a compare operand. Instead, compares are decoded as “cmpw” (or “?cmpd”).

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

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

SW_ADDR Label

When an Agilent Technologies 16600A/700-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 Agilent Technologies B4620B Source Correlation Tool Set.

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.

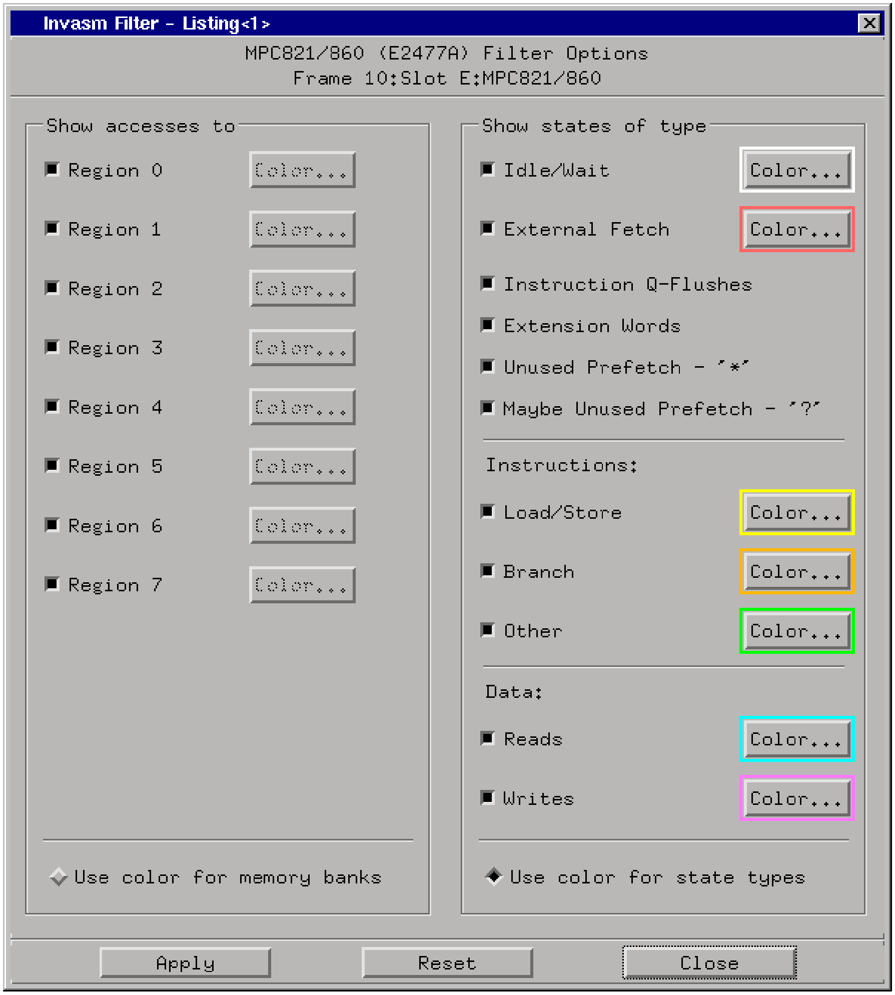
Second, you can isolate particular operations by suppressing all other operations. For example, you can show branches, with all other states suppressed, allowing quick analysis of branch instructions.

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

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

The following figure shows the inverse assembler filter dialog.

Inverse Assembler Filter Dialog



Invasm Filter - Listing<1>

MPC821/860 (E2477A) Filter Options
Frame 10:Slot E:MPC821/860

Show accesses to

■ Region 0
■ Region 1
■ Region 2
■ Region 3
■ Region 4
■ Region 5
■ Region 6
■ Region 7

◇ Use color for memory banks

Show states of type

■ Idle/Wait
■ External Fetch
■ Instruction Q-Flushes
■ Extension Words
■ Unused Prefetch - '*'
■ Maybe Unused Prefetch - '?'

Instructions:

■ Load/Store
■ Branch
■ Other

Data:

■ Reads
■ Writes

◆ Use color for state types

Displaying Source Code

The B4620B Source Correlation Tool Set lets you:

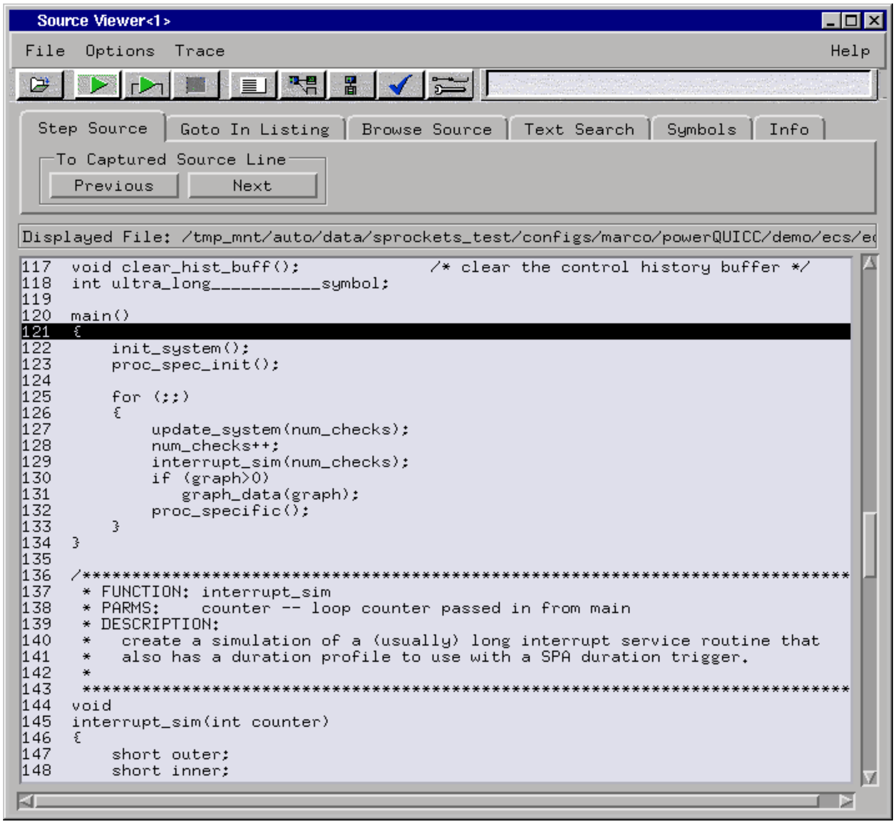
- View the high-level source code associated with captured data.
- Set up triggers based on source code.

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

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

The first time you display the Source Viewer window, it will probably be blank. To see the source code click the Browse Source tab and choose a file to display. To see source code that corresponds to a particular state in the listing, select that state in the Listing window. The figure below shows execution of data that is correlated to the data shown on page 156.

Source Correlation Tool Set Data



Displayed File: /tmp_mnt/auto/data/sprockets_test/configs/marco/powerQUICC/demo/ecs/e

```
void clear_hist_buff();             /* clear the control history buffer */
int ultra_long___________symbol;

main()
{
    init_system();
    proc_spec_init();

    for (;;)
    {
        update_system(num_checks);
        num_checks++;
        interrupt_sim(num_checks);
        if (graph>0)
           graph_data(graph);
        proc_specific();
    }
}

/******************************************************************************
 * FUNCTION: interrupt_sim
 * PARMS:     counter -- loop counter passed in from main
 * DESCRIPTION:
 *   create a simulation of a (usually) long interrupt service routine that
 *   also has a duration profile to use with a SPA duration trigger.
 *
 ******************************************************************************
void
interrupt_sim(int counter)
{
    short outer;
    short inner;
```

If you purchased an emulation solution, the B4620B Source Correlation Tool Set was included. Otherwise, the source correlation tool set is available as an add-on product for the 16600A/700-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 Agilent Technologies E2476B analysis probe meets this requirement.

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

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

- Object file symbols.

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

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

- Access to source code files.

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

Inverse assembler generated PC (software address) label

In the 16600/700-series logic analysis system, the MPC8XX inverse assembler generates a “SW_ADDR” label. The SW_ADDR 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 Agilent Technologies 16600/16700-series logic analysis system perform a pattern search on the SW_ADDR 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 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 Agilent Technologies logic analyzer networking must be compatible with the user's network environment. Agilent Technologies logic analyzers currently support Ethernet networks running a TCP/IP protocol and support ftp, telnet, web, NFS client/server and X-Window client/server applications. Some PC networks may require extensions to the normal LAN protocols to support the TCP/IP protocol and/or these networking applications. Users should contact their LAN system administrators to help setup the logic analyzer on their network.

Source File Version Control

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

9

Troubleshooting the Logic Analyzer

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 Agilent Technologies 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” on page 173 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 Agilent Technologies 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.

Inverse Assembler Problems

This section lists problems that you might encounter while using the inverse assembler.

When you obtain incorrect inverse assembly results, it may be unclear whether the problem is in the analysis probe or in your target system. If you follow the suggestions in this section to ensure that you are using the analysis probe and inverse assembler correctly, you can proceed with confidence in debugging your target system.

