

User's Guide

Publication number E8128-97005
August 2000

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

© Copyright Agilent Technologies 1994-2000
All Rights Reserved

Logic Analysis Support for the Motorola MPC8240

Logic Analysis Support for the Motorola MPC8240—At a Glance

The Agilent Technologies E8128A inverse assembler, in conjunction with an Agilent Technologies 16600/700-series logic analyzer, allows you to view MPC8240 assembly instructions that are executing in your target system.

The inverse assembler model number is Agilent Technologies 9611A Option 001 when ordered alone. You can also order an analysis probe and inverse assembler, which provides the hardware for easy connection to a target system. The model number for the analysis probe and inverse assembler is Agilent Technologies 9611A Option 002.

The inverse assembler is identified as “Agilent Technologies E8128A” in the Setup Assistant. The analysis probe and inverse assembler is identified as “Agilent Technologies E8127A” in the Setup Assistant.

If You Purchased an Emulation Solution

The E9511A emulation solution lets you use an Agilent Technologies 16600/700-series logic analyzer to debug and characterize Motorola MPC8240 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 Motorola MPC8240 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 9, “Coordinating Logic Analysis with Processor Execution”, beginning on page 142 of this manual.

Additional Equipment Included in an Emulation Solution

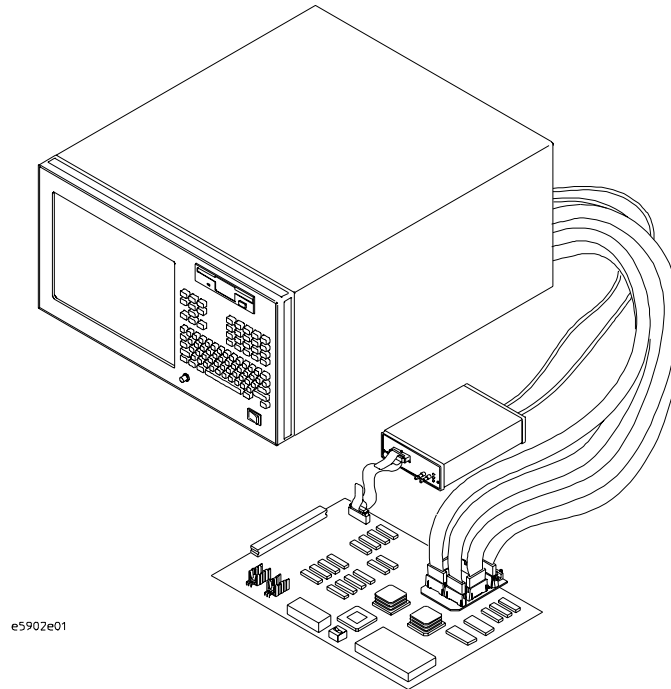
Emulation Module

The emulation module plugs into your Agilent Technologies 16600/700-series logic analysis 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 the microprocessor's built-in debugging features (like starting/stopping program execution, setting breakpoints, and modifying the contents of processor registers and target system 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:

Product Ordered	Supports	Includes
E9611A Option 001 inverse assembler	MPC8240	The E8128A inverse assembler
E9611A Option 002 analysis probe and inverse assembler	MPC8240	The E8127A analysis probe and inverse assembler, and the BGA probing kit (part number E8161-60001)

Related equipment

The following equipment is included in the MPC8240 emulation solution.

Product Ordered	Supports	Includes
E9511A Option 001 emulation solution	MPC8240	The E8128A inverse assembler, emulation module, emulation probe, and B4620B Source Correlation Tool Set
E9511A Option 002 emulation solution	MPC8240	The E8127A analysis probe and inverse assembler, BGA probing kit, emulation module, emulation probe, and 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 18.

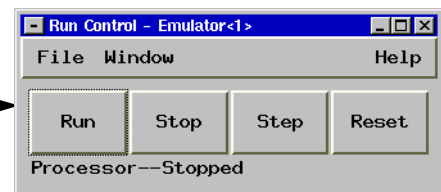


Use the appropriate Run button



Click here to start a measurement.

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



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 microcontroller 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 16600/700-series logic analysis system, the **online help** for the Emulation Control Interface has additional information on using the emulation module. Also, see the emulation manual included with your emulation probe/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.

Logic Analysis Support for the Motorola MPC8240—At a Glance

Additional Equipment Included in an Emulation Solution	3
Emulation Module	3
Source Correlation Tool Set	3
In This Book	4
Related equipment	4

Tips To Save You Time

Use the Setup Assistant	5
Use the appropriate Run button	5
Additional Information Sources	6

1 Equipment and Requirements 15

Setup Checklist	17
Setup Assistant	18
Equipment and Software Supplied	20
Analysis Probe	20
Inverse Assembler (no Analysis Probe)	21
Additional equipment required	22
Additional equipment supported	22
Compatible Logic Analyzers	23
Emulation Solution	25

2 Preparing the Target System 27

Preparing for Logic Analysis (and Inverse Assembly)	29
Design Considerations	29
Designing A Target System for the Analysis Probe	31
Electrical Requirements	31
Mechanical Requirements	32
Attaching the Analysis Probe to the Target System	35
Overview	35
To solder the BGA socket onto the target system	37
To assemble the microprocessor into the BGA carrier	38
To test the target system with the BGA carrier assembly	39
To remove the BGA carrier assembly (or analysis probe) from the socket	40
To attach the analysis probe (with BGA carrier) to the target system	41
Designing Logic Analyzer Connectors into Your Target System	43
Using High-Density Connectors	43
Recommended Connector Layout and Signal Routing	45
Designing a Debug Port Connector into Your Target System	54

3 Setting Up the Logic Analysis System 55

Power-ON/Power-OFF Sequence	56
To power-ON the Agilent Technologies 16600/700-series logic analysis systems	56
To power-OFF	56
Installing Logic Analyzer Modules	57
Installing an emulation module	57

Installing Software	58
To install the software from CD-ROM	59

4 Probing the Target System 61

Connecting the Logic Analyzer to the Target System	62
64-bit data	62
32-bit data	62
No data	63
To connect the high-density termination cables to the analysis probe	64
To connect a two-card 16550A logic analyzer for 64-bit or 32-bit data analysis	65
To connect a 16550A logic analyzer for no-data analysis	67
To connect a two-card 16554/55/56/57 logic analyzer for 64-bit or 32-bit data analysis	68
To connect a 16554/55/56/57 logic analyzer for no-data analysis	69
To connect the 16600A logic analyzer for 64-bit or 32-bit data analysis	70
To connect the 16600A logic analyzer for no-data analysis	71
To connect the 16601A logic analyzer for 64-bit or 32-bit data analysis	72
To connect the 16601A logic analyzer for no-data analysis	73
To connect the 16602/3A logic analyzer for no-data analysis	74
To connect a two-card 16710/11/12A or logic analyzer for 64-bit or 32-bit data analysis	75
To connect a 16710/11/12A logic analyzer for no-data analysis	77
To connect a two-card 16715/16/17/18/19A or 16750/51/52A logic analyzer for 64-bit or 32-bit data analysis	78
To connect a 16715/16/17/18/19A or 16750/51/52A logic analyzer for no-data analysis	79
Connecting the Logic Analyzer to a Motorola PMC8240 Board	80

5 Configuring the Logic Analyzer 83

Configuring 16600/16700-series Logic Analysis Systems	85
To load configuration files (and the inverse assembler) from hard disk	86
To load configuration files (and the inverse assembler) from floppy disk	87
To list software packages that are installed	88
Logic analyzer configuration files	88
Using the Inverse Assembler	89
Using Cache-On Trace Reconstruction	89
Enabling branch exception disassembly	92
Inverse Assembler Modes of Operation	93
To use the Invasm menu	94
Loading the Inverse Assembler	94
Setting Inverse Assembler Preferences	95
To set the inverse assembler preferences	95
To set the Memory Map preferences	96
To set the Processor Options preferences	102
To set the Decoding Options preferences	103
To set the Opcode Source preferences	105
To enable/disable the instruction cache on the MPC8240	107
Loading Symbol Information	109
To view predefined symbols for the MPC8240	109
To load object file symbols	111
To compensate for relocated code	114
Setting Up Labels for Groups of Signals	115
Predefined Label Descriptions	115
To define additional labels	117
Changing the Analysis Mode	118
To change to state analysis	118
To change to timing analysis	119

6 Capturing MPC8240 Execution 121

- Setting Up Logic Analyzer Triggers 123
 - To set up logic analyzer triggers 123
 - To trigger on an access to a memory address 125

- Using the Saved Trigger Specifications 128
 - MIV store qualified 128
 - SDRAM Address 128
 - FLASH/ROM Address 128
 - SDRAM Instruction Fetch 128
 - FLASH/ROM Instruction Fetch 129

- Triggering on Source Code 130
 - To avoid capturing library code execution 130

7 Displaying Captured MPC8240 Execution 131

- To display the captured state data 132
- To display symbols 133
- To interpret the inverse assembled data 134
- To use the inverse assembler filters 135
- To view the source code associated with captured data 137
- To display captured timing analysis mode data 140

8 Coordinating Logic Analysis with Processor Execution 141

- Stopping Processor Execution on a Logic Analyzer Trigger 144
 - To stop on a source line trigger (Source Viewer window) 144

Contents

To stop the processor when the logic analyzer triggers (Intermodule window)	146
To minimize the “skid” effect	147
To stop the analyzer and view a measurement	147
Tracing Until the Processor Halts	149
To capture a trace before the processor halts	149
Triggering the Logic Analyzer when Processor Execution Stops	150
To trigger the analyzer when the processor halts	152
To trigger the analyzer when the processor reaches a breakpoint	154

9 General-Purpose ASCII (GPA) Symbol File Format 157

General-Purpose ASCII (GPA) Symbol File Format	158
GPA Record Format Summary	160
SECTIONS	162
FUNCTIONS	163
VARIABLES	164
SOURCE LINES	165
START ADDRESS	166
Comments	166

10 Specifications and Characteristics 167

Operating Characteristics	168
---------------------------	-----

11 Troubleshooting the Logic Analyzer 175

Solving Logic Analyzer Problems	177
Intermittent data errors	177
Unwanted triggers	177
No activity on activity indicators	178
No trace list display	178
Solving Probing Problems	179
Target system will not boot up	179
Erratic trace measurements	180
Capacitive loading	180
Solving Inverse Assembler Problems	181
No inverse assembly or incorrect inverse assembly	181
Inverse assembler will not load or run	182
Solving Intermodule Measurement Problems	183
An event wasn't captured by one of the modules	183
Logic Analyzer Messages	184
“. . . Inverse Assembler Not Found”	184
“Measurement Initialization Error”	184
“No Configuration File Loaded”	186
“Selected File is Incompatible”	186
“Slow or Missing Clock”	186
“Time from Arm Greater Than 41.93 ms”	187
“Waiting for Trigger”	187

Glossary 189

Index 195

Equipment and Requirements

Chapter 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 20.
- 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 58.
- Install the analysis probe, if ordered. See page 35. 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 18.

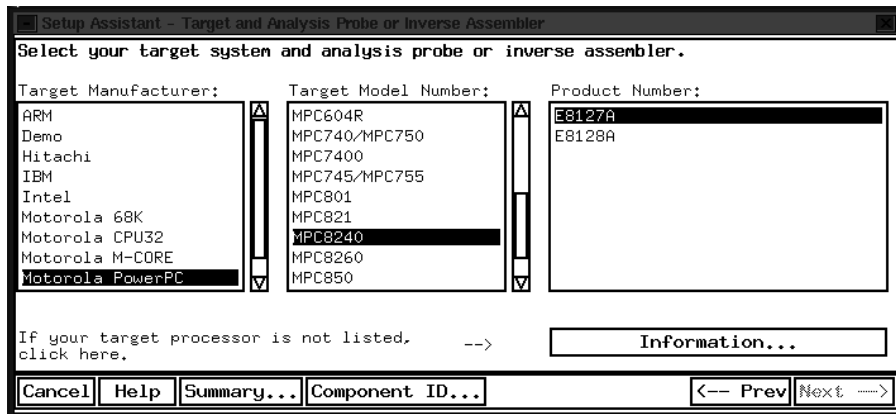
Setup Assistant

The Setup Assistant is an on-line tool for connecting and configuring your logic analysis system for microprocessor and bus analysis. The Setup Assistant is available on the 16600/700-series logic analysis systems. You can use the Setup Assistant in place of the connection and configuration procedures provided in this manual.

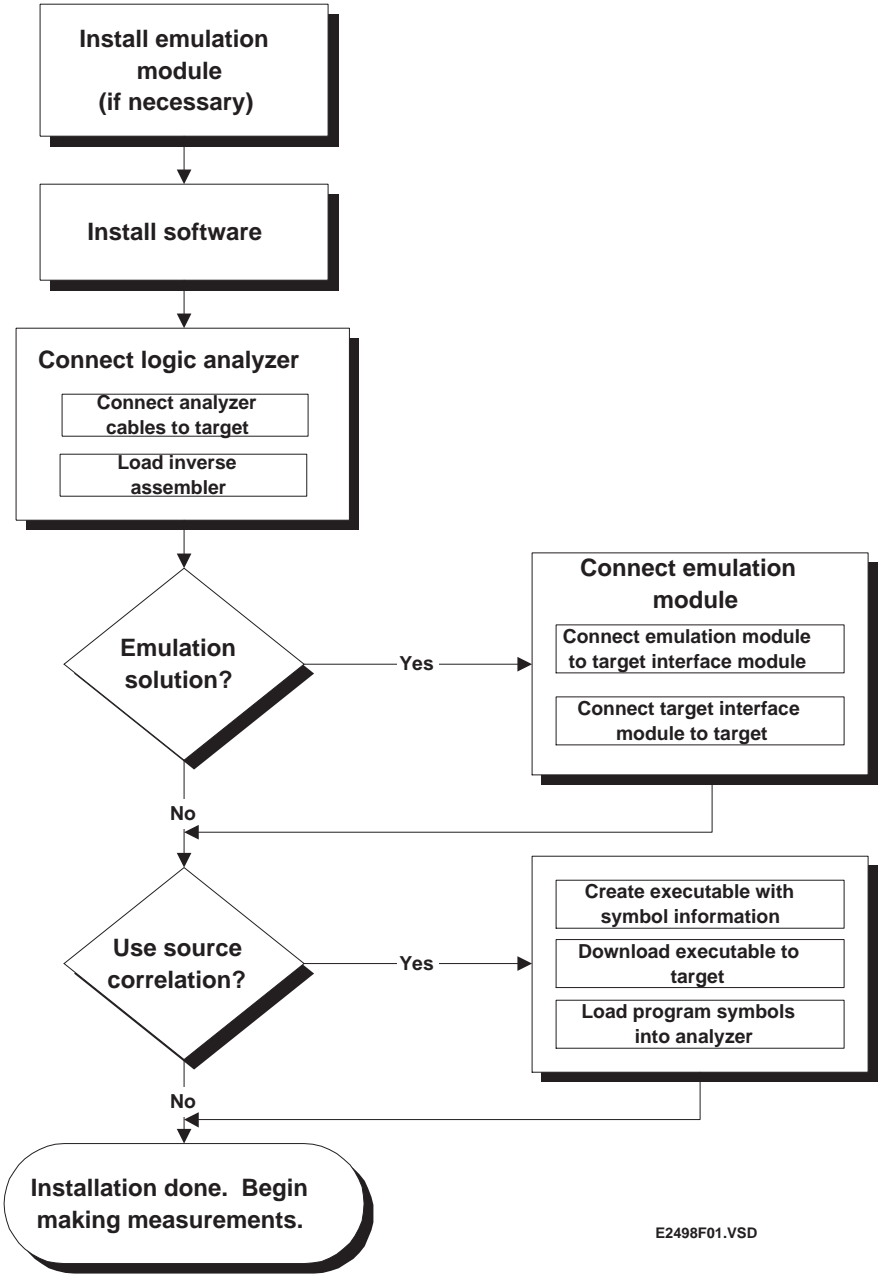
This menu-driven tool will guide you through the connection procedures for connecting the target system to an analysis probe, an emulation module, or other supported equipment. It will also guide you through connecting the logic analyzer pods to connectors on the target system.

The inverse assembler is identified as “Agilent Technologies E8128A” in the Setup Assistant. The analysis probe and inverse assembler is identified as “Agilent Technologies E8127A” in the Setup Assistant.

Start the Setup Assistant by clicking  in the system window.



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.



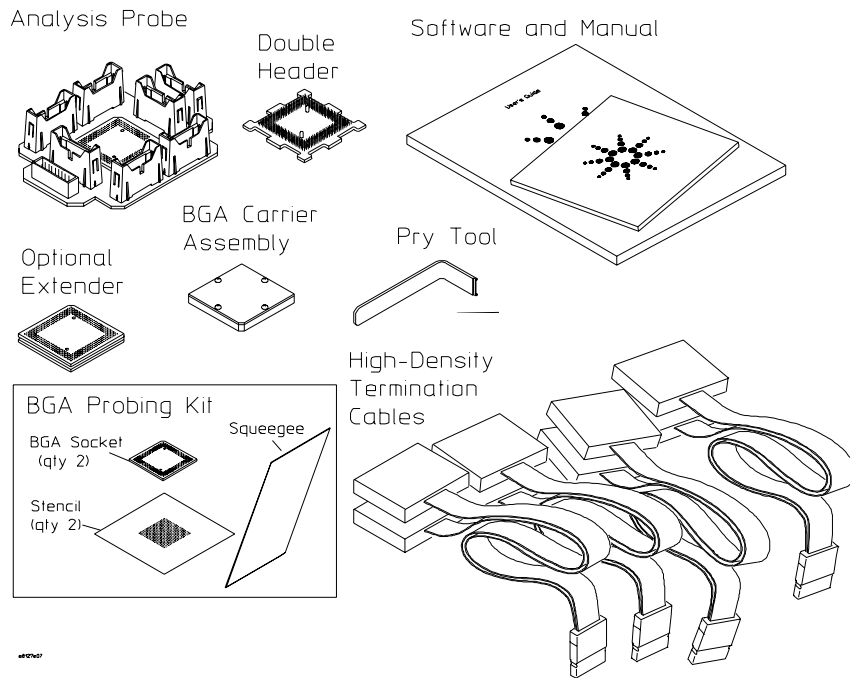
E2498F01.VSD

Equipment and Software Supplied

Listed below is the equipment and software supplied with:

- The Agilent Technologies E9611A Option 002 analysis probe and inverse assembler.
- The Agilent Technologies E9611A Option 001 inverse assembler.

Analysis Probe



The analysis probe includes:

- Agilent Technologies E8127A analysis probe.
- Four Agilent Technologies E5346A high-density termination cables.
- BGA probing kit (Agilent part number E8161-60001).
- BGA carrier assembly.
- Double header.
- Extender.
- One pry tool.
- Logic analyzer configuration files and the inverse assembler software on a CD-ROM (for Agilent Technologies 16600/700-series logic analysis systems).
- This *User's Guide*.

Inverse Assembler (no Analysis Probe)

The inverse assembler (Agilent Technologies E9611A Option 001 when ordered separately) includes:

- Logic analyzer configuration files and the inverse assembler software on a CD-ROM (for Agilent Technologies 16600/700-series logic analysis systems).
- This *User's Guide*.

The inverse assembler, without the analysis probe, is identified as “E8128A” in the Setup Assistant.

Additional equipment required

In addition to the items listed above, the following is required to analyze an MPC8240 target system:

- A target system with an empty 352-pin BGA socket, for the BGA probing kit. A BGA microprocessor is also required.
- One of the logic analyzers listed on page 23. 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 27 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

A minimum of four logic analyzer pods (68 logic analysis channels) are required for inverse assembly. For inverse assembly of both instructions and data, eight pods are required. If optional signals are used, such as PCI analysis, additional logic analyzer pods are required.

The MPC8240 inverse assembler only works in 16600/700-series logic analysis systems. The 16500 logic analysis system and 166x/167x portable logic analyzer families are not supported.

The inverse assembler works with logic analysis system software version A.01.41 or greater. The latest logic analysis system software version is on the CD-ROM shipped with this product.

Given these restrictions, the following logic analyzers can be used:

Logic Analyzers Supported

Logic Analyzer	Channel Count	State Speed	Timing Speed	Memory Depth
16752A (1, 2, or 3 cards)	68/card	400 MHz	2 GHz	32 M states
16751A (1, 2, or 3 cards)	68/card	400 MHz	2 GHz	16 M states
16750A (1, 2, or 3 cards)	68/card	400 MHz	2 GHz	4 M states
16719A (2 or 3 cards)	68/card	333 MHz	2 GHz	32 M states
16718A (2 or 3 cards)	68/card	333 MHz	2 GHz	8 M states
16717A (2 or 3 cards)	68/card	333 MHz	2 GHz	2 M states
16716A (2 or 3 cards)	68/card	167 MHz	2 GHz	512 k states
16715A (2 or 3 cards)	68/card	167 MHz	2 GHz	2 M states
16712A (1 or 2 cards)	102/card	100 MHz	500 MHz	128 k states
16711A (1 or 2 cards)	102/card	100 MHz	500 MHz	32 k states

Compatible Logic Analyzers

Logic Analyzer	Channel Count	State Speed	Timing Speed	Memory Depth
16710A (1 or 2 cards)	102/card	100 MHz	500 MHz	8 k states
16603A	68	100 MHz	125 MHz	64 k states
16602A	102	100 MHz	125 MHz	64 k states
16601A	136	100 MHz	125 MHz	64 k states
16600A	204	100 MHz	125 MHz	64 k states
16557D (1 or 2 cards)	68/card	135 MHz†	250 MHz	2 M states
16556A (1, 2, or 3 cards)	68/card	100 MHz	200 MHz	1 M states
16555D/56D (1, 2, or 3 cards)	68/card	100 MHz	500/400 MHz	2 M states
16555A (1, 2, or 3 cards)	68/card	110 MHz	250 MHz	1 M states
16554A (1, 2, or 3 cards)	68/card	70 MHz	125 MHz	512 k states
16550A (1 or 2 cards)	102/card	100MHz	250 MHz	4 k states
1671A	102	70 MHz	125 MHz	64 k or 0.5 M
1670D/E 1671D/E	136/102	100 MHz	250 MHz	1 M states
1670A	136	70 MHz	125 MHz	64 k or 0.5 M
1661A/AS/C/CS/CP/E/ ES/EP	102	100 MHz	250 MHz	4 k states
1660A/AS/C/CS/CP/E/ ES/EP	136	100 MHz	250 MHz	4 k states

Notes:

- * State and timing speeds are provided for identification purposes only. Actual performance may vary based on system configuration.

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 MPC8240 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 MPC8240 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 B4620B Source Correlation Tool Set to analyze high-level source using the logic analysis system.

Preparing the Target System

Chapter 2: Preparing the Target System

There are two ways to probe an MPC8240 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 MPC8240 inverse assembler for the Agilent Technologies 16600/700-series logic analysis system requires a minimum of four logic analyzer pods.

For inverse assembly of both instructions and data, 8 pods are required. If optional signals are used, such as PCI analysis, additional logic analyzer pods are required.

If you are not using an analysis probe or PMC board, you must design high-density connectors into your target system for logic analyzer probe pods.

This section describes these considerations in more detail.

Design Considerations

There are several things to keep in mind when designing a MPC8240 target system.

Configuring for debug mode

Full address information does not normally appear on the MPC8240 bus. To reconstruct physical addresses, the inverse assembler uses information from the 16 debug address pins, which are enabled by setting the processor to run in debug mode. Debug mode enables the inverse assembler to reconstruct full 32-bit physical addresses.

To enable debug mode in software

Set the WP_DEBUG_ bit in the Watchpoint Control Register (WP_CONTROL bit 28) to 0.

To enable debug mode in hardware

Pull down $\overline{\text{GNT}}[4]$ (pin W26 on the BGA), which is sampled at reset.

Cache memory

Internal Instruction Cache. The microprocessor supplies no external information when the cache is enabled. This poses a problem for logic analysis, since there is no linear address and data cycles to reconstruct code flow. Therefore, cache-on trace reconstruction must be used. See “Using Cache-On Trace Reconstruction” on page 89.

Data Cache. The microprocessor provides no external information for accesses to the internal data cache. The inverse assembler does not support accesses to the internal data cache.

Electrical requirements

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 k Ω shunted by 10 pF. The maximum input voltage for the logic analyzer is +/- 40 volts peak.

For all signals, the logic analyzers require a minimum combined setup/hold window. For Agilent Technologies 16600-series logic analysis systems, the combined setup/hold window must be at least 4.5 ns. For all other logic analyzers, the combined window must be at least 3.5 ns.

The signals labelled “Pull up V_{dd} (TTL)” in the tables beginning on page 47 should be pulled up to V_{dd} using 10 k Ω (approximate value) resistors.

Designing A Target System for the Analysis Probe

Electrical Requirements

In addition to the requirements listed in “Electrical requirements” on page 30, your target system should meet the following requirements:

Electrical requirements for PCI bus analysis

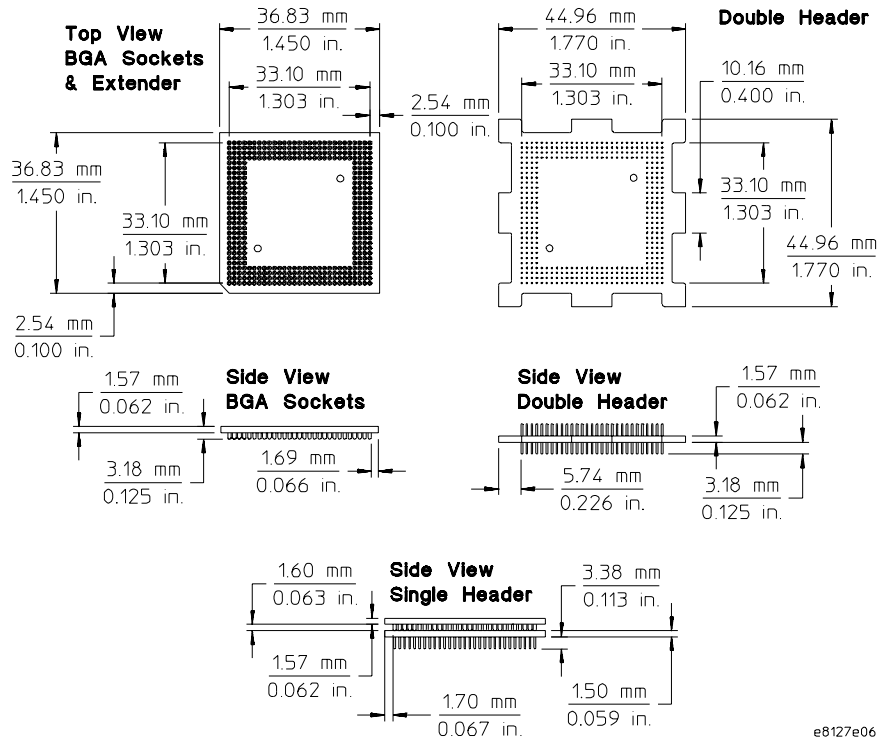
You do not need to condition the PCI bus signals; the analysis probe preserves both the 3.3V and 5V signal levels.

Mechanical Requirements

Keep-Out Area on the Target Board

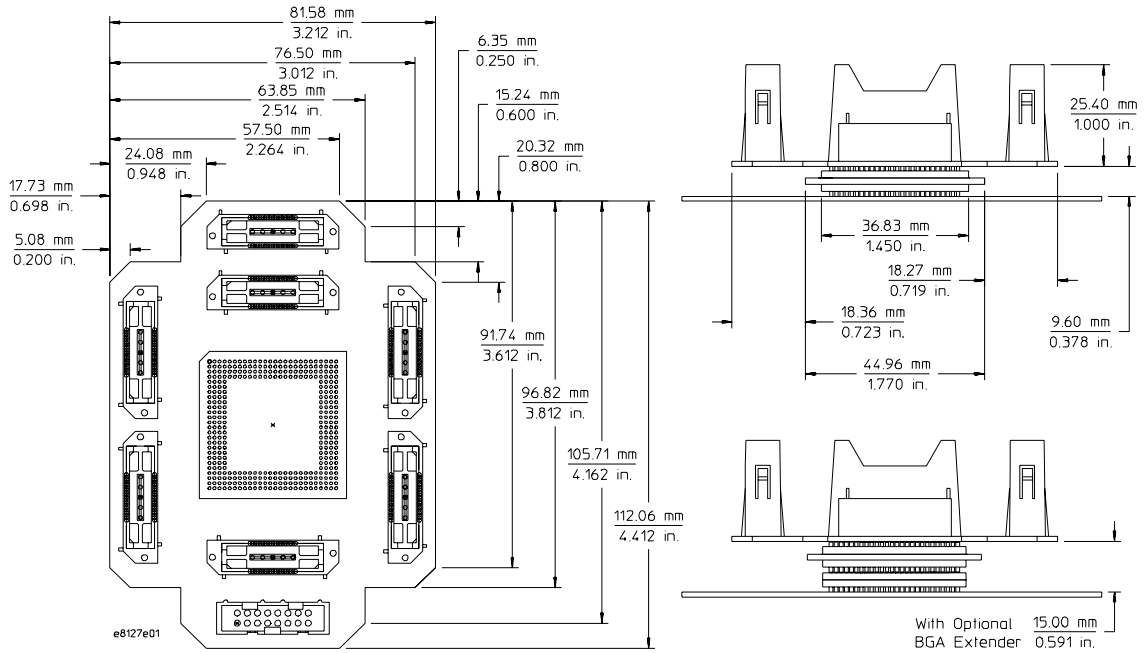
The BGA socket placed into the BGA footprint extends slightly beyond the outer row of pads. It is important to keep components from interfering with the socket in this restricted area.

The analysis probe requires a keep-out area of 44.96 mm by 44.96 mm where it overhangs the BGA socket. Components within the required keep-out area must be no higher than 3.18 mm. If the analysis probe interferes with components of the target system, or if a higher profile is required, additional BGA extenders (Agilent part number E8127-87607) can be used, but they add additional electrical intrusion.



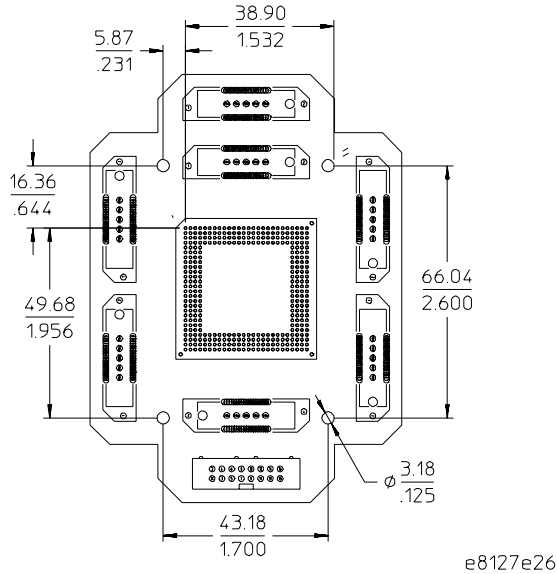
Clearance above the Target Board

Be careful to allow adequate space above the target board for the analysis probe, and for egress of the logic analyzer cables.



Mounting Holes

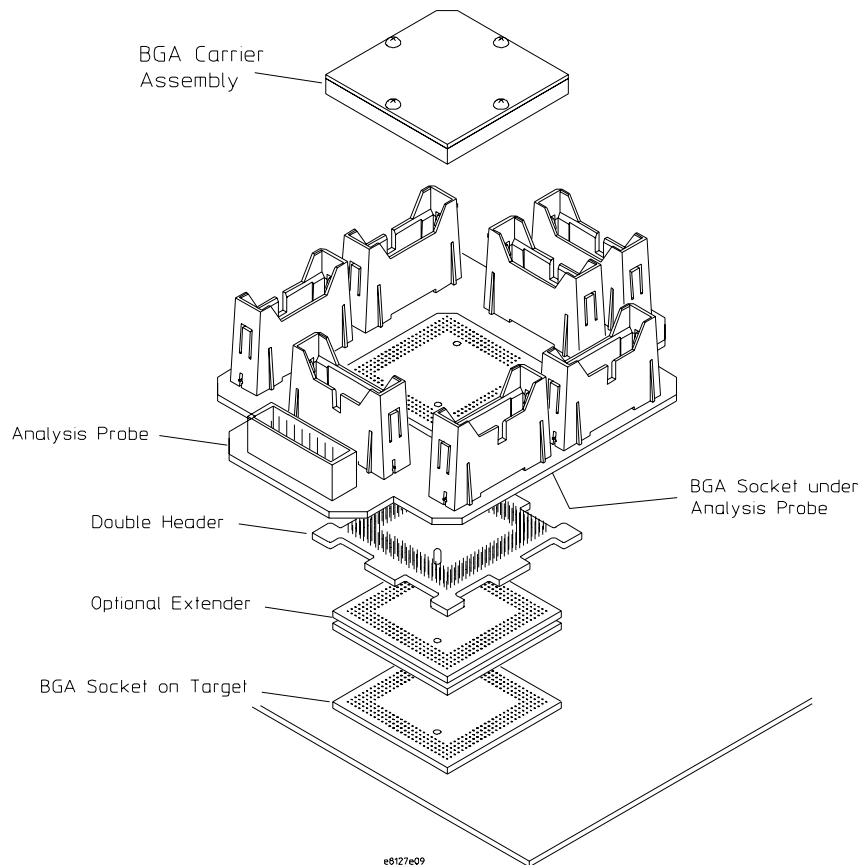
Agilent recommends drilling four non-plated holes according to the following dimensions. Mounting the analysis probe greatly alleviates stress on the fragile solder joints when the logic analyzer cables are connected.



Attaching the Analysis Probe to the Target System

If you are designing logic analyzer and debug port 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.

Overview



Attaching the Analysis Probe to the Target System

Attaching the analysis probe to the target system consists of the following steps, which are described on the following pages:

- Solder the BGA socket onto the target system.
- Assemble the microprocessor into the BGA carrier.
- Test the target system with the BGA carrier assembly, without the analysis probe.
- 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.

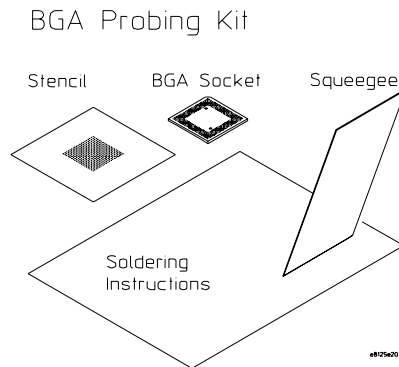
Protect Your Equipment

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

To solder the BGA socket onto the target system

The BGA probing kit, Agilent part number E8161-60001, requires a target system with an empty 352-pin BGA pad array. Install the BGA probing kit using the following instructions.

- 1 Ensure that your target system has a 352-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 Following the soldering instructions in the process sheet that came with the BGA probing kit, install the socket onto the 352-pin BGA pad array, and solder it in place.



To assemble the microprocessor into the BGA carrier

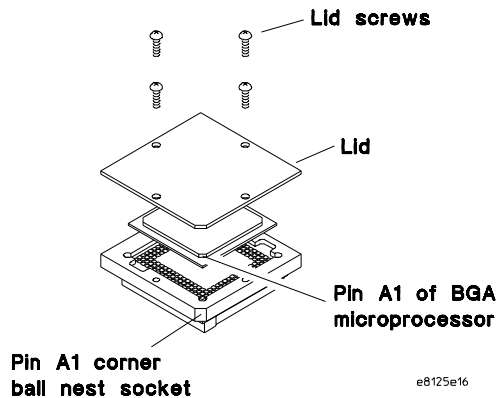
The analysis probe has a BGA carrier for a 352-pin BGA microprocessor. Use the procedure below to install the BGA microprocessor into the BGA carrier.

- 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 lid screws.



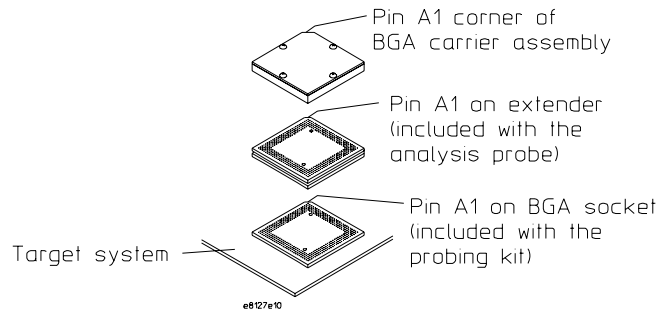
CAUTION:

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 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.



- 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 remove the BGA carrier assembly (or analysis probe) from the socket

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 BGA socket.

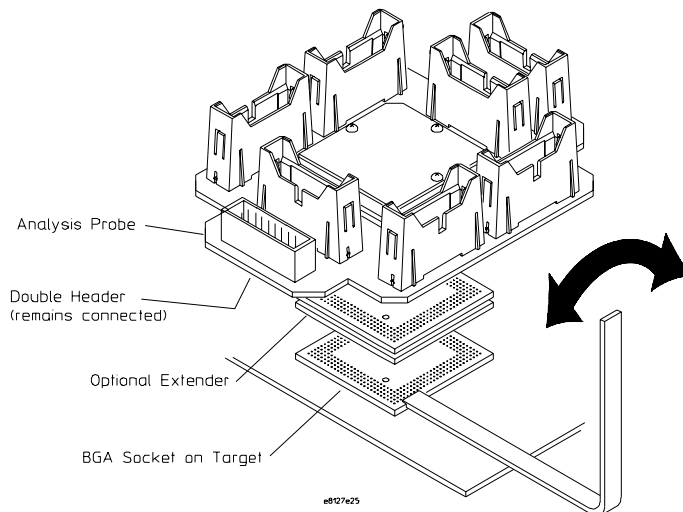
The extractor tool comes with an *Operating Guide* showing how to use the extractor tool to disconnect the BGA carrier assembly or the analysis probe from the BGA socket.

Follow these guidelines:

- Refer to the *Operating Guide* and use the extractor tool to lift the BGA carrier assembly (or the analysis probe) from the socket. Keep all connector pins straight during removal.
- Do not plug anything other than the analysis probe or optional extender into the BGA socket on the target board.
- Separate the BGA carrier assembly (or the analysis probe) from the socket on the target board, or from the optional extender.

CAUTION:

Do not remove the double header from the analysis probe board. Removing the double header could damage the analysis probe.



To attach the analysis probe (with BGA carrier) to the target system

A BGA socket is on the bottom of analysis probe. It connects to the double header which in turn connects to the extender or socket on the target system.

- 1 Install the double header into the BGA socket on the bottom of the analysis probe.
- 2 Install the analysis probe into the extender or socket on the target system. Ensure that pin A1 is properly aligned (see the following figure).

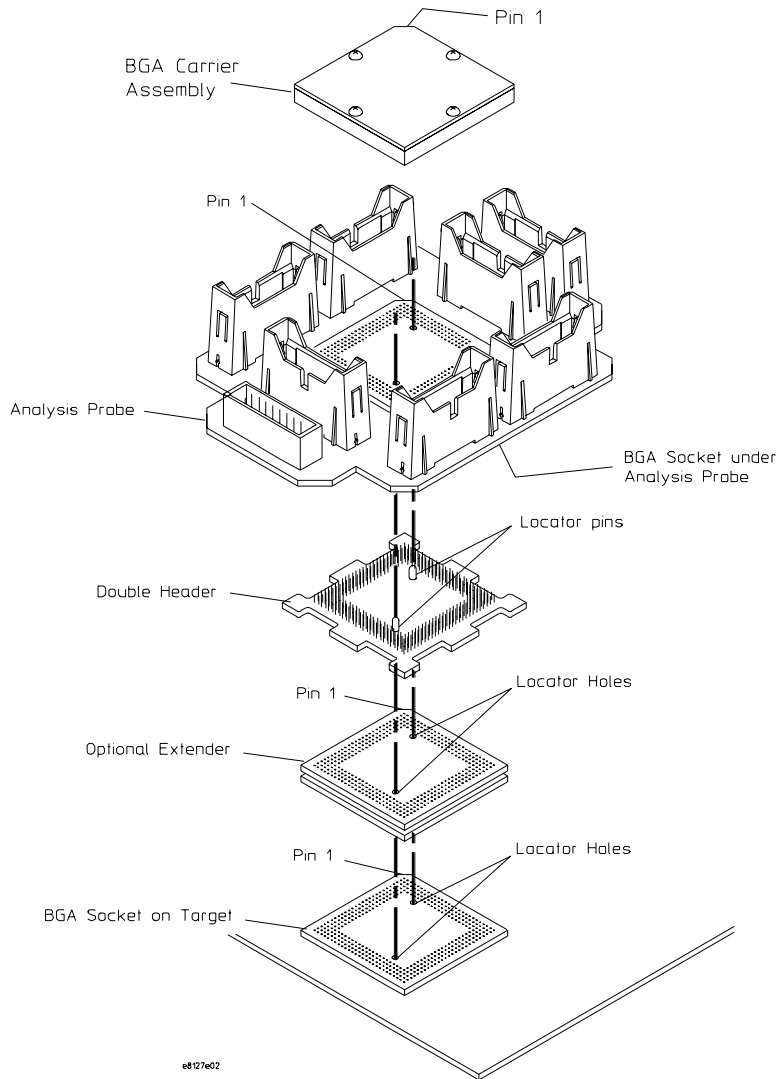
CAUTION:

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, double header, analysis probe, 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 (Agilent part number E8127-87607) can be used.

Chapter 2: Preparing the Target System

Attaching the Analysis Probe to the Target System



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.

Refer to “Electrical Requirements” on page 31 for additional information on routing design.

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 Technologies 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 KOhms shunted by 10 pF. The maximum input voltage for the logic analyzer is +/- 40 volts peak.
- If a printed-circuit board already has a header connector attached, but the signal pinouts do not match the requirement, an adapter (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.

Designing Logic Analyzer Connectors into Your Target System

See Also

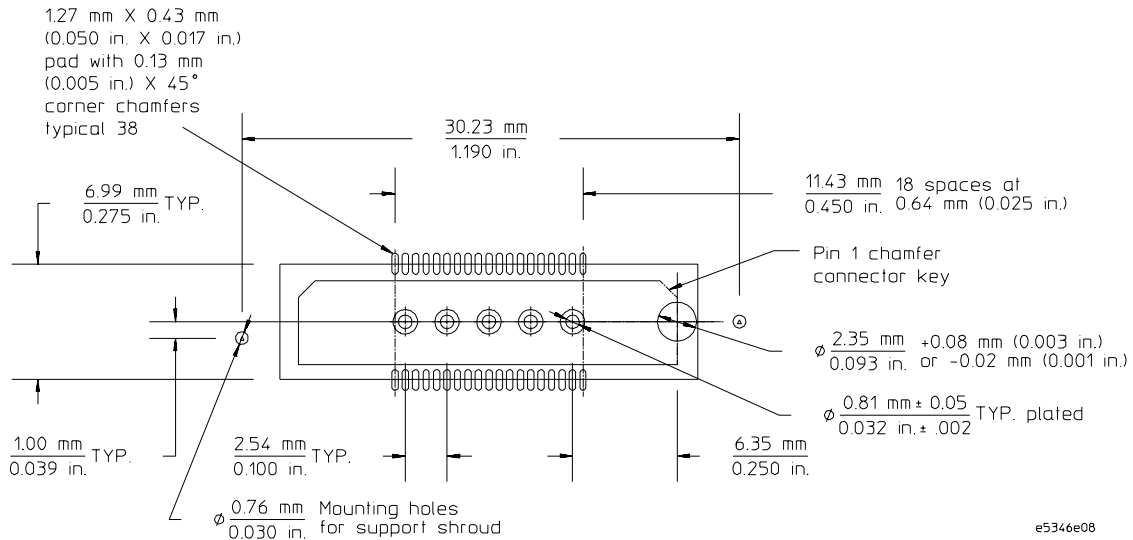
More information on this connector is available in Portable Document Format (PDF) from the web site:

<http://www.tm.agilent.com/>

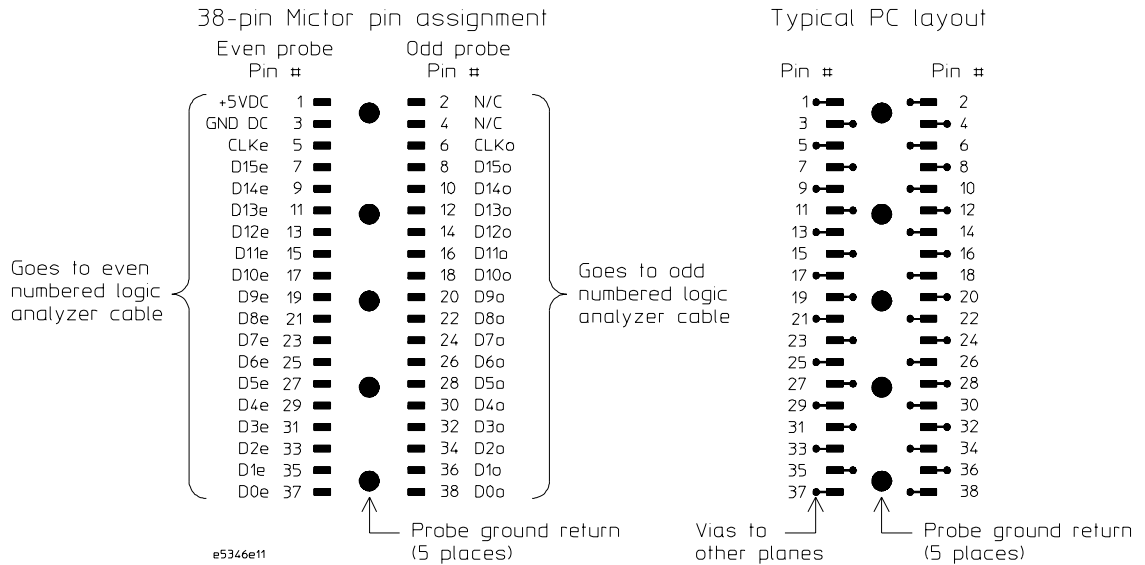
When you reach this web site, search on E5346A.

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.



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

Recommended Connector Layout and Signal Routing

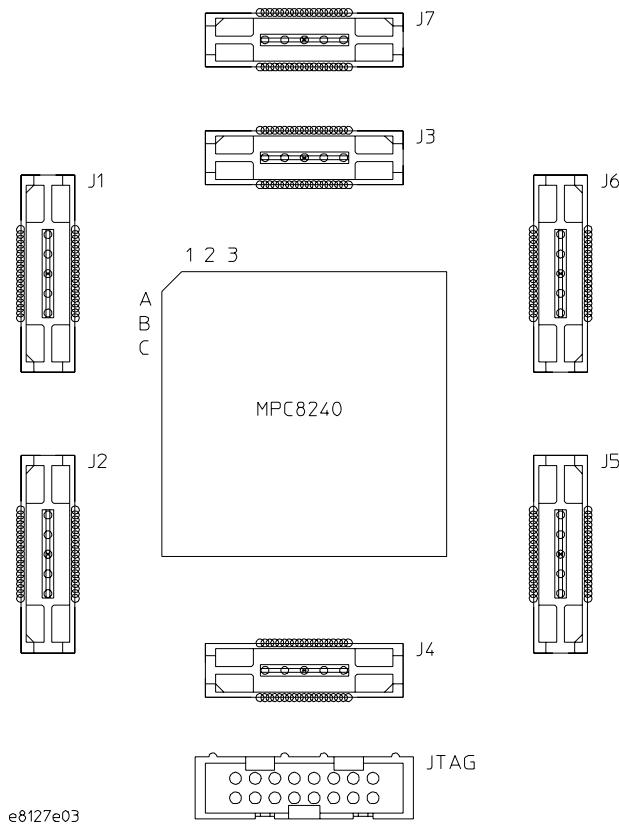
The following shows the recommended connector layout and signal routing.

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 overscore denotes an active low signal.
- J1-J2: Required for inverse assembly.
- J3 odd, J4 odd: Required for inverse assembly with 32-bit data.
- J3 even, J4 even: Required for inverse assembly with 64-bit data.
- J7: Miscellaneous signals
- J5-J6: (Optional) Required for PCI bus.

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

MICTOR Connector J1					
J1 Pin	BGA Pin	MPC8240 Signal	J1 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5	A16	\overline{MIV}	6	D1	SDRAM_CLK[0]
7	Y2	RTC	8	B15	debug_addr[1]/CKO
9	H2	CKE	10	F2	debug_addr[0]/ \overline{QACK}
11	J2	debug_addr[15]/MTP[0]	12	P2	SDBA0
13	F1	debug_addr[14]/MTP[1]	14	P1	SDMA[12]/SDBA1
15	AF19	debug_addr[14]/FTP[0]	16	N1	SDMA[11]
17	AF17	debug_addr[12]/FTP[1]	18	R1	SDMA[10]
19	AD26	debug_addr[11]/FTP[2]	20	R2	SDMA[9]
21	A22	debug_addr[10]/PLL_CFG[0]	22	T1	SDMA[8]
23	B19	debug_addr[9]/PLL_CFG[1]	24	T2	SDMA[7]
25	A21	debug_addr[8]/PLL_CFG[2]	26	U4	SDMA[6]
27	B18	debug_addr[7]/PLL_CFG[3]	28	U2	SDMA[5]
29	B17	debug_addr[6]/PLL_CFG[4]	30	U1	SDMA[4]
31	W26	debug_addr[5]/ \overline{GNT} [4]	32	V1	SDMA[3]
33	Y26	debug_addr[4]/ \overline{REQ} [4]	34	V3	SDMA[2]
35	AF26	debug_addr[3]/PCI_CLK[4]	36	W1	SDMA[1]
37	C25	debug_addr[2]/FTP[3]	38	W2	SDMA[0]
Even Cable			Odd Cable		
Logic Analyzer Pod 2			Logic Analyzer Pod 1		

* The Motorola naming convention for the SDMA and debug_addr signals is now little-endian.