No inverse assembly or incorrect inverse assembly

This problem may be due to incorrect synchronization, modified configuration, incorrect connections, or a hardware problem in the target system. A locked status line can cause incorrect or incomplete inverse assembly.

❑ Ensure that each logic analyzer pod is connected to the correct analysis probe connector.

There is not always a one-to-one correspondence between analyzer pod numbers and analysis probe cable numbers. Analysis Probes must supply address (ADDR), data (DATA), and status (STAT) information to the analyzer in a predefined order. The cable connections for each analysis probe are often altered to support that need. Thus, one analysis probe might require that you connect cable 2 to analyzer pod 2, while another will require you to connect cable 5 to analyzer pod 2. See Chapter 3 for connection information.

❑ Check the activity indicators for status lines locked in a high or low state.

❑ Verify that the STAT, DATA, and ADDR format labels have not been modified from their default values.

These labels must remain as they are configured by the configuration file. Do not change the names of these labels or the bit assignments within the labels. Some analysis probes also require other data labels. See page 111 or page 125 for more information.

❏ Verify that all microprocessor caches and memory managers have been disabled.

NOTE: This is only necessary if cache-on trace reconstruction and cache-on execution tracker are not used.

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.

See “To enable/disable the instruction cache on the MPC8XX” on page 120 or “To disable the instruction cache on the MPC8XX” on page 137.

❏ 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 MPC8XX 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 “To install software from CD-ROM (16600/700-series logic analysis systems)” on page 60 to reinstall the software if you have renamed or deleted the inverse assembler.

If the inverse assembler cannot determine cycle sizes

Agilent Technologies 16600A and 16700 logic analysis systems only

The processor sometimes fails to put correct information on the bus.

- In the Listing window, select the Preferences menu.

Enter each memory bank address into the appropriate field.

Intermodule Measurement Problems

Some problems occur only when you are trying to make a measurement involving multiple modules.

An event wasn't captured by one of the modules

If you are trying to capture an event that occurs very shortly after the event that arms one of the measurement modules, it may be missed due to internal analyzer delays. For example, suppose you set an oscilloscope module to trigger upon receiving a trigger signal from the logic analyzer because you are trying to capture a pulse that occurs right after the analyzer's trigger state. If the pulse occurs too soon after the analyzer's trigger state, the oscilloscope will miss the pulse.

❏ Adjust the skew in the Intermodule menu.

You may be able to specify a skew value that enables the event to be captured.

❏ Change the trigger specification for modules upstream of the one with the problem.

If you are using a logic analyzer to trigger an oscilloscope module, try specifying a trigger state one state before the one you are using. This may be more difficult than working with the skew because the prior state may occur more often and not always be related to the event you are trying to capture with the oscilloscope.

Analysis Probe Messages

This section lists some of the messages that the analyzer displays when it encounters a problem.

". . . Inverse Assembler Not Found"

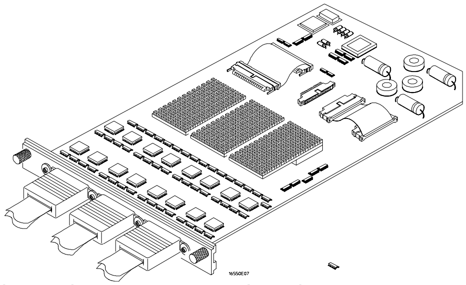
This error occurs if you rename or delete the inverse assembler file that is attached to the configuration file.

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

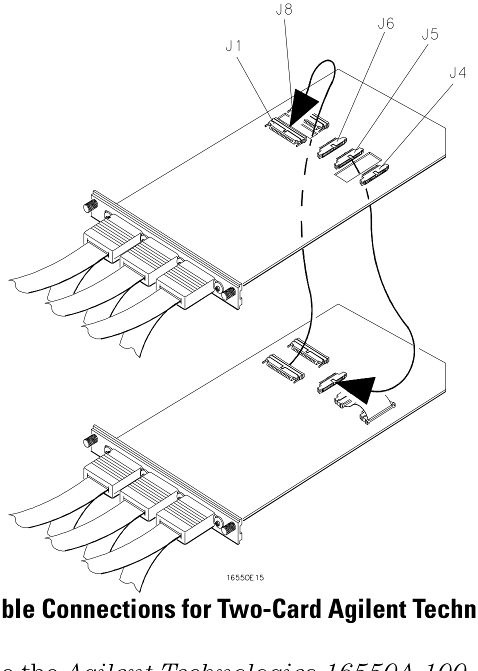
- For Agilent Technologies 16600A/700-series logic analysis systems it should be in /logic/ia.
- For other logic analyzers it should be in the same directory as the configuration file.