Designing Logic Analyzer Connectors into Your Target System

MICTOR Connector J2					
J2 Pin	BGA Pin	MPC8240 Signal	J2 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5	AD2	$\overline{\text{SDCAS}}$	6	AD1	$\overline{\text{SDRAS}}$
7	AF3	PAR/AR[0]	8	H1	$\overline{\text{FOE}}$
9	AE3	PAR/AR[1]	10	AA1	$\overline{\text{WE}}$
11	G4	PAR/AR[2]	12	Y1	$\overline{\text{AS}}$
13	E2	PAR/AR[3]	14	Y4	RAS / $\overline{\text{CS}}$ [0]
15	AE4	PAR/AR[4]	16	AA3	RAS / $\overline{\text{CS}}$ [1]
17	AF4	PAR/AR[5]	18	AA4	RAS / $\overline{\text{CS}}$ [2]
19	D2	PAR/AR[6]	20	AC4	RAS / $\overline{\text{CS}}$ [3]
21	C2	PAR/AR[7]	22	M2	RAS / $\overline{\text{CS}}$ [4]
23	AB1	CAS / $\overline{\text{DQM}}$ [0]	24	L2	RAS / $\overline{\text{CS}}$ [5]
25	AB2	CAS / $\overline{\text{DQM}}$ [1]	26	M1	RAS / $\overline{\text{CS}}$ [6]
27	K3	CAS / $\overline{\text{DQM}}$ [2]	28	L1	RAS / $\overline{\text{CS}}$ [7]
29	K2	CAS / $\overline{\text{DQM}}$ [3]	30	N4	$\overline{\text{RCS0}}$
31	AC1	CAS / $\overline{\text{DQM}}$ [4]	32	N2	$\overline{\text{RCS1}}$
33	AC2	CAS / $\overline{\text{DQM}}$ [5]	34	AF2	MAA[0]
35	K1	CAS / $\overline{\text{DQM}}$ [6]	36	AF1	MAA[1]
37	J1	CAS / $\overline{\text{DQM}}$ [7]	38	AE1	MAA[2]
Even Cable			Odd Cable		
Logic Analyzer Pod 4			Logic Analyzer Pod 3		

MICTOR Connector J3					
J3 Pin	BGA Pin	MPC8240 Signal	J3 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5			6		
7	B1	DL[16]	8	E4	DH[16]
9	A1	DL[17]	10	A2	DH[17]
11	A3	DL[18]	12	B3	DH[18]
13	A4	DL[19]	14	D4	DH[19]
15	A5	DL[20]	16	B4	DH[20]
17	A6	DL[21]	18	B5	DH[21]
19	A7	DL[22]	20	D6	DH[22]
21	D7	DL[23]	22	C6	DH[23]
23	A8	DL[24]	24	B7	DH[24]
25	B8	DL[25]	26	C9	DH[25]
27	A10	DL[26]	28	A9	DH[26]
29	D10	DL[27]	30	B10	DH[27]
31	A12	DL[28]	32	A11	DH[28]
33	B11	DL[29]	34	A13	DH[29]
35	B12	DL[30]	36	B13	DH[30]
37	A14	DL[31] (LSB)	38	A15	DH[31]
Even Cable			Odd Cable		
Logic Analyzer Pod 6			Logic Analyzer Pod 5		

Designing Logic Analyzer Connectors into Your Target System

MICTOR Connector J4					
J4 Pin	BGA Pin	MPC8240 Signal	J4 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5			6		
7	AD17	DL[0]	8	AC17	DH[0] (MSB)
9	AE17	DL[1]	10	AF16	DH[1]
11	AE15	DL[2]	12	AE16	DH[2]
13	AF15	DL[3]	14	AE14	DH[3]
15	AC14	DL[4]	16	AF14	DH[4]
17	AE13	DL[5]	18	AC13	DH[5]
19	AF13	DL[6]	20	AE12	DH[6]
21	AF12	DL[7]	22	AE11	DH[7]
23	AF11	DL[8]	24	AE10	DH[8]
25	AF10	DL[9]	26	AE9	DH[9]
27	AF9	DL[10]	28	AE8	DH[10]
29	AD8	DL[11]	30	AC7	DH[11]
31	AF8	DL[12]	32	AE7	DH[12]
33	AF7	DL[13]	34	AE6	DH[13]
35	AF6	DL[14]	36	DH[14]	DH[14]
37	AE5	DL[15]	38	DH[15]	DH[15]
Even Cable			Odd Cable		
Logic Analyzer Pod 8			Logic Analyzer Pod 7		

MITCOR Connector J5					
J5 Pin	BGA Pin	MPC8240 Signal	J5 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5		nc	6	AC25	PCI_CLK[0]
7		nc	8		nc
9	AD18	PMAA[0]	10		Pull up Vdd (TTL)
11	AF18	PMAA[1]	12		Pull up Vdd (TTL)
13	AE19	PMAA[2]	14		Pull up Vdd (TTL)
15	AA25	$\overline{\text{REQ}}[1]$	16	AC26	$\overline{\text{INTA}}$
17	W23	$\overline{\text{GNT}}[1]$	18		Pull up Vdd (TTL)
19	K26	$\overline{\text{TRDY}}$	20	P25	C/ $\overline{\text{BE}}[3]$
21	J24	$\overline{\text{FRAME}}$	22	K23	C/ $\overline{\text{BE}}[2]$
23	K25	$\overline{\text{IRDY}}$	24	F23	C/ $\overline{\text{BE}}[1]$
25		GND	26	A25	C/ $\overline{\text{BE}}[0]$
27		Pull up Vdd (TTL)	28	H26	$\overline{\text{DEVSEL}}$
29	AB26	$\overline{\text{REQ}}[0]$	30	H25	$\overline{\text{STOP}}$
31	V26	$\overline{\text{GNT}}[0]$	32	J26	$\overline{\text{LOCK}}$
33	P26	IDSEL	34	G26	$\overline{\text{PERR}}$
35		Pull up Vdd (TTL)	36	F26	$\overline{\text{SERR}}$
37		Pull up Vdd (TTL)	38	G25	PAR
Even Cable			Odd Cable		

Designing Logic Analyzer Connectors into Your Target System

MICTOR Connector J6					
J6 Pin	BGA Pin	MPC8240 Signal	J6 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5		nc	6		nc
7	V25	AD[31]	8	F24	AD[15]
9	U25	AD[30]	10	E26	AD[14]
11	U26	AD[29]	12	E25	AD[13]
13	U24	AD[28]	14	E23	AD[12]
15	U23	AD[27]	16	D26	AD[11]
17	T25	AD[26]	18	D25	AD[10]
19	T26	AD[25]	20	C26	AD[9]
21	R25	AD[24]	22	A26	AD[8]
23	R26	AD[23]	24	B26	AD[7]
25	N26	AD[22]	26	A24	AD[6]
27	N25	AD[21]	28	B24	AD[5]
29	N23	AD[20]	30	D19	AD[4]
31	M26	AD[19]	32	B23	AD[3]
33	M25	AD[18]	34	B22	AD[2]
35	L25	AD[17]	36	D22	AD[1]
37	L25	AD[16]	38	C22	AD[0]
Even Cable			Odd Cable		

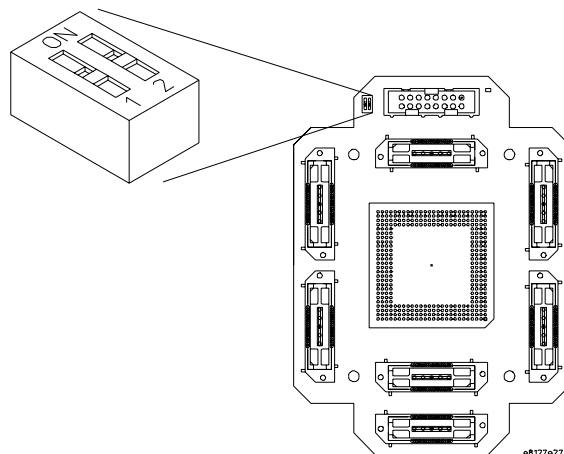
MICTOR Connector J7					
J7 Pin	BGA Pin	MPC8240 Signal	J7 Pin	BGA Pin	MPC8240 Signal
1		nc	2		nc
3		nc	4		nc
5		*	6		*
7		nc	8	B16	$\overline{\text{SRESET}}$
9		nc	10	B20	$\overline{\text{SUSPEND}}$
11		nc	12	B14	TBEN
13		nc	14	AF22	TCK
15		nc	16	AF23	TDI
17		nc	18	AC21	TDO
19		nc	20	AE22	TMS
21		nc	22	AE23	$\overline{\text{TRST}}$
23		nc	24	C19	IRQ_0 / S_INT
25		nc	26	B21	IRQ_1 / S_CLK
27		nc	28	AC22	IRQ_2 / S_RST
29	A20	$\overline{\text{HRST_CTRL}}$	30	AE24	IRQ_3 / S_FRAME
31	A19	$\overline{\text{HRST_CPU}}$	32	A23	IRQ_4 / LINT
33	A17	$\overline{\text{MCP}}$	34	AE20	SDA*
35	D16	NMI	36	AF21	SCL*
37	A18	$\overline{\text{SMI}}$	38	D14	$\overline{\text{CHKSTOP_IN}}$
Even Cable			Odd Cable		
Notes:					
* You may place the SDA and SCL on pins 5 and 6, which are clock lines.					

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 MPC8240 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 Motorola MPC8240 User's Guide* for information on designing a debug port.

The DIP switches configure signals to the JTAG port. To set the analysis probe DIP switches:

Switch	Description	OFF (default)	ON
1	JTAG Control	Microprocessor's JTAG port is isolated to the analysis probe's JTAG port. (default)	Microprocessor's JTAG port is connected to both the analysis probe and the target system's JTAG chain.
2	$\overline{\text{HRST}}$ Connection	$\overline{\text{HRST_CPU}}$ is not connected to $\overline{\text{HRST_CTRL}}$.	$\overline{\text{HRST_CPU}}$ is connected to $\overline{\text{HRST_CTRL}}$. The HRESET signal on the JTAG port will control both of these signals together. (default)



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 the Agilent Technologies 16600/700-series logic analysis systems

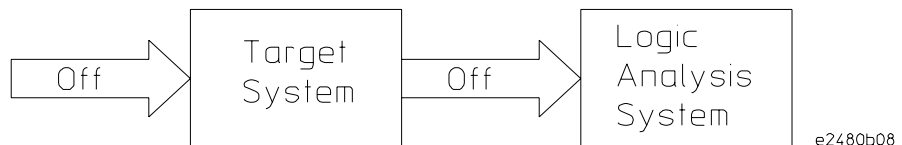
Ensure the target system is powered off.

- 1 Turn on the logic analyzer. The Setup Assistant will guide you through the process of connecting and configuring the logic analyzer.
- 2 When the logic analyzer is connected to the target system, and everything is configured, turn on 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 (if applicable) and software.

CAUTION:

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

Refer to the Agilent Technologies 16600/700-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.

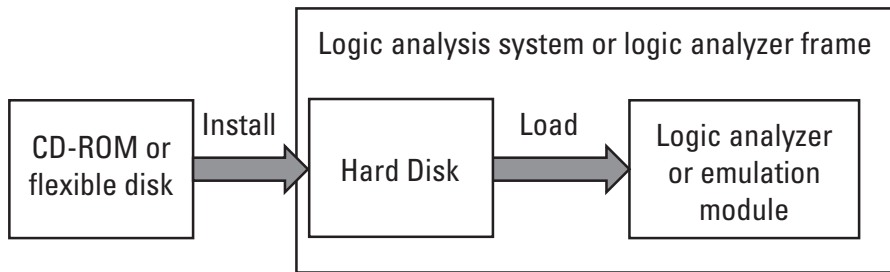
If you ordered an emulation module separately, use the information provided in the *Emulation for the Motorola MPC8240 User's Guide* to install your emulation module.

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

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 the software from CD-ROM

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 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 “**MPC82XX**” package.

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

- 9 Select **Install**.

The Continue dialog box will appear.

- 10 Select **Continue**.

Installing Software

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/mpc82xx/mpc8240/`.
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.

Probing the Target System

Connecting the Logic Analyzer to the Target System

Each table on the following pages corresponds to a particular type of analysis for 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 18.

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.

There are three types of analysis available:

64-bit data

This type of analysis captures all of the data signals through the data bus. Use the appropriate page, listed below, for your logic analyzer. The configuration file names are included with the connection diagrams. Connectors J1 through J4 are required for inverse assembly. Connector J7 contains additional signals you might want to monitor.

- 16550A logic analyzer (two cards)—page 65
- 16554/55/56/57 logic analyzers (two cards)—page 68
- 16600A logic analysis system—page 70
- 16601A logic analysis system—page 72
- 16710/11/12A logic analyzers (two cards)—page 75
- 16715/16/17/18/19A logic analyzers (two cards)—page 78
- 16750/51/52A logic analyzers (two cards)—page 78

32-bit data

This type of analysis will allow you to trace the 32 bits of DATA only, not DATA_B. The connections and configuration files are the same as for 64-bit data.

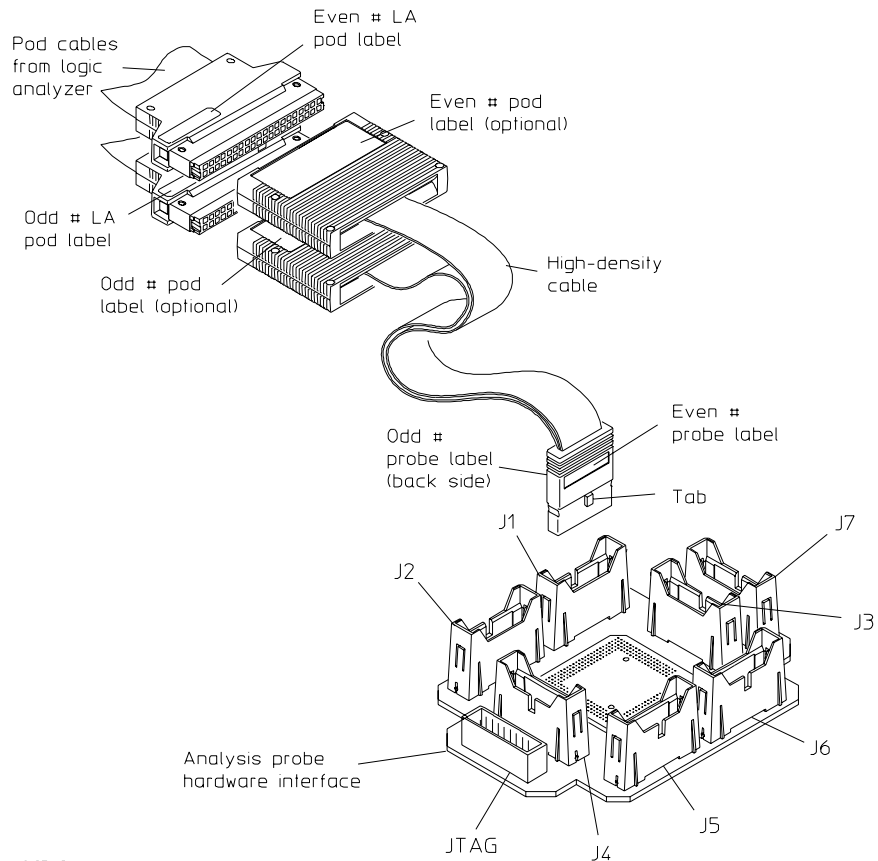
No data

This type of analysis allows you to trace program flow only using J1 and J2. Opcodes in the inverse assembly listing will be taken from an S-Record file, not from the data bus. Use the appropriate page listed below for your logic analyzer. The configuration file names are shown in the table on page 88.

- 16550A logic analyzer—page 67
- 16554/55/56/57 logic analyzers —page 69
- 16600A logic analysis system—page 71
- 16601A logic analysis system—page 73
- 16602A logic analysis system—page 74
- 16603A logic analysis system—page 74
- 16710/11/12A logic analyzers—page 77
- 16715/16/17/18/19A logic analyzers—page 79
- 16750/51/52A logic analyzers—page 79

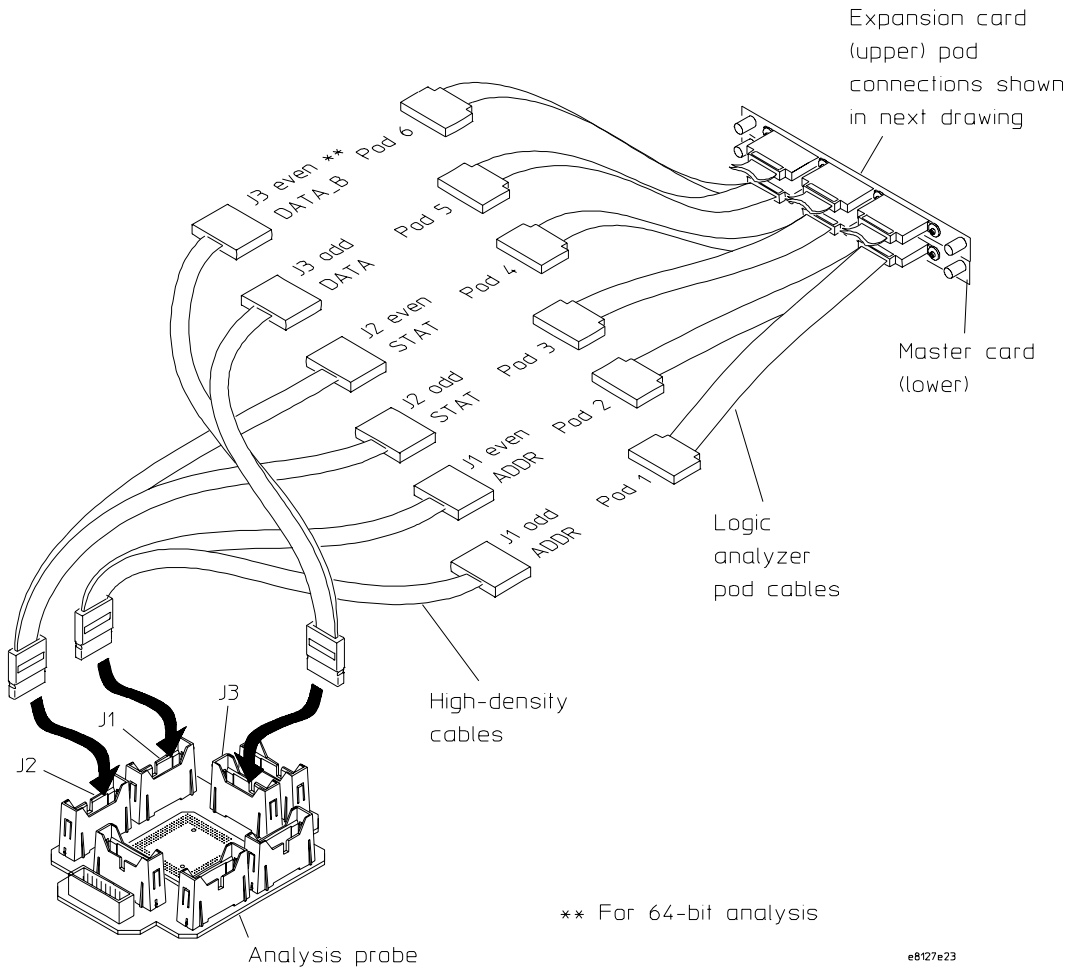
To connect the high-density termination cables to the analysis probe

Four Agilent Technologies E5346A high-density termination cables, and labels to identify them, are included with the analysis probe. 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.

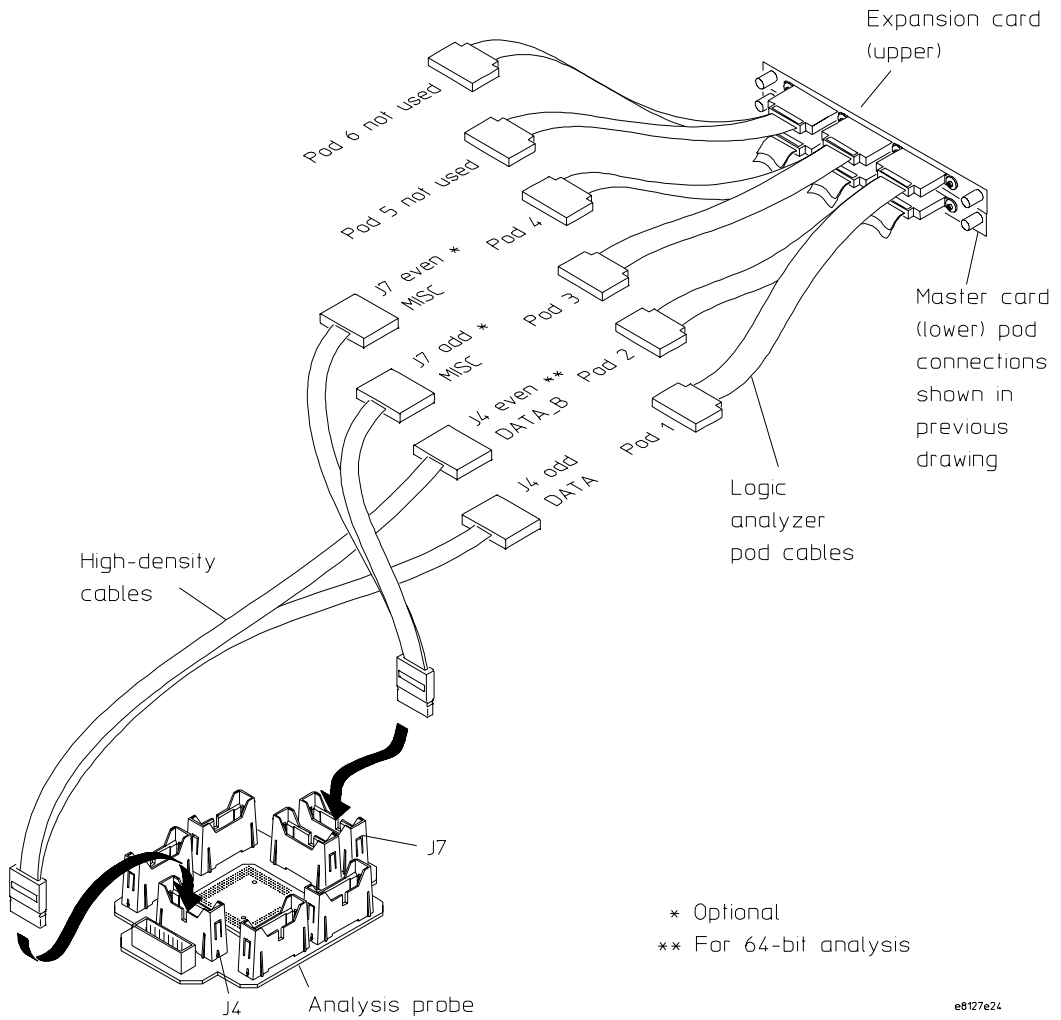


To connect a two-card 16550A logic analyzer for 64-bit or 32-bit data analysis

Use the following figures to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

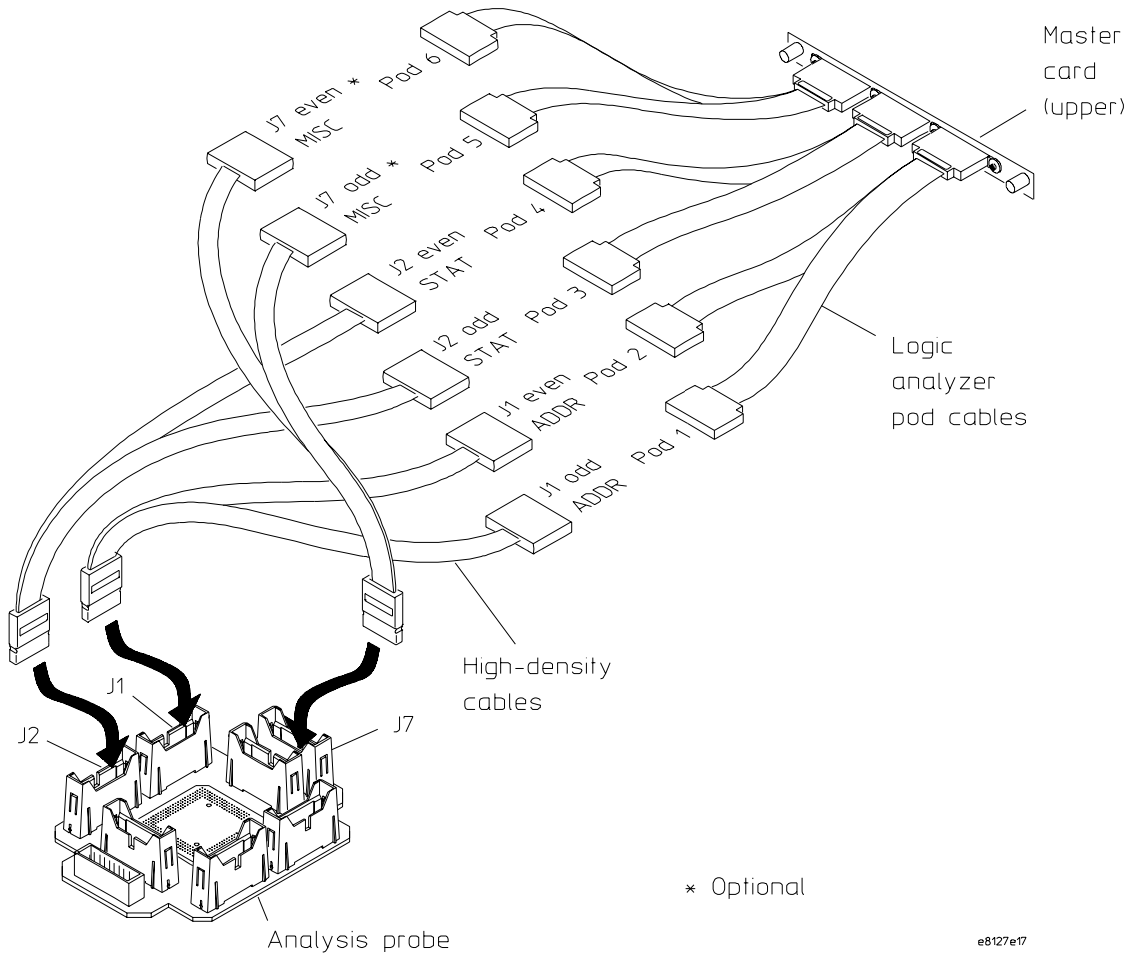


Chapter 4: Probing the Target System
Connecting the Logic Analyzer to the Target System