“Measurement Initialization Error”

This error occurs when you have installed the cables incorrectly for one or two Agilent Technologies 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 Agilent Technologies 16550A Installations



Cable Connections for Two-Card Agilent Technologies 16550A Installations

See Also

See the *Agilent Technologies 16550A 100-MHz State/500-MHz Timing Logic Analyzer Service Guide* for more information.

“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 “To load configuration files (and the inverse assembler) from hard disk” on page 90 or “To load configuration files and the inverse assembler—1660/1670/16500B/C-series logic analysis systems” on page 125.

“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 4, “Probing the Target System,” beginning on page 63, to determine the proper connections.

“Time from Arm Greater Than 41.93 ms”

The Agilent Technologies 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 Agilent Technologies for Service

The repair strategy for this emulation solution is board replacement.

Exchange assemblies are available when a repairable assembly is returned to Agilent Technologies. 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 Agilent Technologies

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 Agilent Technologies sales office. Ask them for the address of the nearest Agilent Technologies service center.

3 Package the part and send it to the Agilent Technologies 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 Agilent Technologies.

The Agilent Technologies 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 Agilent Technologies 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 Agilent Technologies Sales Office for further information.

Analysis Probe Replaceable Parts

Agilent Part Number	Description
E5346A	High-density Cable
E2476-66502	Analysis Probe Circuit Board
E2476-87602	BGA Extender (one pre-installed on the analysis probe board, and one supplied with the product for customer installation. Total of 2)
E2476-87606	Double Header, 19x19
E2476-87607	BGA Carrier
E5355A	BGA Probe Kit

Cleaning the Instrument

If the instrument requires cleaning:

1. Disconnect power from the instrument.
2. Clean the instrument using a soft cloth that has been moistened in a mixture of mild detergent and water.

   Make sure that the instrument is completely dry before reconnecting it to a power source.

10

Coordinating Logic Analysis with Processor Execution

This chapter describes how to use an analysis probe, an emulation module, and other features of your Agilent Technologies 16600A or 16700 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 Agilent Technologies 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 Agilent Technologies 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, select the Help icon in the logic analysis system window, then select "Measurement Examples."

Triggering the Emulation Module from the Analyzer

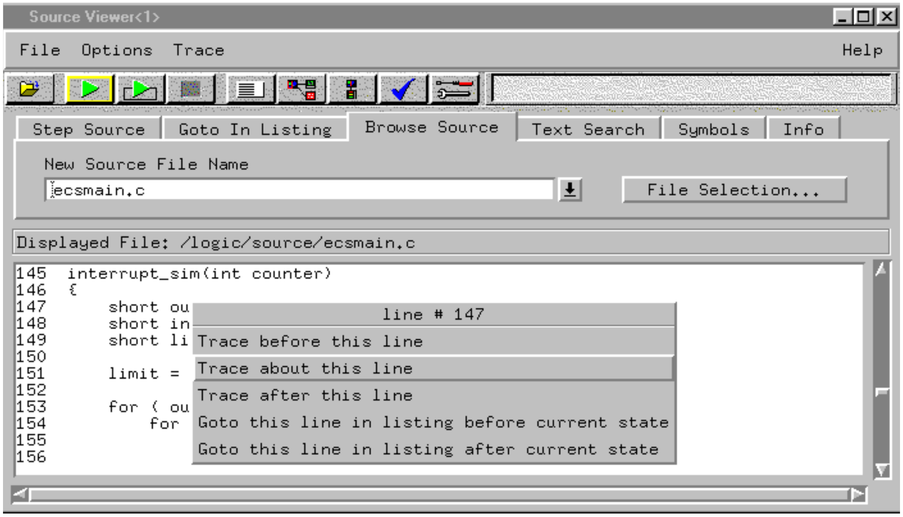
The logic analyzer may be used to signal the emulation module to stop (break) the target processor. This is done from either the Source Viewer window or the Intermodule window. If you are using the Agilent Technologies B4620B Source Correlation Tool Set, using the Source Viewer window is the easiest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

To stop the processor when the logic analyzer triggers on a line of source code (Source Viewer window)

If you have the Agilent Technologies B4620B Source Correlation Tool Set, you can easily stop the processor when a particular line of code is reached.

1. Select the logic analyzer module icon in the System window, and choose **Source Viewer...**.
2. In the Source Viewer window, select 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.