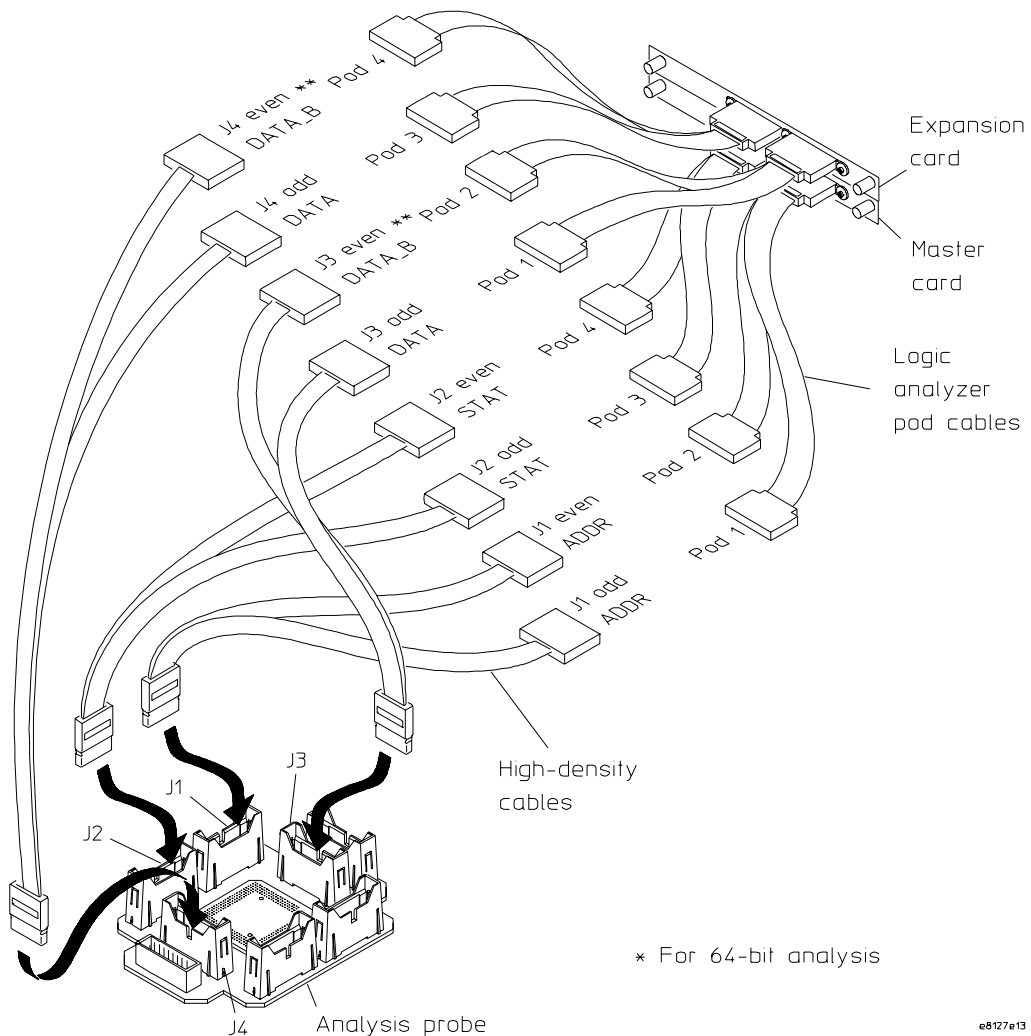
To connect a 16550A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



To connect a two-card 16554/55/56/57 logic analyzer for 64-bit or 32-bit data analysis

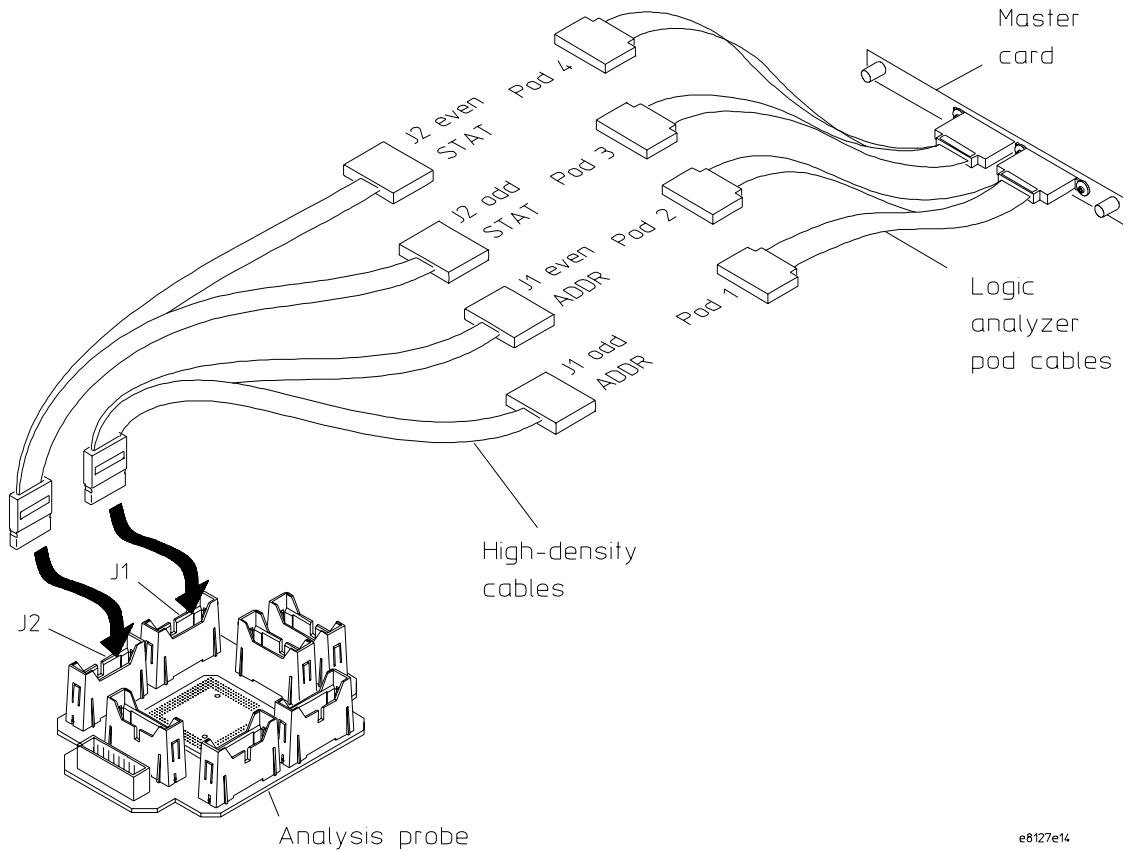
Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



e8127e13

To connect a 16554/55/56/57 logic analyzer for no-data analysis

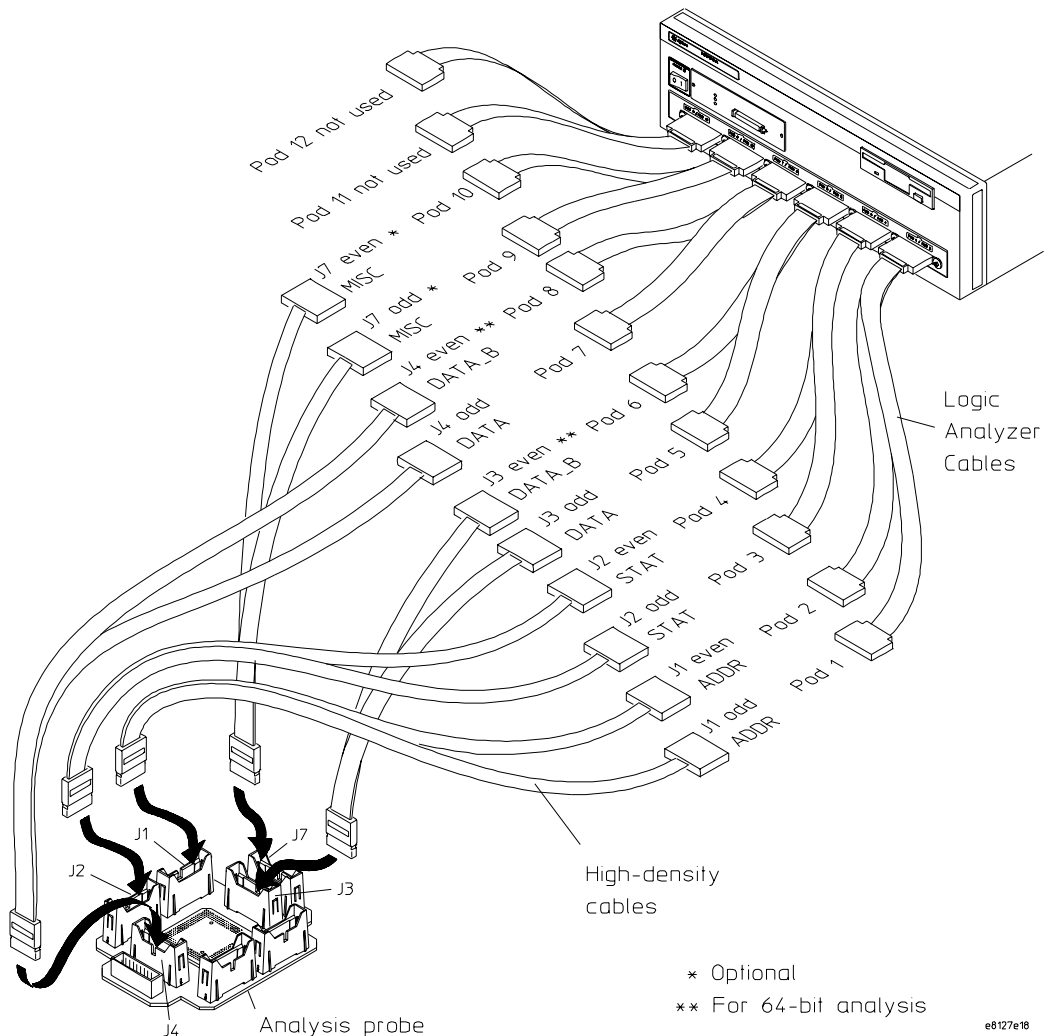
Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



e8127e14

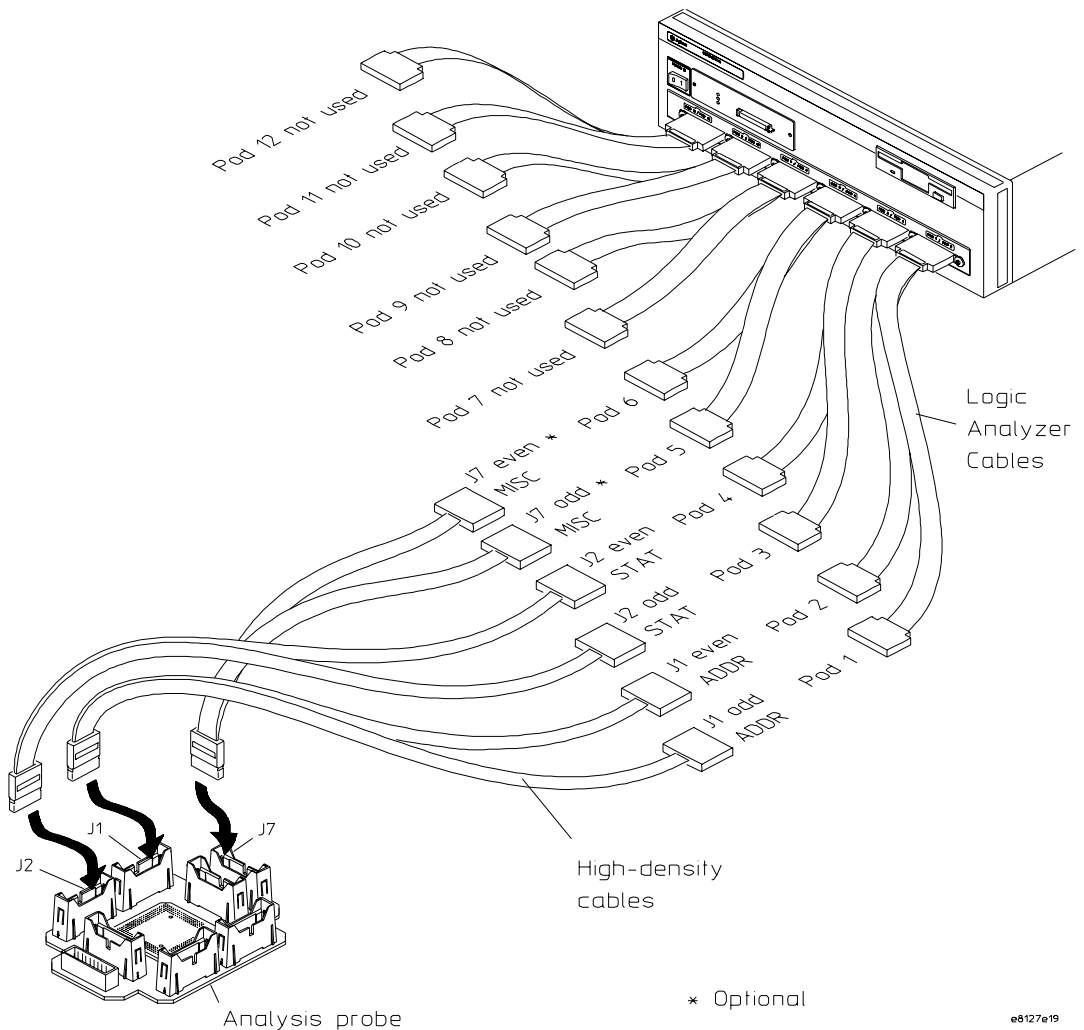
To connect the 16600A logic analyzer for 64-bit or 32-bit data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



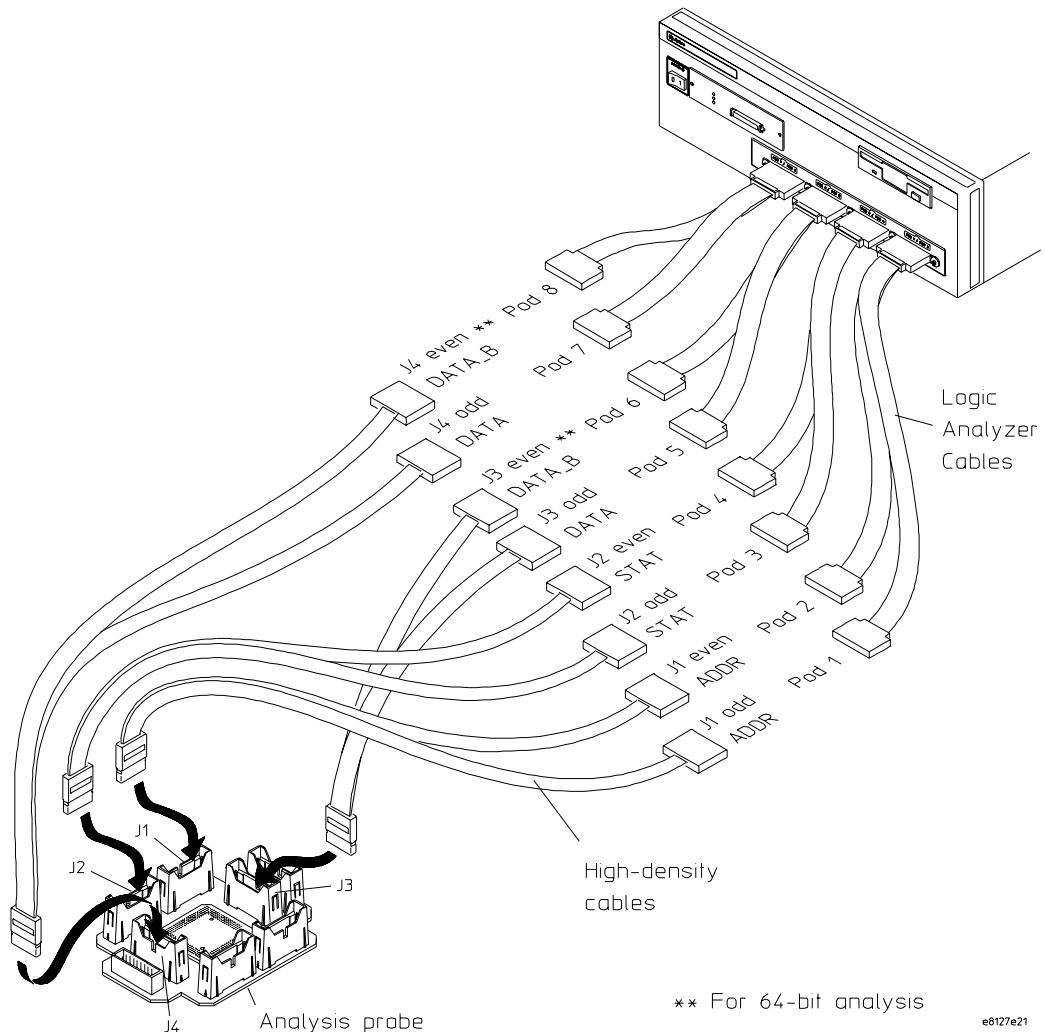
To connect the 16600A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



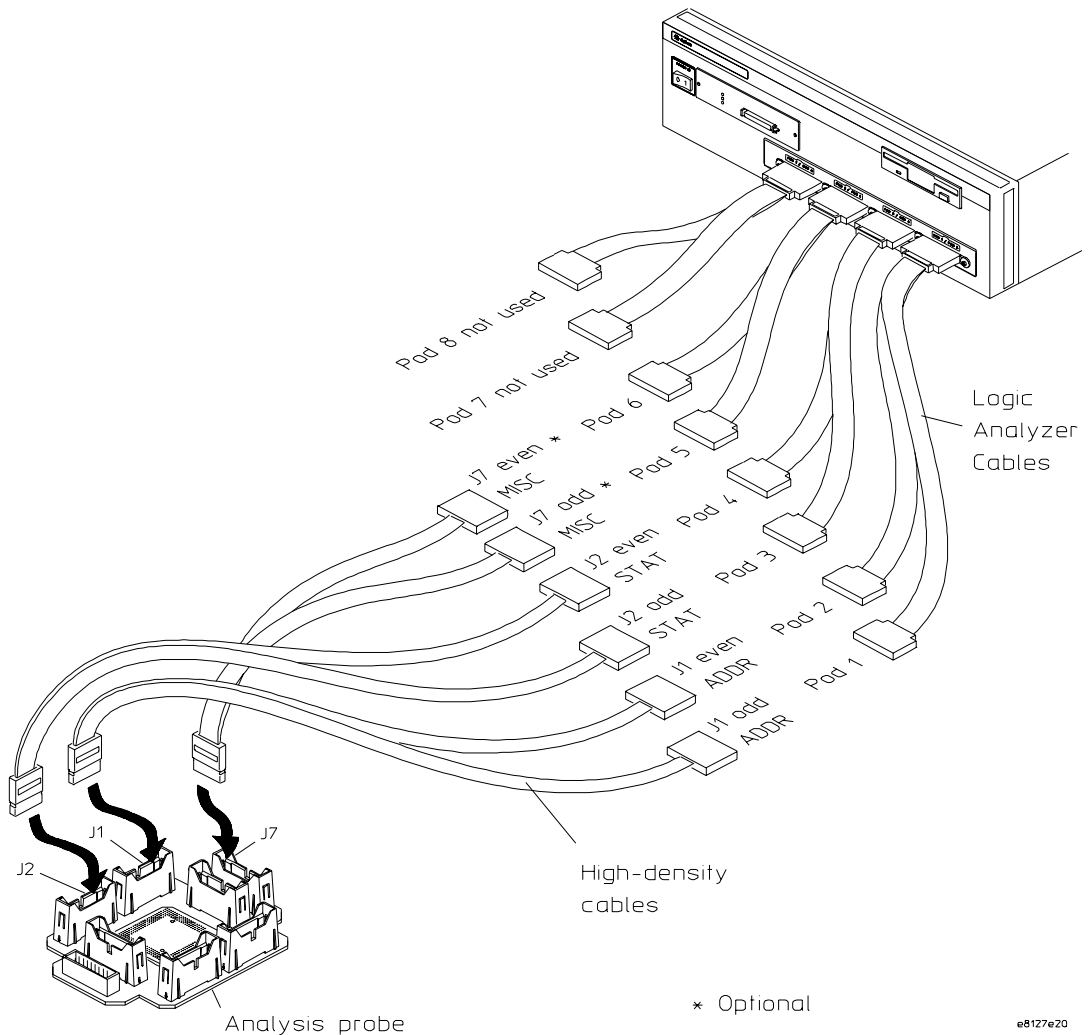
To connect the 16601A logic analyzer for 64-bit or 32-bit data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



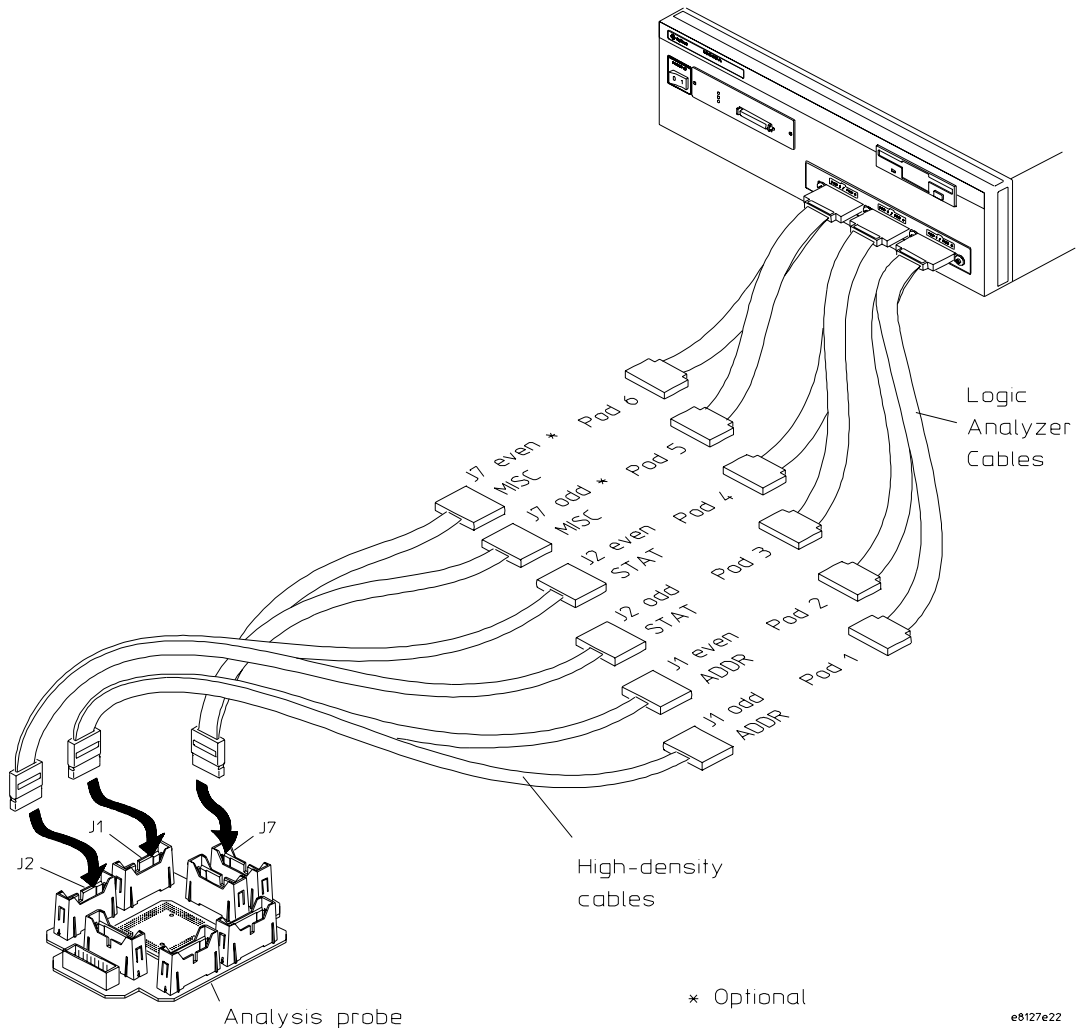
To connect the 16601A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