New Source File Name: ecsmain.c

Displayed File: /logic/source/ecsmain.c

```
145  interrupt_sim(int counter)
146  {
147      short ou
148      short in
149      short li
150
151      limit =
152
153      for ( ou
154          for
155
156
```

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

5 If your target system is not already running, select **Run** in the emulation Run Control window to start your target.

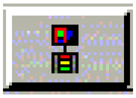
To stop the processor when the logic analyzer triggers (Intermodule window)

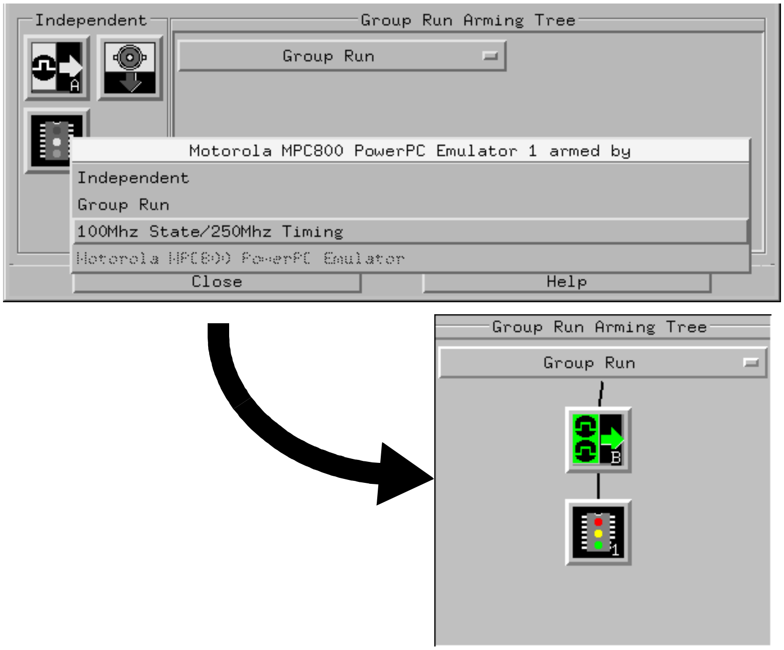
Use the Intermodule window if you do not have the Agilent Technologies B4620B Source Correlation Tool Set or if you need to use a more sophisticated trigger than is possible in the Source Viewer window.

1 Create a logic analyzer trigger.

2 Select the **Intermodule** icon in the System window.

3 In the Intermodule window, select 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 Select **Group Run** in the Source window (or other logic analyzer window).

5 If your target system is not already running, select **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 "To configure the processor clock speed" on page -209 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 select **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 selects **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

1. Set the sampling to **state mode** and the trigger condition to **Run until user stop**.
2. Set the trigger point (position) to **End**.
3. In a logic analyzer window, select **Run**.
4. In the Emulation Control Interface or debugger select **Run**.
5. When the target processor halts, select **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 194.

Before you trigger a logic analyzer (or another module) from the emulation module, you should understand a few things about the emulation module trigger:

The emulation module trigger signal

The trigger signal coming from the emulation module is an "In Background Debug Monitor" (In Monitor) signal. This may cause confusion because a variety of conditions could cause this signal and falsely trigger your analyzer.

The In Monitor trigger signal can be caused by:

- The most common method to generate the signal is to select **Run** and then select **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 select **Reset** and then select **Break**. Going from the reset state of the processor to the In Monitor state will generate the signal. Some processors that do not remain in reset will not generate an In Monitor signal in the reset to break transition.
- 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:

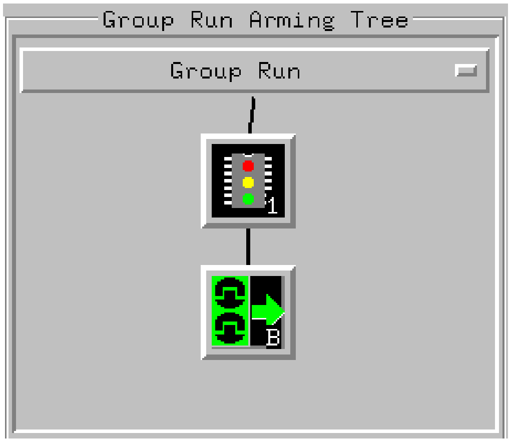
- If Group Run is armed from Port In and an emulation module is connected to Group Run, then any Port In signal will cause the emulation module to go into monitor. The Group Run button does not have to be selected for this to operate.
- If two emulation modules are connected together so that one triggers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



Selecting 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 select **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.