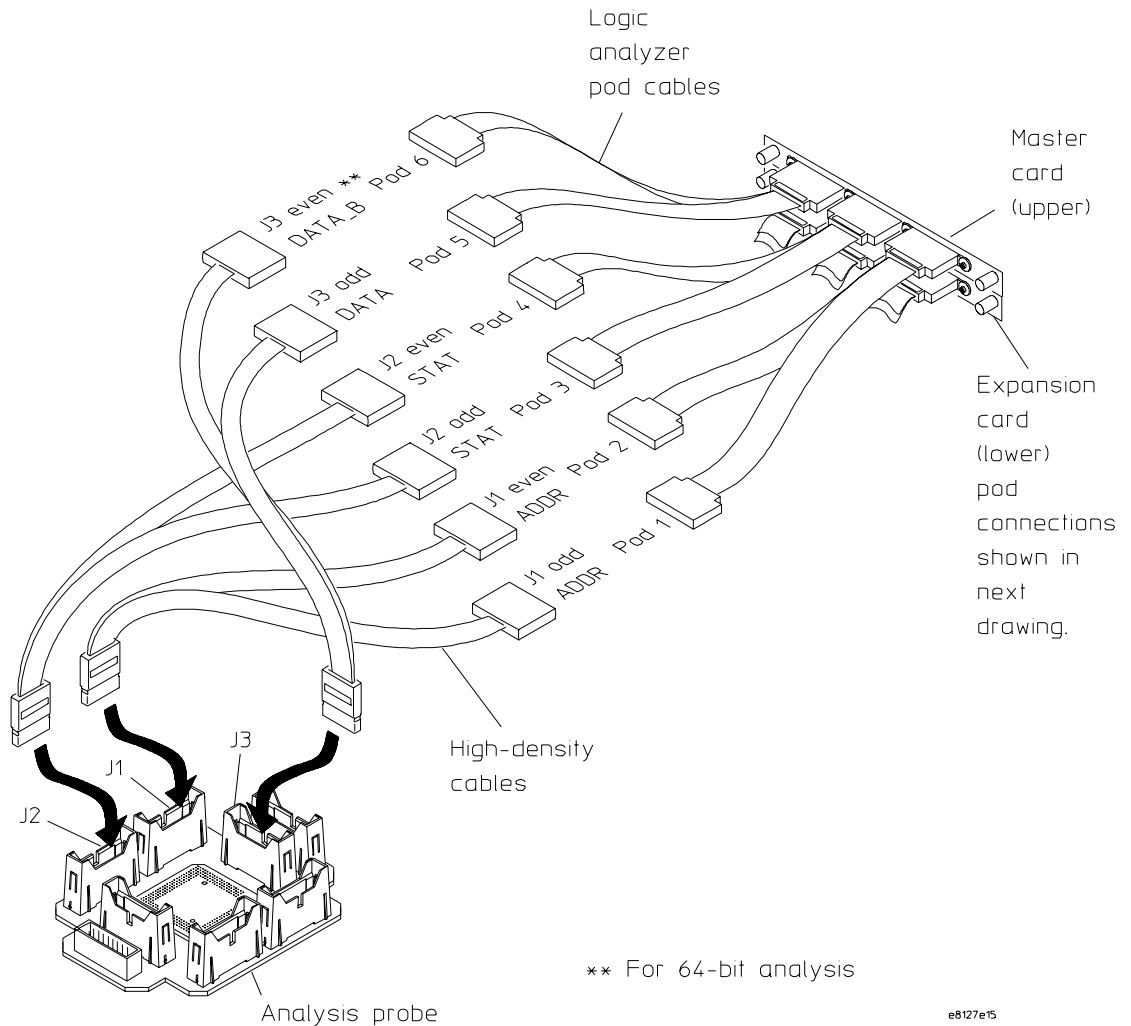
To connect the 16602/3A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

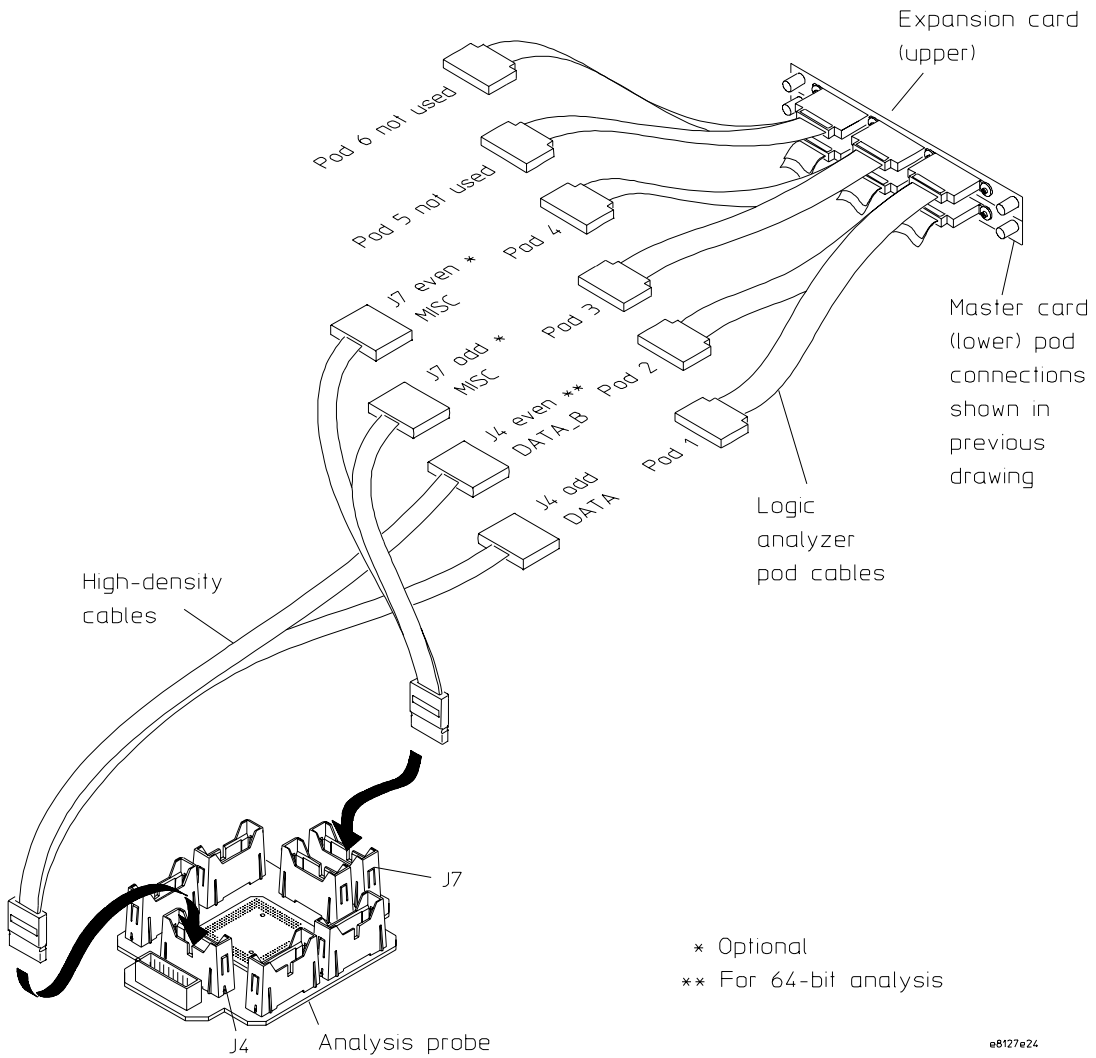


To connect a two-card 16710/11/12A or logic analyzer for 64-bit or 32-bit data analysis

Use the following figures to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

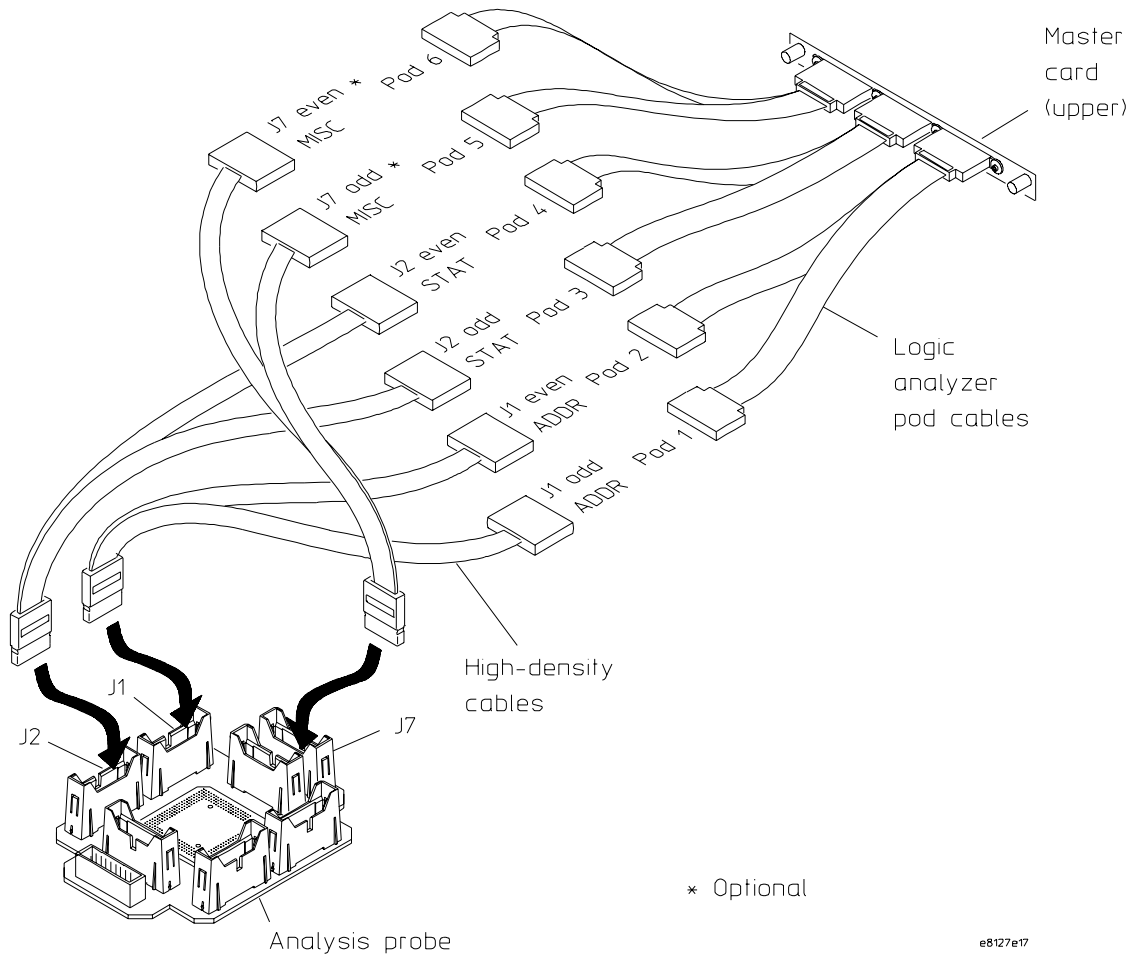


Chapter 4: Probing the Target System
Connecting the Logic Analyzer to the Target System



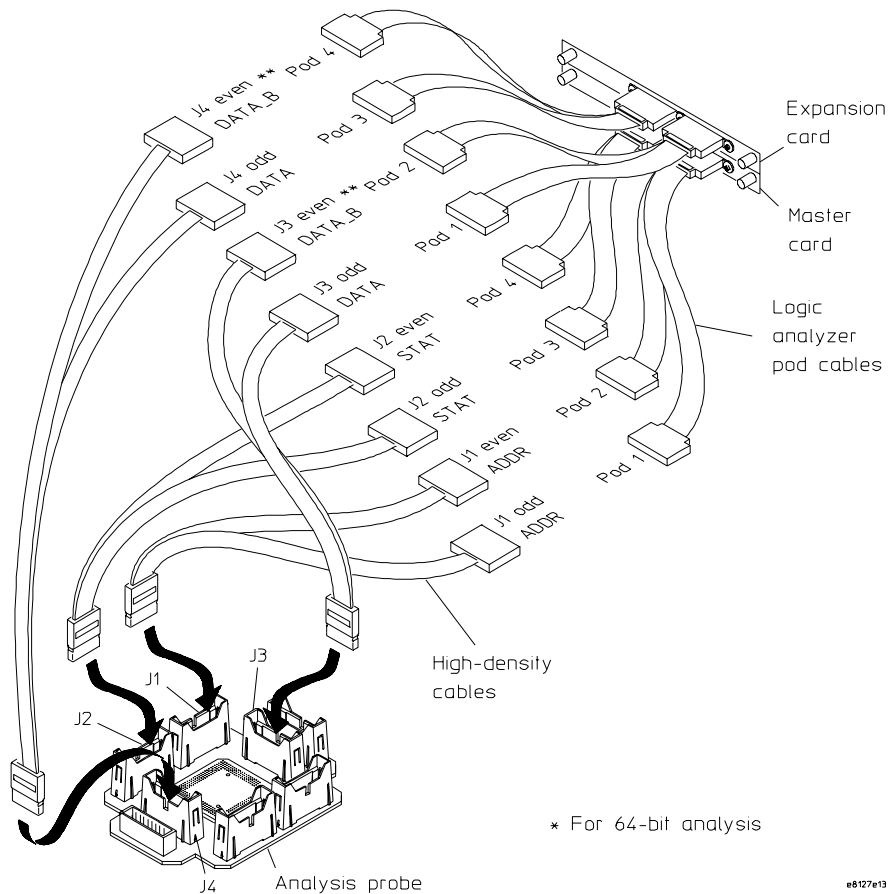
To connect a 16710/11/12A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



To connect a two-card 16715/16/17/18/19A or 16750/51/52A logic analyzer for 64-bit or 32-bit data analysis

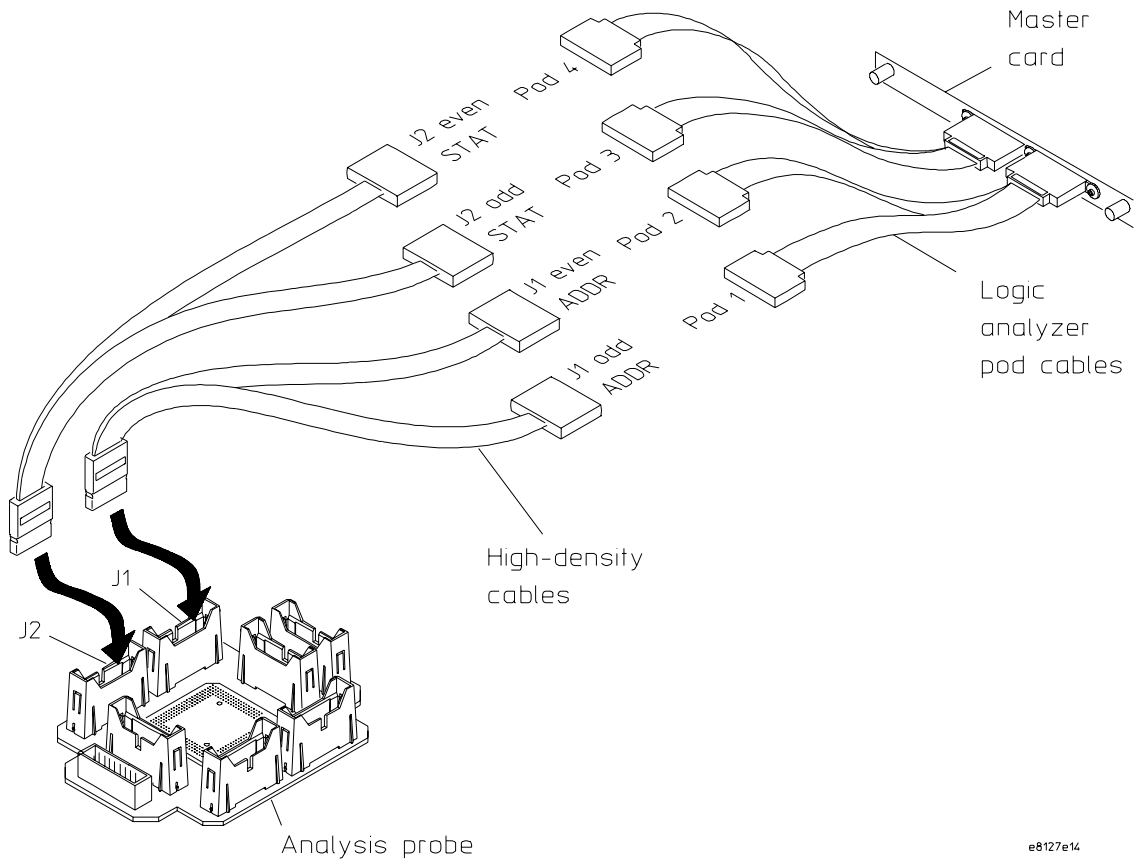
Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



e8127e13

To connect a 16715/16/17/18/19A or 16750/51/52A logic analyzer for no-data analysis

Use the following figure to connect cables from the logic analyzer to the connectors on the analysis probe or on the target system. Find the labels that were shipped with the high-density cables and use them to help identify the connections.

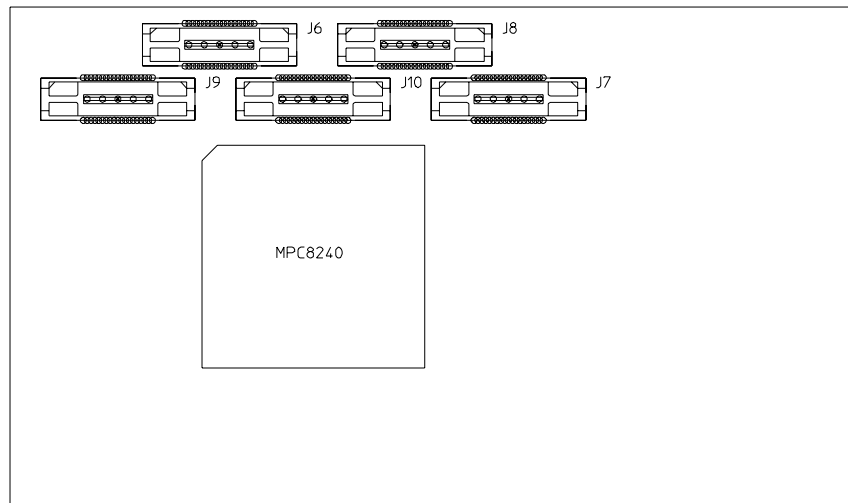


e8127e14

Connecting the Logic Analyzer to a Motorola PMC8240 Board

Disconnect power from the logic analyzer and your target system before you make or break connections.

This section shows the connections between the logic analyzer pod cables and the connectors on the PMC8240 board. The 32-bit and no-data analysis options are not available for this board.



e8127e04

To connect a two-card 16710/11/12A or 16550A logic analyzer to a PMC8240 board

	Pod 6	Pod 5	Pod 4	Pod 3	Pod 2	Pod 1
Expansion Card 1	unused	unused	unused	unused	J6, Even	J6, Odd
Master Card	J9, Even	J9, Odd	J10, Even	J10, Odd	J7, Even	J7, Odd

To connect a 16600A or 16601A logic analyzer to a PMC8240 board

Pod 8	Pod 7	Pod 6	Pod 5	Pod 4	Pod 3	Pod 2	Pod 1
J6, Even	J6, Odd	J9, Even	J9, Odd	J10, Even	J10, Odd	J7, Even	J7, Odd

To connect a two-card 16554/55/56/57 logic analyzer to a PMC8240 board

	Pod 4	Pod 3	Pod 2	Pod 1
Expansion Card 1	J6, Even	J6, Odd ¹	J9, Even	J9, Odd
Master Card	J10, Even	J10, Odd	J7, Even	J7, Odd

To connect a two-card 16715/16/17/18/19A or 16750/51/52A logic analyzer to a PMC8240 board

	Pod 4	Pod 3	Pod 2	Pod 1
Expansion Card 1	J6, Even	J6, Odd ¹	J9, Even	J9, Odd
Master Card	J10, Even	J10, Odd	J7, Even	J7, Odd

Configuring the Logic Analyzer

Chapter 5: Configuring the Logic Analyzer

The sections of this chapter describe setting up and using the MPC8240 inverse assembler. Because your MPC8240 target system is designed uniquely according to your needs, it is important that you specify the available signals and memory regions to the inverse assembler.

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

- Loading the configuration file and the inverse assembler
- Tables showing configuration file names
- Using the inverse assembler
- Setting the inverse assembler preferences
- Symbols
- Changing analysis mode

Configuring 16600/16700-series Logic Analysis Systems

You configure the logic analyzer by loading a configuration file. Normally this is done using the Setup Assistant (see page 18). 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, how signals are routed to the connectors in the target system, and the type of analysis (64-bit data, 32-bit data, or no-data).

The MPC8240 inverse assembler decodes captured data into software addresses (SW_ADDR label) and assembly language mnemonics.

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/mpc82xx/mpc8240/` exists.

If the above directory does not exist, you need to install the MPC8240 Processor Support Package. Close File Manager, then use the procedure on the CD-ROM jacket to install the MPC8260 Processor Support Package before you continue. See “To install the software from CD-ROM” on page 59 for details.

- 2** Using File Manager, select the configuration file you want to load in the `/logic/configs/hp/mpc82xx/mpc8240/` 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 MPC8240 analysis by loading the appropriate MPC8240 configuration file. Loading the indicated file also automatically loads the inverse assembler. The configuration file names are shown in the table on page 88.

- 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

The following table shows the configuration files for the supported logic analyzers.

Analyzer Model	64/32-bit Configuration File	No-data Configuration File	PMC8240 Configuration File
16550A	c8240F_2	c8240F_1	cPMC8240F_2
16554A	c8240M_2	c8240M_1	cPMC8240M_2
16555A/D	c8240M_2	c8240M_1	cPMC8240M_2
16556A/D	c8240M_2	c8240M_1	cPMC8240M_2
16557D	c8240M_2	c8240M_1	cPMC8240M_2
16600A	c8240F_2	c8240F_1	cPMC8240F_2
16601A	c8240F_2	c8240F_1	cPMC8240F_2
16602A	na	c8240F_1	na
16603A	na	c8240F_1	na
16710/11/12A	c8240F_2	c8240F_1	cPMC8240F_2
16715/16/17/18/19A	c8240L_2	c8240L_1	cPMC8240L_2
16750/51/52A	c8240L_2	c8240L_1	cPMC8240L_2

Using the Inverse Assembler

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

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

Using Cache-On Trace Reconstruction

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

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

Example branch exception routine:

```
0x00000d00:  mfspr    r7, d26
0x00000d04:  addis   r8, r0, 0x0000
0x00000d08:  stw    r7, 0x0100(r8)
0x00000d0C:  rfi
```

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

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

Chapter 5: Configuring the Logic Analyzer

Using the Inverse Assembler

Example program exception routine:

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

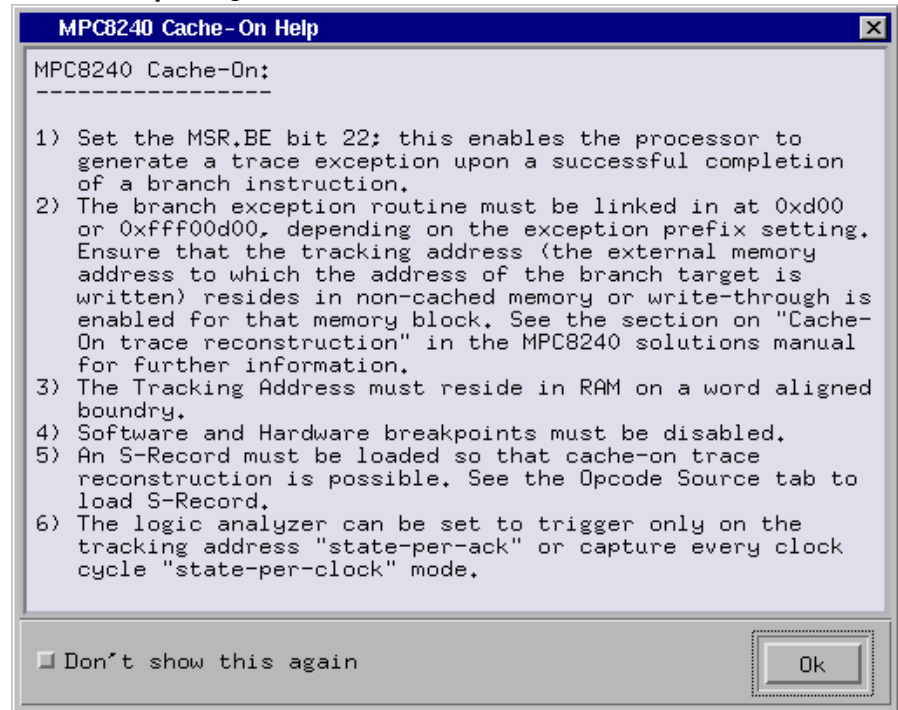
To enable cache-on trace reconstruction:

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

- 1 Set the cache-on mode
- 2 Set **data bus connected**
- 3 Provide the tracking address in the **Opcode Source** tab
- 4 Load an S-Record executable file

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

Cache-on help dialog



The **Don't show this again** button can be selected to prevent this dialog from appearing until the inverse assembler is reloaded.

Enabling branch exception disassembly

The following trace shows cache-on execution using branch trace exception disassembly. See page 89 for an explanation of this feature.

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

Cache-on trace, S-Record executable file loaded, data bus connected, tracking address 0x00001000

The screenshot shows the Inverse Assembler software interface. The main window displays a disassembly trace for state 0. The trace is organized into columns: State Number, SW_ADDR, and Mnemonics/Hex. The trace shows instructions for state 0, including stwu, mfspr, stw, and bl. The trace also shows the state number and the SW_ADDR for each instruction. The trace is displayed in a table format with a header row and a data row for each state.