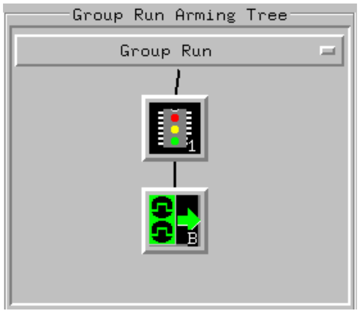
To trigger the analyzer when the processor halts - timing mode

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 Agilent Technologies 16600A/16700 logic analysis system with VisiTrigger. This measurement can also be set up using Agilent Technologies 16600A/16700 logic analysis systems without VisiTrigger, and Agilent Technologies 1660/1670/16500-series logic analysis systems.

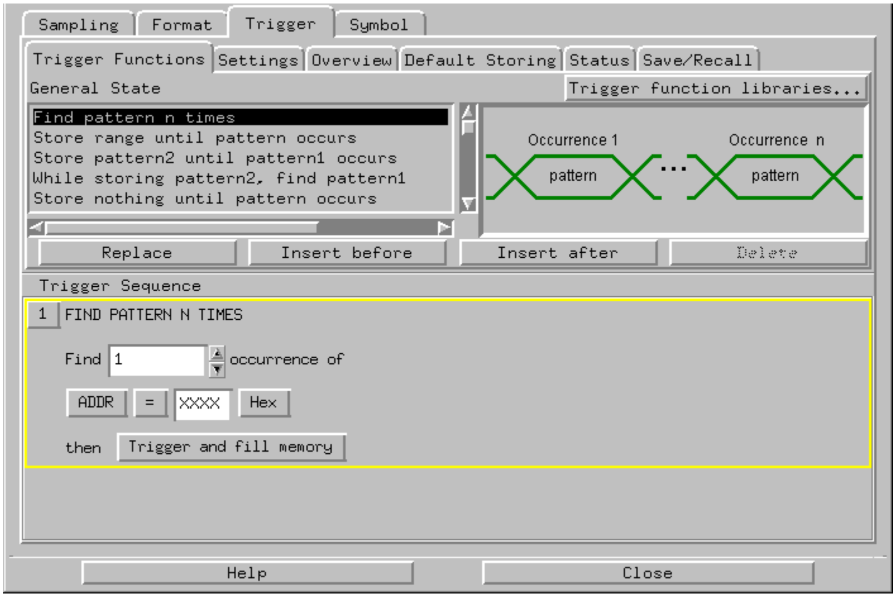
NOTE: If you only need state information leading up to a processor halt, and timing information is not important, use the procedure called "To capture a trace before the processor halts" on page 194. It is much simpler.

1 In the Intermodule window, select 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.

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



Find pattern n times
Store range until pattern occurs
Store pattern2 until pattern1 occurs
While storing pattern2, find pattern1
Store nothing until pattern occurs

Trigger Sequence

1 FIND PATTERN N TIMES

Find 1 occurrence of

ADDR = XXXX Hex

then Trigger and fill memory

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

4 Select **Group Run** to start the analyzer(s).

5 Select **Run** in the Emulation Control Interface or use your debugger to start the target processor running.

Selecting **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

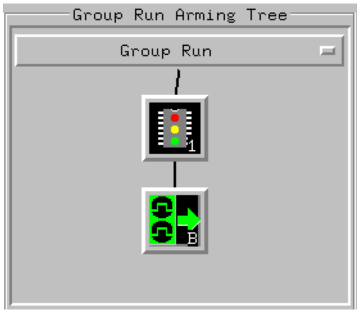
This measurement is exactly like the one on the previous page, but with the one additional complexity of setting breakpoints. Be aware that setting breakpoints may cause a false trigger and that the breakpoints set may not be valid after a reset.

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 194.

1 Set the logic analyzer to trigger on **anystate**.

2 Set the trigger point to **center** or **end**.

3 In the Intermodule window, select 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 Select **Group Run** to start the analyzer(s).

6 Select **Run** in the Emulation Control Interface or use your debugger to start the target processor running.

Selecting **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, select **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, select **Stop** if needed to complete the measurement.

11

General-Purpose ASCII (GPA) Symbol File Format

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 115, 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

```
[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

```
[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.