State Number	SW_ADDR	Mnemonics/Hex
0	ABSOLUTE 00001500	stwu r1,0xffff(r1)
	ABSOLUTE 00001504	mfspr r0,d8
	ABSOLUTE 00001508	stw r0,0x000c(r1)
	ABSOLUTE 0000150C	bl 0x00001030
1		TS
2		Address only 0x00001500
3		ext fetch@<0x00001500>
		ext fetch@<0x00001504>
4		ext fetch@<0x00001508>
		ext fetch@<0x0000150c>
5		ext fetch@<0x00001510>
		ext fetch@<0x00001514>
6		ext fetch@<0x00001518>
		ext fetch@<0x0000151c>
7		TS
8	ABSOLUTE 0000BF00	mem write 0x0000bfb8
9	ABSOLUTE 0000BF0C	mem write 0x00003a30

Inverse Assembler Modes of Operation

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

Inverse Assembler Modes of Operation

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

NOTE:

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

To use the Invasm menu

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

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

Loading the Inverse Assembler

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

Setting Inverse Assembler Preferences

The Invasm Preferences dialog lets you give the inverse assembler information about your target system so that it can properly disassemble signal values captured by the logic analyzer.

Trigger Tool

The Preferences dialog also contains the Trigger Tool, which you will use when setting logic analyzer triggers for memory addresses. See “To trigger on an access to a memory address” on page 125.

To set the inverse assembler preferences

To open the Preferences dialog:

- 1** Load a configuration file, if needed.
- 2** Open the **Listing** window.
- 3** Select **Preferences...** from the **Invasm** menu at the top of the Listing window.

To set the Memory Map preferences

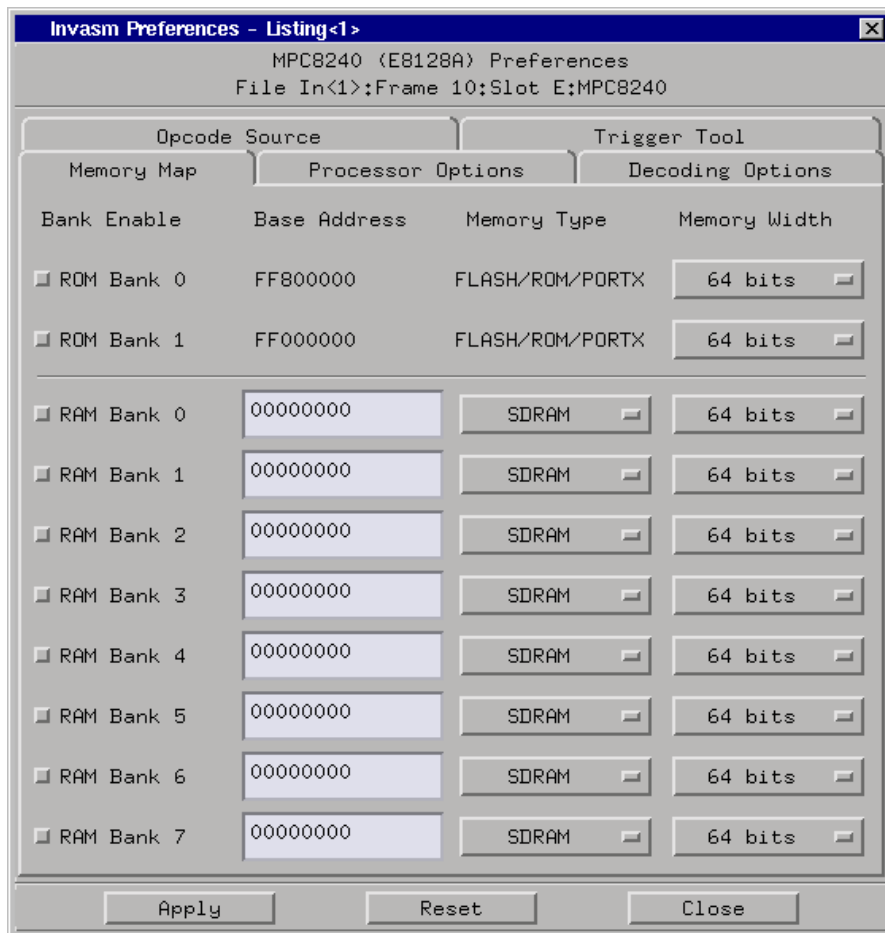
The MPC8240 does not provide all of the signals required to reconstruct physical addresses. It is therefore necessary to reconstruct the address from various pieces of information:

- Memory map configuration (set in the Preferences dialog)
- The 8 RAS_ and 8 CAS_ signals
- The 16 debug address signals (present when the MPC8240 debug mode is enabled)
- 11 SDMA signals

NOTE:

You can enable debug mode by setting WP_CONTROL (WP_DEBUG_) to zero or pulling down the GNT_[4] pin.

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



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

Setting Inverse Assembler Preferences

Bank Enable/Disable. Enable the banks for all of the chip selects that are being used in the system; otherwise, the inverse assembler may not function correctly. These enables are required because chip selects are active low and unconnected logic analyzer channels float low; without the enables, the inverse assembler cannot tell the difference between active and unconnected chip selects.

You can find out which banks are enabled by examining the Memory Bank Enable Register (MER). Each bit in this register corresponds to a memory bank.

Memory Bank Enable Register		
Register	Address	Size
Memory Bank Enable Register	0xA0	1 byte

bit 7	6	5	4	3	2	1	bit 0
bank 7	bank 6	bank 5	bank 4	bank 3	bank 2	bank 1	bank 0
1 = bank is enabled				0 = bank is disabled			

Examples: reading the Bank Enable register using the emulator's built-in command line interface:

Map A, Big Endian

```
m -a4 -d4 80000CF8=A0000080
m -a1 80000CFC
```

In this case, the gateway register is 0x80000CFC and the address register is 0x80000CF8. The first command sets the configuration register address to **A0** (the Memory Bank Enable Register). The second command displays the value of this 1-byte register.

Map B, Big Endian

```
m -a4 -d4 FEC00000=A0000080
m -a1 FEE00000
```

In this case, the gateway register is 0xFEE00000 and the address register is 0xFEC00000.

Base Address. Specifies the starting address of the memory bank.

You can find out the base address by examining the Memory Starting Address register and the Extended Memory Starting Address register. This can be done with either a debugger interface or the Emulation Control Interface in the 16600/700 logic analysis systems.

Memory Starting Address Registers		
Register	Address	Size
Memory Starting Address Register for banks 0-3	0x80	4 bytes
Memory Starting Address Register for banks 4-7	0x84	4 bytes
Extended Memory Starting Address Register for banks 0-3	0x88	4 bytes
Extended Memory Starting Address Register for banks 4-7	0x8C	4 bytes

These values are combined into a base address as follows

bit	31	30	29	28	27	20	19	0
	0	0	extended starting address		starting address		0x00000	

Examples: reading a Starting Address register using software

Map A, Big Endian

First set:

```
r0 = 0x80000080  
r1 = 0x80000CF8
```

Next, run this code:

```
stw r0,0(r1)  
lwz r2,4(r1)
```

This will set the address (0x80000CF8) to 0x80000080. r2 will be set to the value of the configuration register at 0x80 (the value might be something like 0x00102030).

Map B, Big Endian

First, set:

```
r0 = 0x80000080  
r1 = 0xFEC00000  
r2 = 0xFEE00000
```

Next, run this code:

```
stw r0,0(r1)  
lwz r3,0(r2)
```

This sets the address (0xFEC00000) to 0x80000080. r3 will be set to the value of the configuration register at 0x80.

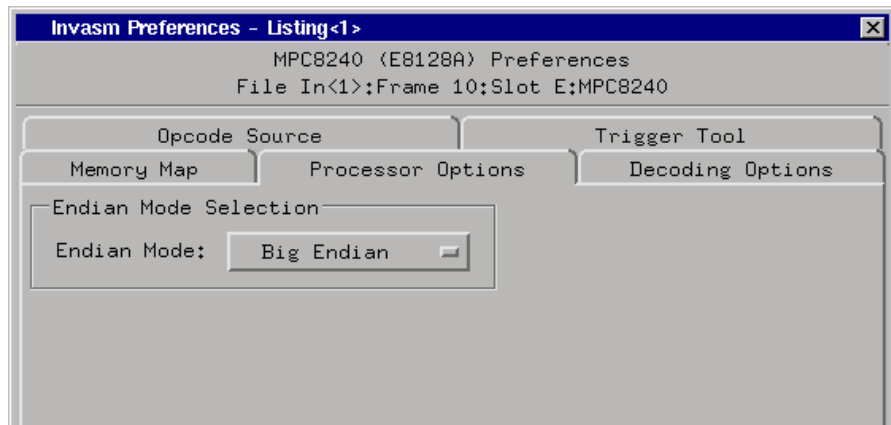
Memory Type. Depending upon the type of memory that is accessed, there are different signals that assert when a valid address is on the bus. For this reason, the inverse assembler must know what type of memory is being accessed.

You can find out the memory type for the RAM banks by examining bit 17 of the Memory Control Configuration register at 0xF0. “0” indicates SDRAM and “1” indicates FPM/EDO DRAM.

The MPC8240 does not support mixed DRAM/SDRAM configurations, so all of the memory banks will be configured to the same memory type.

To set the Processor Options preferences

The Processor Options tab of the Preferences dialog lets you tell the inverse assembler which mode the MPC8240 is operating in.



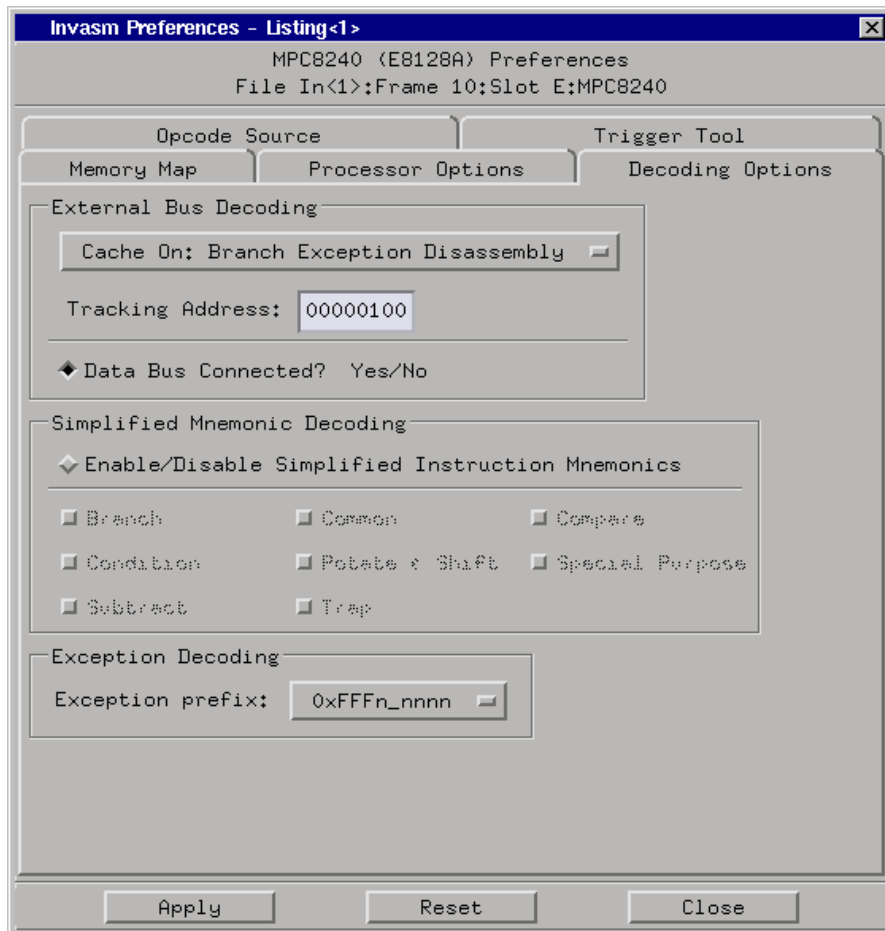
Endian Mode Selection

Big Endian Mode. Big endian mode is the native mode of the processor.

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

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

To set the Decoding Options preferences



External Bus Decoding. Choose **Cache Off: External Bus Disassembly** for traditional inverse assembly or **Cache On: Branch Exception Disassembly** for cache-on trace reconstruction, and provide the tracking address.

Data Bus Connected. Read and write states are always indicated regardless of whether the data bus is connected. However, when the data bus is connected, read/write data will also be displayed. See “Inverse Assembler Modes of Operation” on page 93

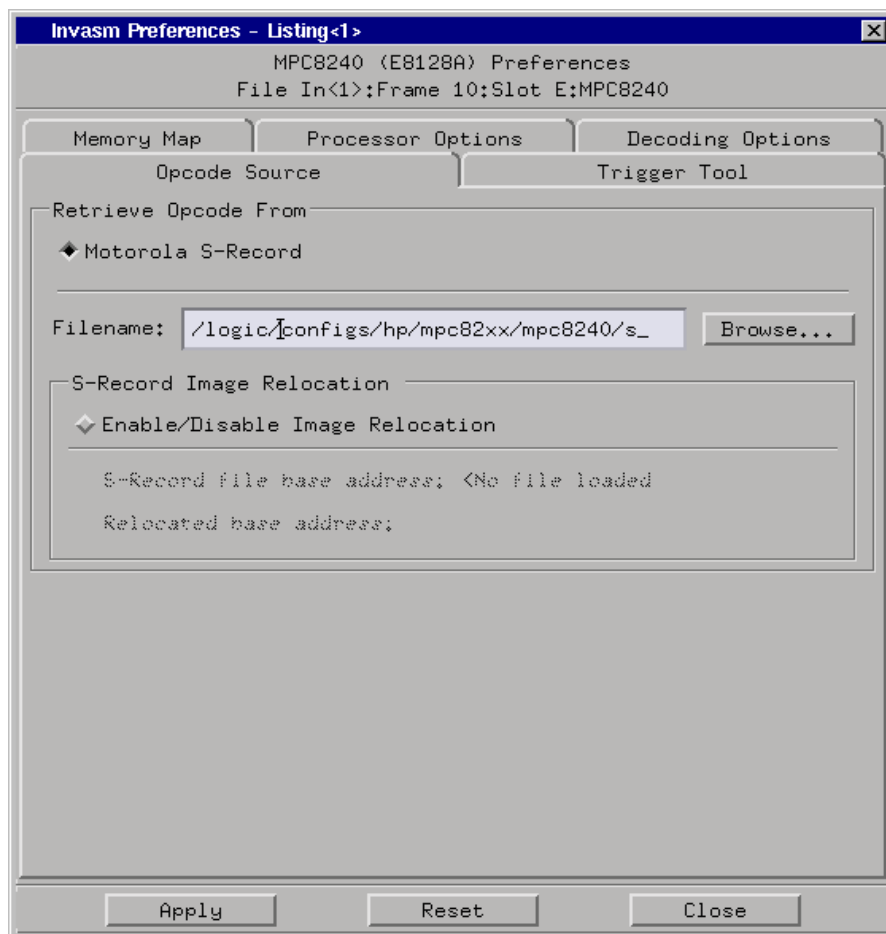
Setting Inverse Assembler Preferences

Simplified Mnemonic Decoding. PowerPC assemblers support a number of simplified mnemonics for some popular assembly language instructions, as described in the *PowerPC Programming Environments Guide* (Appendix E). The inverse assembler will show those extensions if you wish to see them. By enabling the Simplified Mnemonic Decoding, you can select which types of simplified mnemonics will be shown. Click the options for the simplified mnemonics you desire.

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

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

To set the Opcode Source preferences



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

S-Record Image Relocation. The Image Relocation portion of the dialog box allows you to relocate the SREC file to some other location in memory. This is useful when the loaded file is moved to some other location in memory.

Setting Inverse Assembler Preferences

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.

To enable/disable the instruction cache on the MPC8240

When the instruction cache is enabled, many PowerPC instructions are executed from the cache and do not appear on the external bus. Cache-on trace reconstruction is used to trace execution in cache.

To get an execution trace on the bus, the instruction cache can be disabled. This must be done in supervisor mode.

To disable the cache with the emulation module:

Use your debugger or the Emulation Control Interface to configure the HID0 register.

Register values for controlling the cache

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

To disable the cache with code:

- Disable the instruction cache with the following code:

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

- To also disable the data cache use:

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

Setting Inverse Assembler Preferences

- To invalidate and disable both caches use:

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

- Enable the instruction cache with the following code:

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

Loading Symbol Information

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 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.

To view predefined symbols for the MPC8240

User-defined symbols are symbols you create in the logic analyzer 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 logic analyzer configurations. The logic analyzer configuration files included with the MPC8240 inverse assembler contain predefined symbols for logic analyzer labels.

To display the predefined symbols for the MPC8240:

- 1** Open the logic analyzer's **Setup** window.
- 2** Select the **Symbol** tab.
- 3** Select the **User Defined** tab.
- 4** Choose a label name from the Label list.

The logic analyzer will display the symbols associated with the label.

Loading Symbol Information**Predefined Symbols**

Label	Value (hex)	Symbol
MAASRC*	8	mem read
	9	touch load
	A	inst fetch
	B	reserved read
	C	PCI mem read
	D	DMA0 mem read
	E	DMA1 mem read
	F	I2O mem read
	0	mem write
	1	reserved write
	2	reserved write
	3	reserved write
	4	PCI mem write
	5	DMA0 mem write
	6	DMA1 mem write
7	I2O mem write	
RAS_	7F	CS0
	BF	CS1
	DF	CS2
	EF	CS3
	F7	CS4
	FB	CS5
	FD	CS6
	FE	CS7
RCS0_	0	CS0
	1	1
RCS1_	0	CS1
	1	1
WE_	1	read
	0	write

* MAASRC is composed of the MAA and \overline{WE} signals.

To load 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. The object file containing symbolic debug information must be in a format the logic analyzer understands.

If your compiler generates object files in a format that the logic analyzer doesn't understand, you can use a General-Purpose ASCII (GPA) symbol file (see Chapter , "General-Purpose ASCII (GPA) Symbol File Format," on page 157).

To load symbols in the Agilent Technologies 16600A/16700A-series logic analysis system:

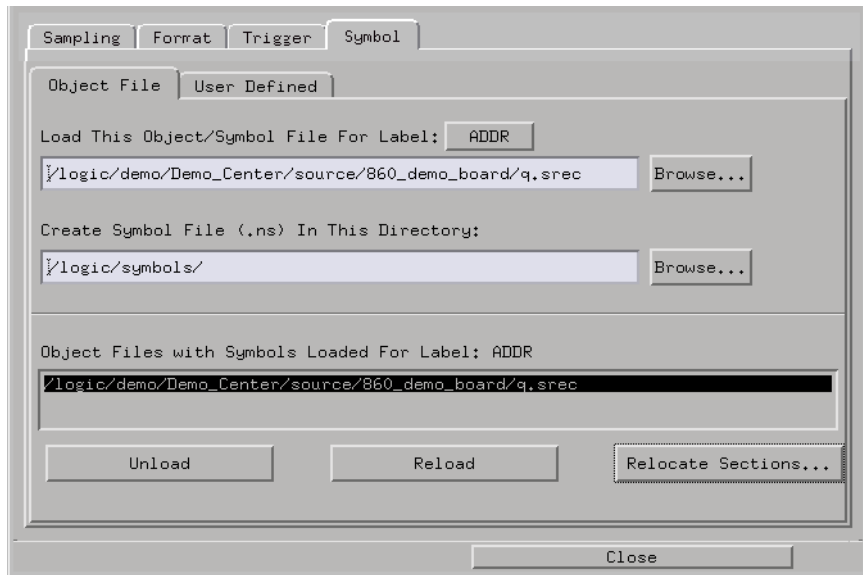
- 1** Open the logic analyzer module's **Setup** window.
- 2** Click the **Symbol** tab.
- 3** Click the **Object File** tab.

Make sure the label is ADDR.

Chapter 5: Configuring the Logic Analyzer

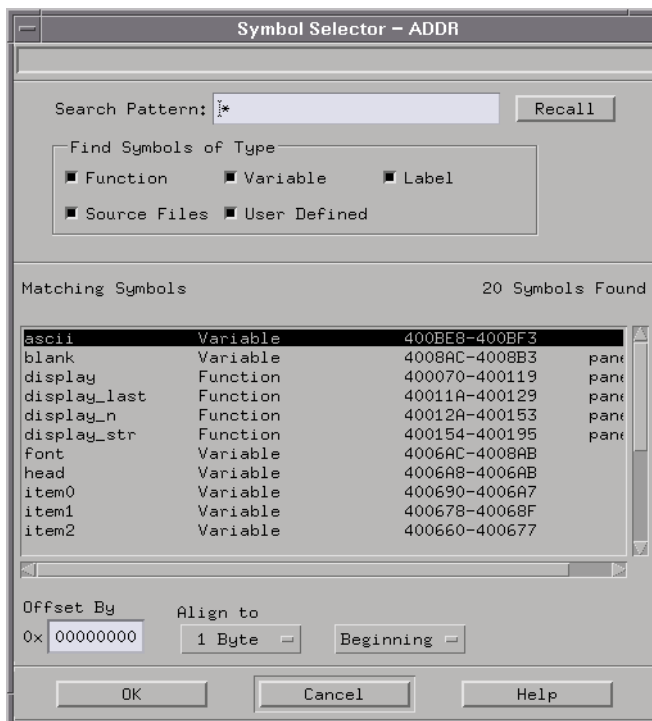
Loading Symbol Information

From this dialog you can select object files and load their symbol information.



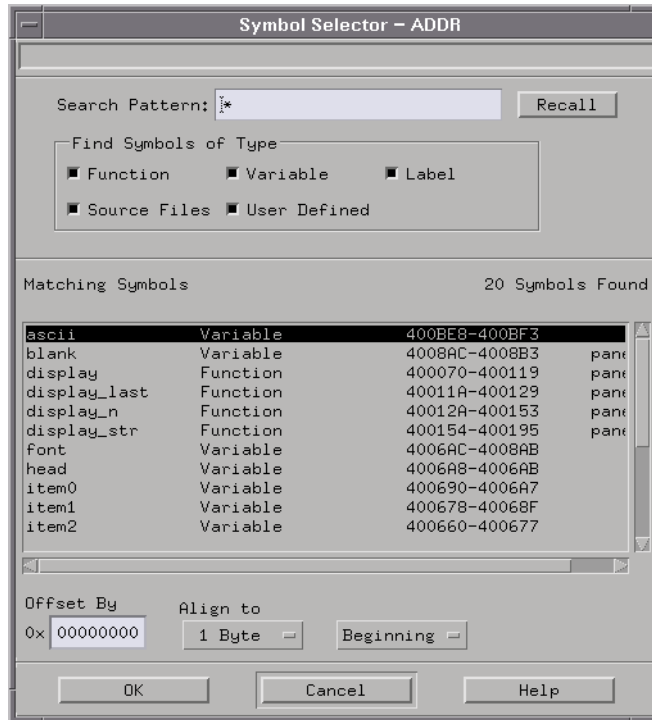
When you load object file symbols into a logic analyzer, a database of symbol/line number to address assignments is generated from the object file.

The Symbol Selector dialog allows you to use a symbol in place of a hexadecimal value when defining trigger patterns, trigger ranges, and so on. To select this dialog, select Trigger tab, then Pattern. Set the numeric base field to Symbols, then select the field to the right of the numeric base field.



To compensate for relocated code

When code segments are relocated, or when memory management units produce fixed code offsets, you can compensate by using the address offset field in the Symbol Selector dialog.



Entering the appropriate address offset will cause the logic analyzer to reference the correct symbol information for the relocatable or offset code.

Setting Up Labels for Groups of Signals

Predefined Label Descriptions

The logic analyzer configuration files automatically set up labels for most MPC8240 signals. The following tables show some of the predefined labels for the most commonly used signals.

ADDR Label

ADDR	Signal Name
Bit 0	SDMA[0]
Bit 1	SDMA[1]
Bit 2	SDMA[2]
Bit 3	SDMA[3]
Bit 4	SDMA[4]
Bit 5	SDMA[5]
Bit 6	SDMA[6]
Bit 7	SDMA[7]
Bit 8	SDMA[8]
Bit 9	SDMA[9]
Bit 10	SDMA[10]
Bit 11	SDMA[11]
Bit 12	SDMA[12] / SDBA1
Bit 13	SDBA0
Bit 14	debug_address[0]
Bit 15	debug_address[1]
Bit 16	debug_address[2]
Bit 17	debug_address[3]
Bit 18	debug_address[4]
Bit 19	debug_address[5]
Bit 20	debug_address[6]
Bit 21	debug_address[7]
Bit 22	debug_address[8]
Bit 23	debug_address[9]
Bit 24	debug_address[10]
Bit 25	debug_address[11]
Bit 26	debug_address[12]

Setting Up Labels for Groups of Signals

ADDR	Signal Name
Bit 27	debug_address[13]
Bit 28	debug_address[14]
Bit 29	debug_address[15]

DATA Label

DATA	Signal Name
Bit 0	DH[31]
...	...
Bit 31	DH[0] (MSB)

DATA_B Label

DATA_B	Signal Name
Bit 0	DL[31]
...	...
Bit 31	DL[0] (LSB)

STAT Label

STAT	Signal Name
Bit 0	MAA[2]
Bit 1	MAA[1]
Bit 2	MAA[0]
Bit 3	RCS1
Bit 4	RCS0
Bit 5	RAS / CS[7]
Bit 6	RAS / CS[6]
Bit 7	RAS / CS[5]
Bit 8	RAS / CS[4]
Bit 9	RAS / CS[3]
Bit 10	RAS / CS[2]
Bit 11	RAS / CS[1]
Bit 12	RAS / CS[0]
Bit 13	AS
Bit 14	WE
Bit 15	FOE
Bit 16	CAS / DQM[7]

STAT	Signal Name
Bit 17	CAS / $\overline{\text{DQM}}[6]$
Bit 18	CAS / $\overline{\text{DQM}}[5]$
Bit 19	CAS / $\overline{\text{DQM}}[4]$
Bit 20	CAS / $\overline{\text{DQM}}[3]$
Bit 21	CAS / $\overline{\text{DQM}}[2]$
Bit 22	CAS / $\overline{\text{DQM}}[1]$
Bit 23	CAS / $\overline{\text{DQM}}[0]$
Bit 24	SDRAM_CLK[0]*
Bit 25	$\overline{\text{MIV}}$
Bit 26	$\overline{\text{SDRAS}}$
Bit 27	$\overline{\text{SDCAS}}$

Clock & Qualifier

CLK	Signal Name
J	SDRAM_CLK[0] (SYSCLK)
K	$\overline{\text{MIV}}$

To define additional labels

- 1 Open the **Setup** window.
- 2 Click the **Format** tab.
- 3 Click a label and select **Insert before...** or **Insert after....**
- 4 Click the signals under the appropriate pod, then select which bits to include in the label.

Changing the Analysis Mode

The logic analyzer can be set up to operate in the following analysis modes:

- State.
- Timing.

Inverse assembly is available in the state analysis mode.

To change to state analysis

In state mode, the logic analyzer uses the CLKIN signal from the MPC8240 target system to capture data synchronously. This mode allows inverse assembly of MPC8240 instructions and is the default mode set up by the configuration files.

To configure the logic analyzer for state mode:

- Load the appropriate logic analyzer configuration file (see “To load configuration files (and the inverse assembler) from hard disk” on page 86).

The state configuration files set up the rising edge of the J clock ($J\uparrow$) as the master clock signal.

You can change the master clock setting by opening the logic analyzer’s Setup window, selecting the **Format** tab, and clicking the **Master Clock** button to open the master clock dialog.

To change to timing analysis

In timing mode, the logic analyzer samples the microprocessor pins asynchronously, according to an internal, adjustable sample rate clock. The minimum sample period for a 250 MHz timing analyzer is 4 ns.

Inverse assembly is not available in the timing analysis mode.

To configure the logic analyzer for timing analysis:

- 1** Load the appropriate logic analyzer configuration file (see “To load configuration files (and the inverse assembler) from hard disk” on page 86).
- 2** Open the logic analyzer’s **Setup** window.
- 3** Select the **Sampling** tab.
- 4** Change the type option from **State Mode** to **Timing Mode**.

Changing the Analysis Mode

Capturing MPC8240 Execution

Chapter 6: Capturing MPC8240 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 MPC8240 inverse assembler, the logic analyzer is configured, and labels are created (formatted) for the logic analysis channels when configuration files are loaded (see “To load configuration files (and the inverse assembler) from hard disk” on page 86).

You can load program object file symbols into the logic analyzer when configuring it (see “Loading Symbol Information” on page 109).

This chapter describes setting up logic analyzer triggers when using the inverse assembler and the Agilent Technologies B4620B Source Correlation Tool Set.

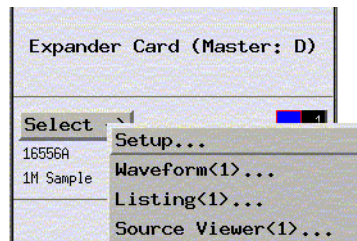
See Chapter 7, “Displaying Captured MPC8240 Execution,” beginning on page 131 for information on displaying captured data.

Note: The screens you see may be different from what you see in this manual, depending on the version of your logic analyzer system software.

Setting Up Logic Analyzer Triggers

To set up logic analyzer triggers

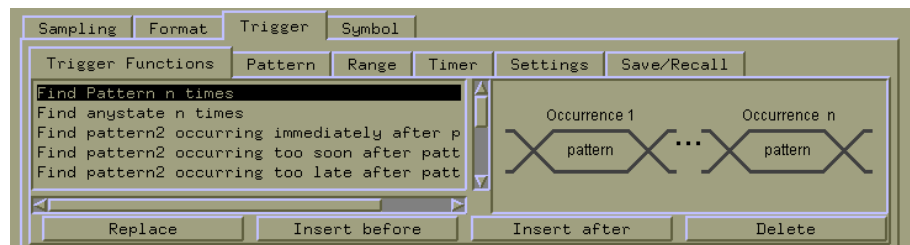
- 1 Open the logic analyzer's Setup window.



- 2 Select the Trigger tab.



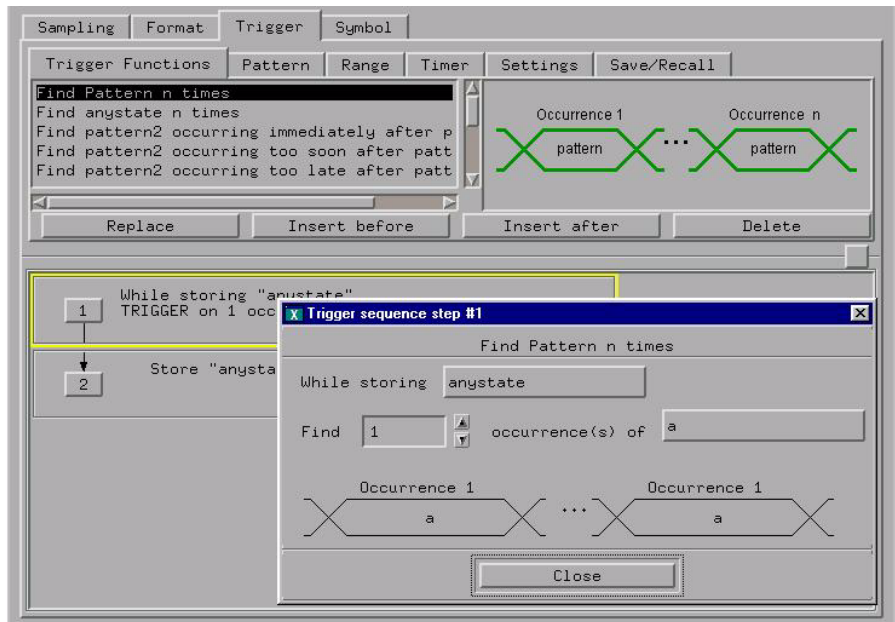
- 3 Define the patterns, ranges, and other resources that will be used in the logic analysis measurement.



Chapter 6: Capturing MPC8240 Execution

Setting Up Logic Analyzer Triggers

4 Set up the trigger sequence.



5 Run the measurement.



See Also

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

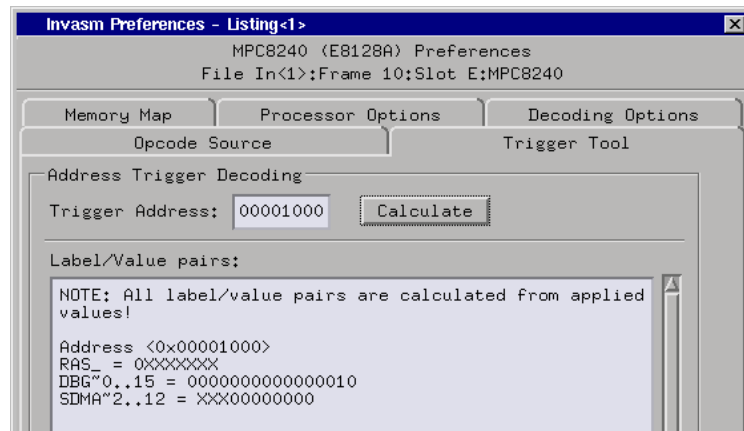
To trigger on an access to a memory address

The address bus is comprised of a group of signals.

- 1 In the Listing window, open the Invasm Preferences dialog.
- 2 Click the **Memory Map** tab and check that you have set up the memory banks correctly. Click **Apply**.

For more information on the memory map, see “To set the Memory Map preferences” on page 96.

- 3 Click the **Trigger Tool** tab.
- 4 Enter the trigger address in the Trigger Address field, then click **Calculate**.



For DRAM or SDRAM addresses:

If the Trigger Tool shows a RAS_ value, the address is in DRAM or SDRAM.

- 5 In the logic analyzer Setup window, click the **Trigger** tab, then click **Recall** and select the **SDRAM Address** trigger.

- 6 Click the **Pattern** tab and scroll down to the “SDRAM” pattern.



- 7 Enter the values from the Trigger Tool.

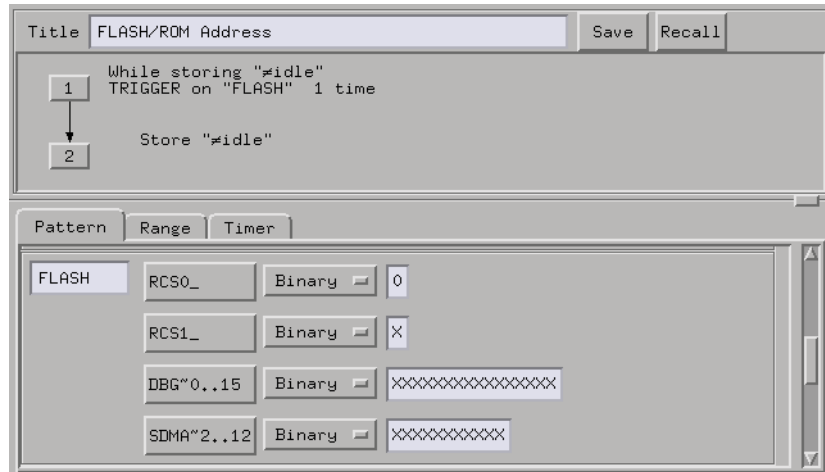
If you are working at a display which is connected directly to the logic analysis system, you can cut text by holding down the left mouse button dragging the cursor over the text. You can paste text by clicking the middle button.

For FLASH/ROM addresses:

If the Trigger Tool shows a RCS0_ or RCS1_ value, the address is in FLASH/ROM.

- 5 In the logic analyzer Setup window, click the **Trigger** tab, then click **Recall** and select the **FLASH/ROM Address** trigger.

6 Click the **Pattern** tab and scroll down to the “FLASH” pattern.



7 Enter the values from the Trigger Tool.

If the Trigger Tool does not show a value for one of the chip selects, enter ‘X’ (“don’t care”). For example, if the Trigger tool shows “RCS0_ = 0”, enter ‘0’ in the RCS0_ field, and enter ‘X’ for RCS1_.

If you are working at a display which is connected directly to the logic analysis system, you can cut text by holding down the left mouse button dragging the cursor over the text. You can paste text by clicking the middle button.

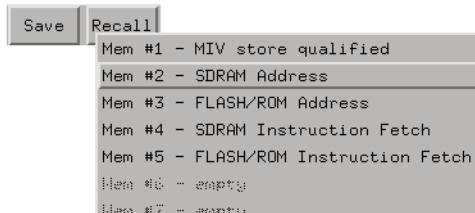
For both kinds of addresses:

8 **Run** the logic analyzer measurement.



Using the Saved Trigger Specifications

Logic analyzer configuration files for the MPC8240 contain saved trigger specifications. You can recall these trigger specifications from the Recall button of the Trigger tab.



MIV store qualified

This is the default trigger specification. Any captured state will trigger the logic analyzer and any non-idle ($\overline{\text{MIV}}=0$) state after that is stored. It gives you an easy way to return to the default.

SDRAM Address

Any access to the address defined by the “SDRAM” pattern will trigger the logic analyzer and any non-idle ($\overline{\text{MIV}}=0$) state after that is stored.

The inverse assembler Preferences dialog has a Trigger Tool tab that will show you the values to use for a particular SDRAM address (see “To trigger on an access to a memory address” on page 125).

FLASH/ROM Address

Any access to the address defined by the “FLASH” pattern will trigger the logic analyzer and any non-idle state after that is stored.

The inverse assembler Preferences dialog has a Trigger Tool tab that will show you the values to use for a particular address.

SDRAM Instruction Fetch

Any instruction fetch from address defined by the “SDRAM” pattern will trigger the logic analyzer and any non-idle state after that is stored.

The inverse assembler Preferences dialog has a Trigger Tool tab that will show you the values to use for a particular address.

FLASH/ROM Instruction Fetch

Any instruction fetch from address defined by the “FLASH” pattern will trigger the logic analyzer and any non-idle state after that is stored.

The inverse assembler Preferences dialog has a Trigger Tool tab that will show you the values to use for a particular address.

Triggering on Source Code

When setting up trigger specifications to capture MPC8240 execution:

- 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).

To avoid capturing library code execution

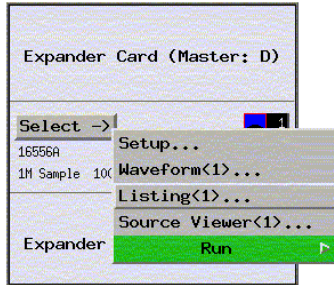
When viewing the source code associated with captured data, the source correlation tool set can exhibit long response times to requests for the next source line if the current trace listing corresponds to code from a library that is not in the source code search path. Logic analyzer storage qualification can be used to avoid capturing library code routines.

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

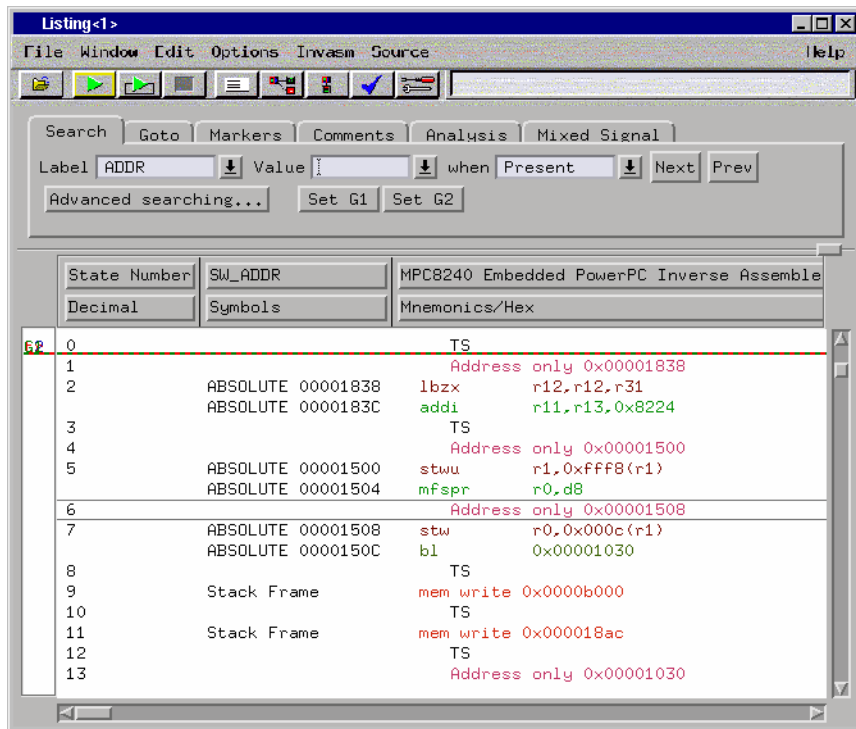
Displaying Captured MPC8240 Execution

To display the captured state data

- 1 Open the Listing display window.



The logic analyzer displays captured state data in the Listing display.



The inverse assembler is already loaded when state configuration files are loaded, but it can also be loaded into a Listing display using the Invasm menu. The name of the inverse assembler file is I8620E, and it is located in the /logic/ia directory.

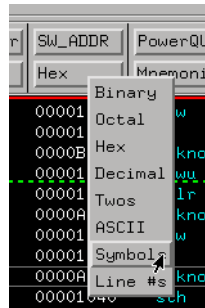
See Also

“To use the inverse assembler filters” on page 135 for information on displaying or hiding certain types of microprocessor bus cycles.

The Agilent Technologies 16600/700-series logic analysis system on-line help for information on using the Listing display.

To display symbols

- Over a Listing display’s label base, right-click the mouse button, and select Symbols.



Any symbols that have been defined will be displayed for equivalent captured values.

See Also

“To load object file symbols” on page 111.

To interpret the inverse assembled data

Registers

General purpose registers are displayed as r0, r1, ..., r31. Floating point registers are displayed as f0, f1, ..., f31. Condition registers are displayed as cr0, cr1, ..., cr7. Special purpose registers are displayed using their mnemonic.

Numbers.

Most numerical data is displayed in hexadecimal, for example, "lwz r28, 0x0044(r1)."

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

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

Instructions.

The inverse assembler decodes the full 32-bit mode PowerPC instruction set architecture. Instructions that are 64-bit mode or optional instructions not implemented on the MPC8240 are decoded as "illegal". Any instruction that does not decode to a valid opcode is shown as "unknown".

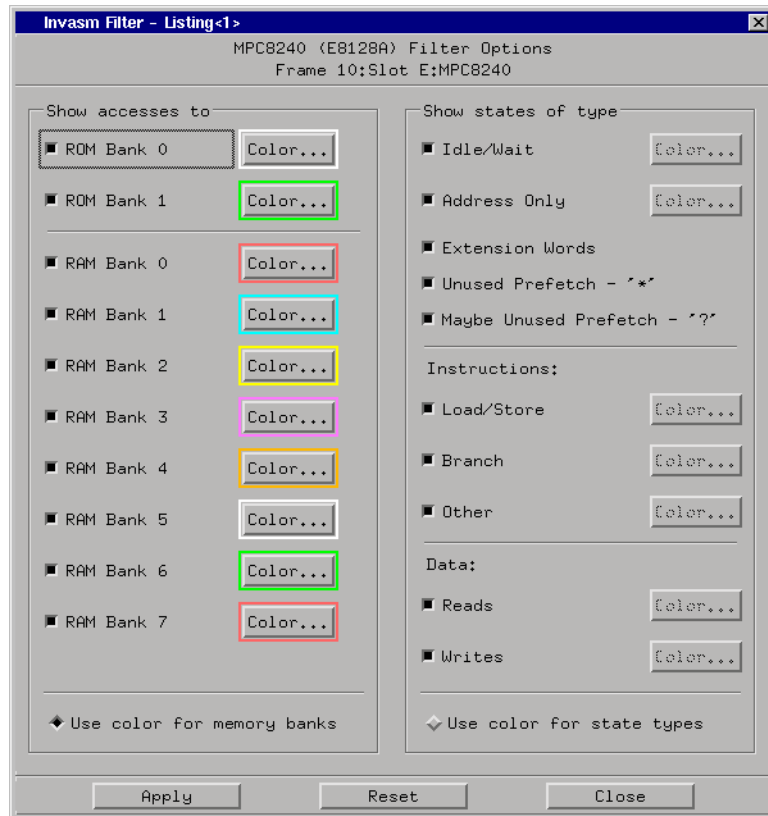
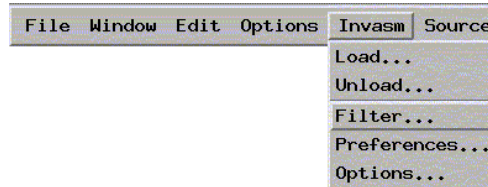
A "*" is used to mark unused prefetches. A "?" is used to mark *possibly* unused prefetches.

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

When an exception occurs, the exception type is displayed in parentheses after the instruction.

To use the inverse assembler filters

- In the Listing display window, choose the Filter command from the Invasm menu.



Chapter 7: Displaying Captured MPC8240 Execution

The inverse assembler filtering options allow you to display or hide certain types of microprocessor bus cycles or memory bank accesses.

Because the filter options do not affect the data that is stored by the logic analyzer (they only affect whether that data is displayed), they let you display the same data in different ways.

Filtering allows faster analysis in two ways:

- Unneeded information can be taken out of the display. For example, suppressing idle/wait states will let you view more instruction cycles in each screenful.
- Particular operations can be isolated by suppressing all other operations. For example, Branch instructions can be shown, with all other states suppressed, allowing quick analysis of branch instructions.

You can also use color to distinguish between cycle types or memory bank accesses (when they are displayed). Color can be used for distinguishing between memory bank accesses or cycle types, but not both at the same time.

You can display or hide the following types of cycles:

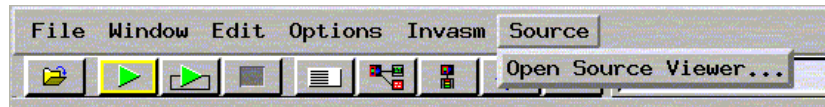
- Idle/Wait States.
- Address Only States.
- Unused or possibly unused prefetches.
- Extension Words.
- Branch Instructions.
- Load/Store Instructions.
- Other Instructions.
- Data Reads.
- Data Writes.

See Also

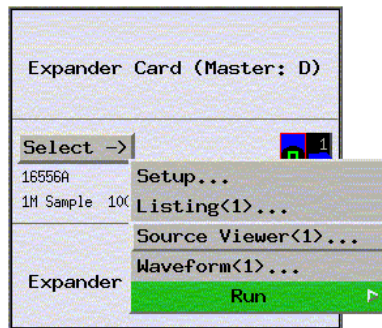
The address ranges for memory banks 0-11 are specified in the Preferences menu (see “To set the inverse assembler preferences” on page 95).

To view the source code associated with captured data

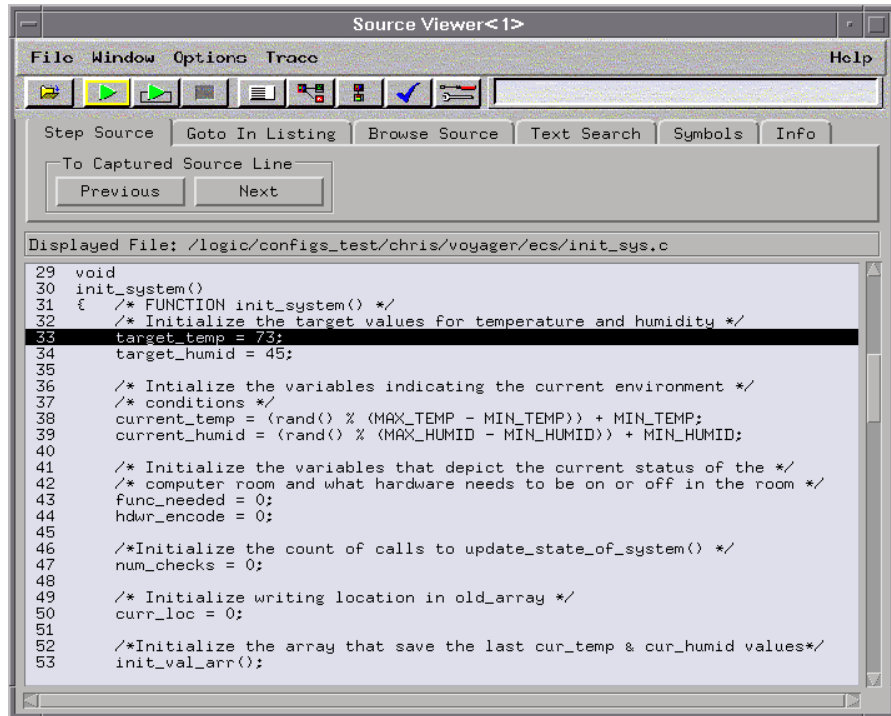
- In the Listing display window, select Source Viewer from the Source menu.



- Or, open the Source Viewer window from the logic analyzer's icon in the main system window.



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 (see “To load object file symbols” on page 111).



Inverse Assembler Generated SW_ADDR (Software Address) Label

In the Agilent Technologies 16600/700-series logic analysis system, the MPC8240 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 16600/700-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 line of source code will generate many assembly instructions. The “Goto this line in listing” commands will not find a given line of source code 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 Agilent Technologies 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, NFS client/server and X-Window client/server applications. Some PC networks may require extensions to the normal LAN protocols to support the TCP/IP protocol and/or these networking applications. Users should contact their LAN system administrators to help set up the logic analyzer on their network.

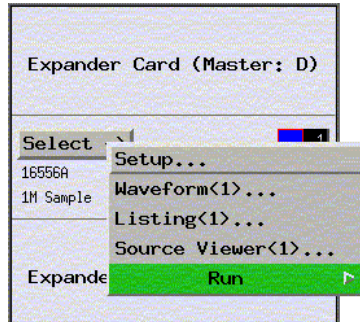
Source File Version Control. If the source code files are under a source code or version control utility, check the file names and paths carefully. These utilities can change source code file paths and file names. If these names are changed from the information contained in the symbol file, the source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

See Also

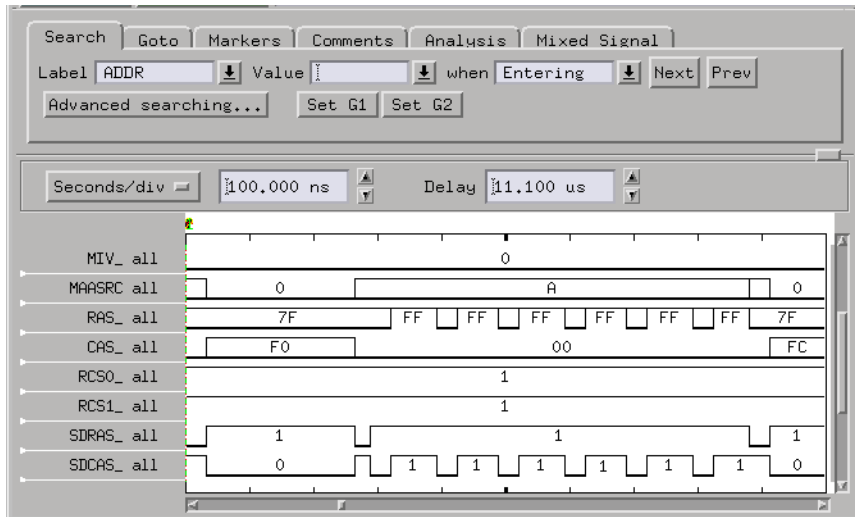
More information on configuring and using the source correlation tool set can be found in the on-line help for your logic analysis system.