Define sections first

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

```
[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

```
[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

```
[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

```
[START ADDRESS]
address
```

`address` The address of the program entry point, in hexadecimal.

Example

```
[START ADDRESS]
00001000
```

Comments

```
#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.
```

12

Specifications and Characteristics

Operating Characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the E2476B MPC8XX analysis probe.

Operating Characteristics	
Microprocessor Compatibility	Motorola MPC860 and MPC821, versions DC, DE, DH, DT, DP, EN, MH, SAR, T, and P. Motorola MPC855T.
Package Supported	357-pin BGA
Microprocessor Bus Speed	50 MHz maximum
Agilent Technologies Logic Analyzers Supported	1660A/AS/C/CS/CP/E/ES/EP, 1661A/AS/C/CS/CP/E/ES/EP, 1670A/D/E, 1671A/D/E, 16550A (one or two cards), 16554A/55A/56A (two or three cards), 16555D/56D/57D (two or three cards), 16600/01/02A, 16710/11/12A (one or two cards), 16715/16/17/18/19A (2 or 3 cards), 16750/51/52 (2 or 3 cards).
Accessories Required	For state and timing analysis, the Agilent Technologies E5355A Probing Kit and the Agilent Technologies E5346A High-density Cables are required (included with the Agilent Technologies E2476B).
Optional Accessories	An emulation module can be connected to the analysis probe.
Pods Required	Six 16-channel pods are required for disassembly of MPC821/860. Connectors for six additional 16-channel pods are available.

Electrical Characteristics	
Power Requirements	100 mA @ 5V, supplied by the logic analyzer. CAT I, mains isolated. Approximately 0.1 µF decoupling on VDDH, VDD, VDDSYN, and KAPWR. Maximum draw of 2 mA from target system VDD @ 3.3 V.
Signal Line Loading	Approximately 25 pF on $\overline{\text{SRESET}}$, $\overline{\text{HRESET}}$, DSDI, DSDO, and DSCK. Approximately 15 pF on TMS and $\overline{\text{TRST}}$. Approximately 10 pF on all other signals.
Propagation Delays	Approximately 1.5 ns on DSDI, DSDO, and DSCK

Environmental Characteristics	
Temperature	Operating: 0 to +50 degrees C +32 to +122 degrees F
Altitude, Operating	4,600 m 15,000 feet
Humidity	Up to 75% non-condensing. Avoid sudden, extreme temperature changes which could cause condensation on the circuit board.
Pollution	IEC pollution degree 2. Normally only dry non-conductive pollution occurs. Occasionally a temporary conductivity caused by condensation may occur. Indoor use only.

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 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*.

Debug Port A hardware interface designed into a microprocessor that allows developers to control microprocessor execution, set breakpoints, and access microprocessor registers or target system memory using a tool like the emulation probe.

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 The hardware and software required to use an emulation probe with a new processor family.

Emulation Module An emulation module is installed within the mainframe of a logic analysis system. An E5901A emulation module is used with a *target interface module* (TIM) or an analysis probe. An E5901B emulation module is used with an E5900B *emulation probe* and does not use a TIM.

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."

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 devices coupled with a flexible cable. Used for connecting probing hardware on the target microprocessor to the analysis probe.

Gateway Address An IP address entered in integer dot notation. The default gateway address is 0.0.0.0, which allows all connections on the local network or subnet. If connections are to be made across networks or subnets, this address must be set to the address of the gateway machine.

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.

IP address Also called Internet Protocol address or Internet address. A 32-bit network address. It is usually represented as decimal numbers separated by periods; for example, 192.35.12.6.

Jumper Moveable direct electrical connection between two points.

JTAG (OnCE) port See *debug port*.

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.

Link-Level Address The unique address of the LAN interface. This value is set at the factory and cannot be changed. The link-level address of a particular piece of equipment is often printed on a label above the LAN connector. An example of a link-level address in hexadecimal: 0800090012AB. Also known as an LLA, Ethernet address, hardware address, physical address, or MAC address.

Mainframe Logic Analyzer A logic analyzer that resides on one or more board assemblies installed in a 16500, 1660-series, or 16600/700-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.

MICTOR Connector A high-density matched impedence connector manufactured by AMP Corporation. *High-density adapter cables* can be used to connect the logic analyzer to MICTOR connectors on the target system.

Monitor, In See *Background Debug Monitor*.

Pod A collection of logic analyzer channels associated with a single cable and connector.

Preprocessor See *Analysis Probe*.

Preprocessor Interface See *Analysis Probe*.

Probe Adapter See *Elastomeric Probe Adapter*.

Processor Probe See *Emulation Probe*.

Run Control Probe See *Emulation Probe* and *Emulation Module*.

Setup Assistant Wizard software program which guides a user through the process of connecting and configuring a logic analyzer to make measurements on a specific microprocessor. The setup assistant icon is located in the main system window.

Shunt Connector. See *Jumper*.

Solution A set of tools for debugging your target system. A solution includes probing, inverse assembly, the 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. 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 may be installed within its frame.

State Analysis A mode of logic analysis in which the logic analyzer is configured to capture data synchronously with a clock signal in the target system.

Subnet Mask A subnet mask blocks out part of an IP address so the networking software can determine whether the destination host is on a local or remote network. It is usually represented as decimal numbers separated by periods; for example, 255.255.255.0.

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 Board Adapter A daughter board inside the E5900B emulation probe which customizes the emulation probe for a particular microprocessor family. The target board adapter provides an interface to the ribbon cable which connects to the debug port on the target system.

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 E5901A emulation module or E5900A emulation probe to signals from the debug port on a target system. Not used with the E5900B emulation probe.

TIM See *Target Interface Module*.

Timing Analysis A mode of logic analysis in which the logic analyzer is configured to capture data at a rate determined by an internal sample rate clock, asynchronous to signals in the target system.

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.

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.

1/4-Flexible Adapter An adapter that obtains one-quarter of the signals from an elastomeric probe adapter (one side of a target microprocessor) and makes them available for probing.

Index

Index

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.
Agilent Technologies,
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.

Agilent Technologies 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.

Agilent Technologies shall not be liable for errors contained herein or for damages in connection with the furnishing, performance, or use of this material.

Safety

This apparatus has been designed and tested in accordance with IEC Publication 1010, Safety Requirements for Measuring Apparatus, and has been supplied in a safe condition. This is a Safety Class I instrument (provided with terminal for protective earthing). Before applying power, verify that the correct safety precautions are taken (see the following warnings). In addition, note the external markings on the instrument that are described under "Safety Symbols."

Warning

- Before turning on the instrument, you must connect the protective earth terminal of the instrument to the protective conductor of the (mains) power cord. The mains plug shall only be inserted in a socket outlet provided with a protective earth contact. You must not negate the protective action by using an extension cord (power cable) without a protective conductor (grounding). Grounding one conductor of a two-conductor outlet is not sufficient protection.
- Only fuses with the required rated current, voltage, and specified type (normal blow, time delay, etc.) should be used. Do not use repaired fuses or short-circuited fuseholders. To do so could cause a shock of fire hazard.
- Service instructions are for trained service personnel. To avoid dangerous electric shock, do not perform any service unless qualified to do so. Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.
- If you energize this instrument by an auto transformer (for voltage reduction), make sure the common terminal is connected to the earth terminal of the power source.
- Whenever it is likely that the ground protection is impaired, you must make the instrument inoperative and secure it against any unintended operation.
- Do not operate the instrument in the presence of flammable gasses or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.
- Do not install substitute parts or perform any unauthorized modification to the instrument.
- Capacitors inside the instrument may retain a charge even if the instrument is disconnected from its source of supply.

Safety Symbols

⚠

Instruction manual symbol: the product is marked with this symbol when it is necessary for you to refer to the instruction manual in order to protect against damage to the product.

⚡

Hazardous voltage symbol.

⏚

Earth terminal symbol: Used to indicate a circuit common connected to grounded chassis.

WARNING

The Warning sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to, could result in personal injury. Do not proceed beyond a Warning sign until the indicated conditions are fully understood and met.

CAUTION

The Caution sign denotes a hazard. It calls attention to an operating procedure, practice, or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product. Do not proceed beyond a Caution symbol until the indicated conditions are fully understood or met.

Agilent Technologies
P.O. Box 2197
1900 Garden of the Gods Road
Colorado Springs, CO 80901-2197, U.S.A.

Product Warranty

This Agilent Technologies product has a warranty against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Agilent Technologies 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 Agilent Technologies.

For products returned to Agilent Technologies for warranty service, the Buyer shall prepay shipping charges to Agilent Technologies and Agilent Technologies 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 Agilent Technologies from another country.

Agilent Technologies warrants that its software and firmware designated by Agilent Technologies for use with an instrument will execute its programming instructions when properly installed on that instrument. Agilent Technologies 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. Agilent Technologies 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. Agilent Technologies 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 Agilent Technologies products. For any assistance, contact your nearest Agilent Technologies Sales Office.

Certification

Agilent Technologies certifies that this product met its published specifications at the time of shipment from the factory. Agilent Technologies 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 *Logic Analysis Support for Motorola PowerPC MPC8XX User's Guide*.

Publication number
E2476-97007, June, 2000
Printed in USA.

The information in this manual previously appeared in:

E2476-97006,	November, 1999
E2476-97005,	July, 1999
E2476-97004	January 1999
E2476-97003	March 1998
E2476-97002	June 1997
E2476-97001	March 1997
E2467-97000	January 1997
E2477-97000	September 1996
E3497-97001	March 1997

New editions are complete revisions of the manual. Many product updates do not require manual changes, and manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.