To display captured timing analysis mode data

- Open the Waveform display for your logic analyzer.



You can also use the Waveform display in the state analysis mode to display state timing diagrams.



Coordinating Logic Analysis with
Processor Execution

This chapter describes how to use a logic analyzer, an emulation module, and other features of your Agilent Technologies 16600/700-series 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 logic analyzer, 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 provide different views of the data collected using the logic analyzer.
- 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. You can 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 on-line help contains quick reminders of how to perform common measurements.

A few of the many things outlined in 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.”

Stopping Processor Execution on a Logic Analyzer Trigger

You can trigger the emulation module from the logic analyzer using 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 on a source line trigger (Source Viewer window)

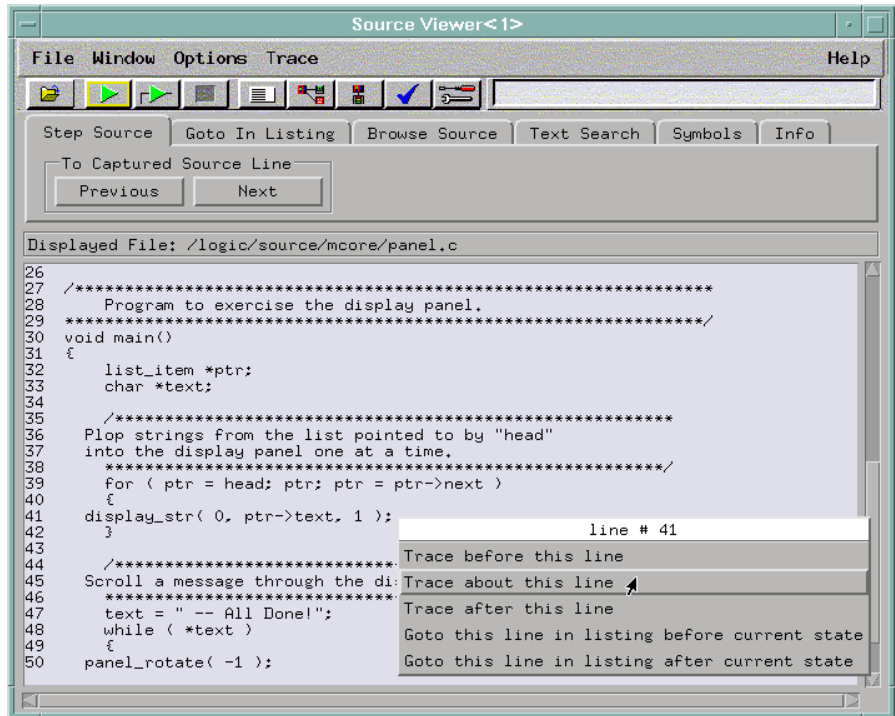
If you have the Agilent Technologies B4620B Source Correlation Tool Set, you can easily stop the processor when a particular line of code is reached.

- 1 In the Source 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.

Chapter 8: Coordinating Logic Analysis with Processor Execution

Stopping Processor Execution on a Logic Analyzer Trigger



2 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**.

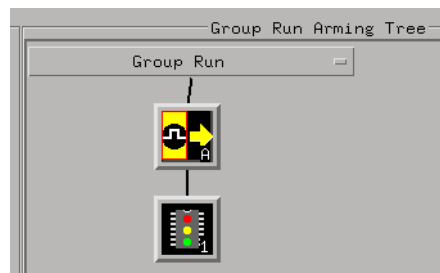
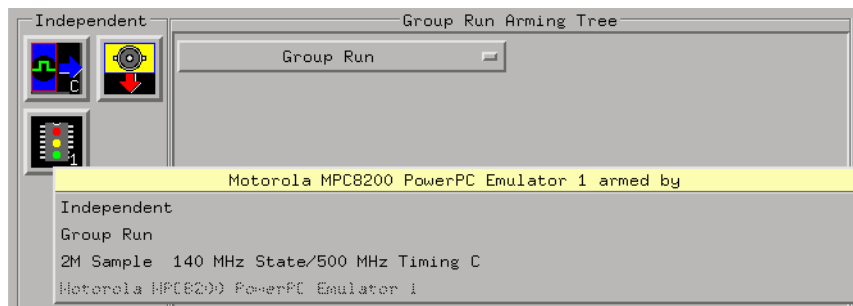
3 Select **Group Run** in the Source window (or other logic analyzer window).

4 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 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.

- 3 Select **Group Run** in the Source window (or other logic analyzer window).
- 4 If your target system is not already running, select **Run** in the emulation Run Control window to start your target.

See Also

See the on-line 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 which your target can support.

The amount of skid will depend on the processor’s execution speed and whether code is executing from the cache.

To stop the analyzer and view a measurement

- To view an analysis measurement you may have to select **Stop** after the trigger occurs.

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.

Chapter 8: Coordinating Logic Analysis with Processor Execution

Stopping Processor Execution on a Logic Analyzer Trigger

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.
If the trigger position is not “End”, the analyzer may continue waiting for more states.
6. 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. This kind of measurement is easier than setting up an intermodule measurement trigger.

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. (You can set a breakpoint at the start of the function then use this measurement to see how the function is getting called.)

To capture a trace before the processor halts

- 1 Set the logic analyzer to trigger on **nostate**.
- 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 emulation module halts, select **Stop** in the logic analyzer window to complete the measurement.

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 when Processor Execution Stops

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, it will be much simpler to trace until a processor halts (see “Tracing Until the Processor Halts” on page 149).

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 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.

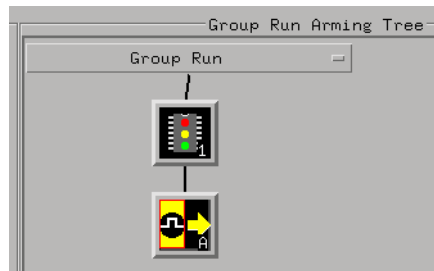
Here are some examples:

- If “Group Run” is armed from “Port In” and an emulation module is connected to Group Run, any “Port In” signal will cause the emulation module to go into monitor. The Group Run button does not have to be pressed for this to operate.
- If two emulation modules are connected together so that one triggers another, the first one going into monitor will cause the second one to go into monitor.
- If an emulation module is connected to Port Out, 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

Chapter 8: Coordinating Logic Analysis with Processor Execution

Triggering the Logic Analyzer when Processor Execution Stops

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 which appear in the listing.

You can often distinguish these additional states because the time tags will be in the microsecond and millisecond 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 microsecond and millisecond 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 your 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

Remember: if you are only using a state analyzer to capture the processor bus then it will be much simpler to trace until a processor halts (see “Tracing Until the Processor Halts” on page 149).

- 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 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.

NOTE:

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 stopped.

The logic analyzer will store states up until the processor stops, but may continue running.

You may or may not see a “slow clock” error message. In fact, if you are using a state analyzer on the processor bus, the status may never change upon receiving the emulation module trigger (analysis arm). This occurs because the qualified processor clock needed to switch the state analyzer to the next state is stopped. For example, the state analyzer before the arm event may have a status of “Occurrences Remaining in Level 1: 1” and after the arm event it may have the same status of “Occurrences Remaining in Level 1: 1”.

- 7 If necessary, in the logic analyzer window, 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 logic analyzer, select **Stop** if needed to complete the measurement.

To trigger the analyzer when the processor reaches a breakpoint

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

Remember: if you are only using a state analyzer to capture the processor bus, it will be much simpler to trace until a processor halts (see “Tracing Until the Processor Halts” on page 149).

- 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 which 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.

NOTE:

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 up until the processor stops, but may continue running.

You may or may not see a “slow clock” error message. In fact, if you are using a state analyzer on the processor bus, the status may never change upon receiving the emulation module trigger (analysis arm). This occurs because the qualified processor clock needed to switch the state analyzer to the next state is stopped. For example, the state analyzer before the arm event may have a status of “Occurrences Remaining in Level 1: 1” and after the arm event it may have the same status of “Occurrences Remaining in Level 1: 1”.

- 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 logic analyzer, select **Stop** if needed to complete the measurement.

Chapter 8: Coordinating Logic Analysis with Processor Execution
Triggering the Logic Analyzer when Processor Execution Stops

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 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.
```

Specifications and Characteristics

Operating Characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the E8128A MPC8240 analysis probe.

Operating Characteristics	
Microprocessor Compatibility	Motorola MPC8240
Package Supported	BGA
Agilent Technologies Logic Analyzers Supported	16550A 16554/55/56/57 16600/1/2/3A 16710/11/12A 16715/16/17/18/19A 16750/51/52A
Accessories Required	For state and timing analysis, the E8161A BGA probing kit and the E5346A high-density cables are required.
Optional Accessories	The 16610A emulation module or the E3453A emulation probe can be connected to the analysis probe.
Probes Required	At least four 16-channel probes are required for disassembly.

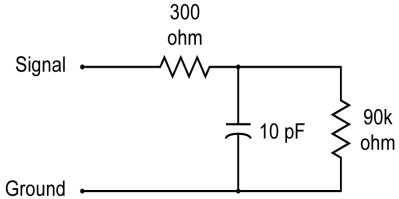


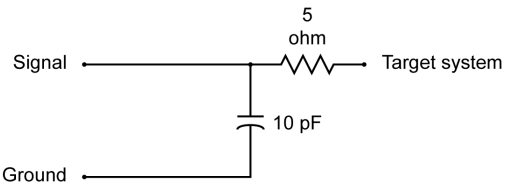
Electrical Characteristics

Power Requirements	300mA @ 5.0V, supplied by the logic analyzer, CAT 1, Pollution degree 2. Approximatley 0.1 μ F decoupling on GVdd, LVdd, 2.5V (Vdd), and 3.3V (OVdd) supplies. Maximum draw of 4mA from target system 3.3V (OVdd).
Signal Line Loading	(See the processor signal line loading table that follows.)

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% noncondensing. 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.

Processor Signal Line Loading

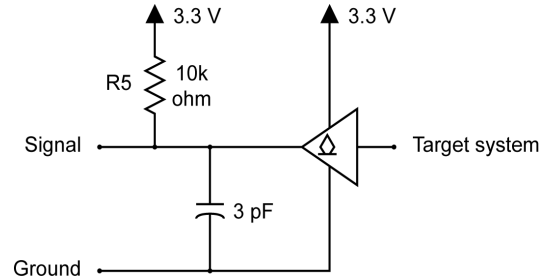
Signal	MPC 8240 Pin	Analysis Probe Switch	Equivalent Load (includes logic analyzer)
Any probed signal		N/A	 <p>(E5346A)</p>
SDRAM_CLK[0]	D1	N/A	
TDO*	AC21	S1 = OFF	
		S1=ON	

*Signal is on the JTAG port. The equivalent load does not include the emulation module /probe.

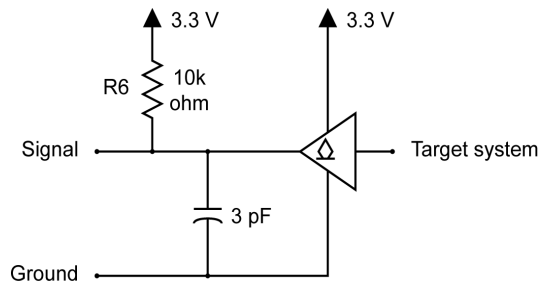
Signal	MPC Analysis Pin	8240 Probe DIP Switch	Equivalent Load (includes logic analyzer)
TDI*	AF23	S1=OFF	
		S1=ON	
TCK*	AF22	S1 = OFF	
TMS*	AE22		
TRST*	AE23		
		S1 = ON	

Signal	MPC 8240 Pin	Analysis Probe Switch	Equivalent Load (includes logic analyzer)
--------	--------------	-----------------------	---

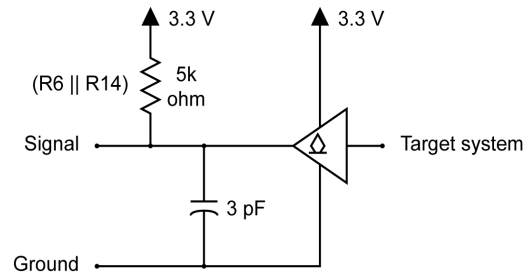
$\overline{\text{SRESET}}^*$ B16 N/A



$\overline{\text{HRST_CPU}}^*$ A19 S2 = OFF



S2 = ON



Signal	MPC Analysis	Equivalent Load
	8240 Probe DIP	(includes logic analyzer)
Pin	Switch	
HRST_CTRL*	A20 S2 = OFF	
	S2 = ON	
CHKSTOP_IN	D14 N/A	
All other Signals	N/A	

Troubleshooting the Logic Analyzer

Chapter 11: 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, and probes. Otherwise, you may damage circuitry in the logic analyzer or target system.

Solving Logic Analyzer Problems

This section lists general problems that you might encounter while using the analyzer.

Intermittent data errors

This problem is usually caused by poor connections, incorrect signal levels, or marginal timing.

- ❑ Remove and reseal all cables and probes, ensuring that there are no bent pins or poor 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 180 for information on other sources of intermittent data errors.

Unwanted triggers

Unwanted triggers can be caused by instructions that were fetched but not executed.

- ❑ Set the trigger at an address which is not the target of a conditional branch to avoid this problem. For example, set the trigger in the middle of a subroutine, rather than at the first instruction of a subroutine.

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 or board connections.
- Check for bent or damaged pins.

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 sequencer specification 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.

Solving Probing Problems

This section lists probing problems that you might encounter when using a logic analyzer. If the solutions suggested here do not correct the problem, you may have a damaged logic analyzer. 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 target system, the microprocessor (if socketed) or the probe cables 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 logic analyzer and target system.
 1. Power up the logic analyzer.
 2. Power up the target system.
- ❑ If you power up the target system before you power up the logic analyzer, interface circuitry may latch up and prevent proper target system operation.
- ❑ Verify that the logic analyzer cables are in the proper target system connector headers and are firmly inserted.

Erratic trace measurements

There are several general problems that can cause erratic variations in trace lists and inverse assembly failures.

- ❑ Do a full reset of the target system before beginning the measurement.

Some designs require a full reset to ensure correct configuration.

- ❑ Ensure that your target system meets the timing requirements of the processor with the logic analyzer connected.

See “Capacitive loading” on page 180. While logic analyzer 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 airflow that meet or exceed the requirements of the microprocessor manufacturer.

Capacitive loading

Excessive capacitive loading can degrade signals, resulting in incorrect capture by the logic analyzer, or system lockup in the microprocessor. All logic analyzer probes 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.

Solving 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 logic analyzer or in your target system. If you follow the suggestions in this section to ensure that you are using the logic analyzer 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 connector.

There is not always a one-to-one correspondence between analyzer pod numbers and probe cable numbers. Probes must supply address (ADDR), data (DATA), and status (STAT) information to the analyzer in a predefined order. The cable connections for each probe are often altered to support that need. Thus, one 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 “Connecting the Logic Analyzer to the Target System” on page 62 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.

Chapter 11: Troubleshooting the Logic Analyzer

Solving Inverse Assembler Problems

Some inverse assemblers also require other data labels. See “Configuring the Logic Analyzer” on page 83 for more information.

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

In most cases, if the microprocessor caches and memory managers remain enabled you should still get inverse assembly; however, it will consist of cache line fills.

- ❑ Verify that storage qualification has not excluded storage of all the needed opcodes and operands.

Inverse assembler will not load or run

You need to ensure that you have the correct system software loaded on your analyzer.

- ❑ Ensure that the inverse assembler is on the same disk as the configuration files you are loading.

Configuration files for the state analyzer contain a pointer to the name of the corresponding inverse assembler. If you delete the inverse assembler or rename it, the configuration process will fail to load the disassembler.

See “Configuring the Logic Analyzer” on page 83 for details.

Solving 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 the oscilloscope 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 the oscilloscope, 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.

Logic Analyzer 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.

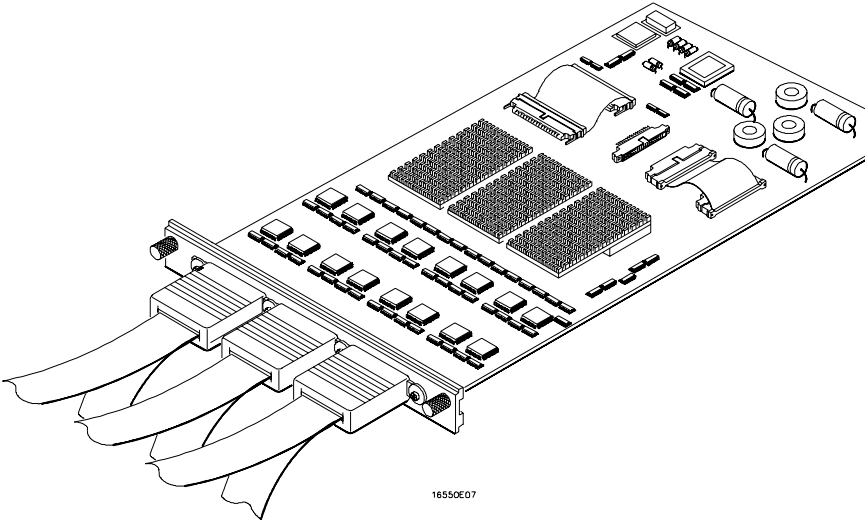
Ensure that the inverse assembler file is not renamed or deleted, and that it is located in the correct directory:

- For Agilent 16600/700-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.
-

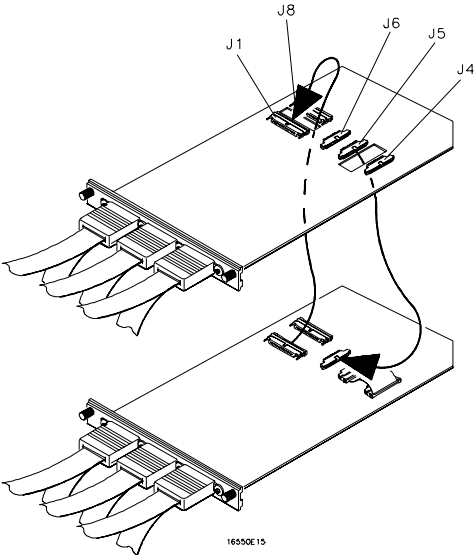
“Measurement Initialization Error”

This error occurs when you have installed the cables incorrectly for one or two Agilent Technologies 16550A logic analyzer cards. The following diagrams show the correct cable connections for one-card and two-card installations. Ensure that your cable connections match the silk-screened labels on the card, and that they are fully seated in the connectors. Then, repeat the measurement.

Cable Connections for One-Card 16550A Installations



Cable Connections for Two-Card 16550A Installations



See Also

The Agilent Technologies 16550A 100-MHz State/500-MHz Timing Logic Analyzer Service Guide.

“No Configuration File Loaded”

This is usually caused by trying to load a configuration file for one type of module/system into a different type of module/system.

- ❑ Verify that the appropriate module has been selected from the Load {module} from File {filename} in the disk operation menu. Selecting Load {All} will cause incorrect operation when loading most configuration files.

See Also

See “Configuring the Logic Analyzer” on page 83 for a description of how to load configuration files.

“Selected File is Incompatible”

This occurs when you try to load a configuration file for the wrong module. Ensure that you are loading the appropriate configuration file for your logic analyzer.

“Slow or Missing Clock”

- ❑ This error message might occur if the logic analyzer cards are not firmly seated in the Agilent Technologies 16600/700-series logic analysis system frame or in the Agilent Technologies 16701A expansion 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 target system. See “Connecting the Logic Analyzer to the Target System” on

page 62 to determine the proper connections.

“Time from Arm Greater Than 41.93 ms”

The 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, if the trigger condition is set to look for an opcode fetch at an address not corresponding to a word boundary, the trigger will never be found.

Chapter 11: Troubleshooting the Logic Analyzer
Logic Analyzer Messages

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

Glossary

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 impedance 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

Glossary

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.

Symbols

- * (unused prefetch), 134
- ? (maybe unused prefetch), 134

Numerics

- 32-bit mode, 134
- 64-bit mode, 134

A

- access to source code files, 139
- accessories required, analysis probe, 168
- acquire data, 142
- active low signal, 45
- activity indicators, 178, 181
- adapter, 43
- adapters, 180
- ADDR label, 111, 181
- address offset field, 114
- address only states, 136
- address range
 - memory banks 0-11, 136
- addresses
 - mask, 94
- Agilent Technologies 1252-7431, 43
- Agilent Technologies 16550A logic analyzer, 65, 67, 81, 88, 184
- Agilent Technologies 16554/55/56/57 logic analyzers, 68, 69
- Agilent Technologies 16554A logic analyzer, 88
- Agilent Technologies 16555A/D logic analyzer, 88
- Agilent Technologies 16556A/D logic analyzer, 88
- Agilent Technologies 16557D logic analyzer, 88
- Agilent Technologies 16600A logic analyzer, 70, 71, 88
- Agilent Technologies 16601A logic analyzer, 72, 73, 74, 88

- Agilent Technologies 16610A
 - emulation module, 168
- Agilent Technologies 16710A logic analyzer, 88
- Agilent Technologies 16715/16/17A logic analyzers, 78, 79, 81, 88
- Agilent Technologies B4620B
 - source correlation tool set, 122, 142, 143, 144, 146
- Agilent Technologies E3453A
 - emulation probe, 168
- Agilent Technologies E5346-44701 shroud, 43
- Agilent Technologies E5346-60002 adapter, 43
- Agilent Technologies E5346A high-density termination adapter cable, 43
- Agilent Technologies E5346A high-density termination cables, 64
- Agilent Technologies E8160A BGA probing kit, 37
- Agilent Technologies E9611A
 - Option 001 inverse assembler, 21
- analysis arm, 153
- analysis mode
 - changing, 118
 - state, 118, 140
 - timing, 118
- analysis probe, 2, 28, 35, 40, 41, 64
 - accessories required, 168
 - definition, 189
 - DIP switches, 54
 - electrical characteristics, 169
 - environmental characteristics, 169
 - equipment supplied, 20
 - humidity, 169
 - inverse assembly, 89
 - logic analyzers supported, 168
 - microprocessor compatibility, 168

- analysis probe (continued)
 - operating altitude, 169
 - operating temperature, 169
 - optional accessories, 168
 - package supported, 168
 - power requirements, 169
 - probes required, 168
 - signal line loading, 169
- anystate trigger, 128
- arm, 153, 155, 183, 187
- arm event, 153, 155
- arm signal, 152
- ASCII format (GPA), 158
- assembly language mnemonics, 85
- assembly-level listing, 142
- asynchronous sampling, 119

B

- B4620B source correlation tool set, 2, 3, 4, 58
- background debug monitor, 189
- ball nest socket, 38
- bank enable/disable, 98
- base address, 99
- BGA carrier, 36, 38
- BGA carrier assembly, 41
- BGA pad array, 37
- BGA probing kit, 4, 37
- BGA socket, 40
- bit numbers, 134
- blank pins, 45
- board space, 43
- branch instructions, 136
- break emulator on trigger, 145
- break into monitor, 152
- break temporarily, 152
- breakpoint, 149, 154
 - triggering analyzer on, 154
- breakpoint registers, 154
- breakpoints, 150
 - software, 154
- bus activity, processor, 149

C

cache
 disabling, 107
cache memory, 30
cache-on trace reconstruction, 93
caches, 182
 enabling and disabling, 107
capacitive loading, 180
captured data, source code
 associated with, 137
captured execution, displaying,
 131
capturing execution, 121
cards
 See logic analyzers
CD-ROM, 21, 23
CD-ROM, installing software from,
 59
center inline pins, 45
characteristics, 167
Chart tool, 142
clearance, mechanical, 32
CLKIN signal, 118
clock signals, 177
clock speed
 processor, 147
color, 136
comments, in GPA files, 166
condition registers, 134
conductive foam wafer, 36
conductive plastic pin protector,
 36
configuration files, 21, 58, 88, 109,
 118, 119, 122, 133, 182, 184,
 186
Configuration window, 147
configuring the logic analyzer, 85
connection notes
 recommended configuration, 45
connector board, 189
connector layout, 45
 recommended, 46

connectors for logic analyzer probe
 pods, 29
control execution, 142
cooling, 180
coordinating logic analysis, 141
counter overflow, 187
crash, system, 149
creating GPA symbol files, 158
custom probe fixtures, 180
cycle types, 136

D

data cache, 30
DATA label, 181
data read/write, 105
data reads, 136
data values, inverse assembly, 105
data writes, 136
debug mode, 189
 enable, 96
debug port, 189
 definition, 189
debugger, 142, 143, 149, 152, 153,
 154
decoding
 exception, 104
 simplified mnemonic, 104
decoding options, 103
deep memory logic analyzer, 138
default trigger specification, 128
design considerations, 29
DIP switches, analysis probe, 54
directories
 configuration files, 86
display timing analysis mode data,
 140
Display tools, 142
displaying captured execution, 131
double header, removing, 40
download symbol information, 109

E

E8125A analysis probe and inverse
 assembler, 4
E8126A inverse assembler, 4
E8160-60001 BGA probing kit, 4,
 37
E9503A emulation solution, 2
E9503A Option 001 emulation
 solution, 20
E9503A Option 002 emulation
 solution, 4
E9603A Option 001 inverse
 assembler, 4
E9603A Option 002 analysis probe
 and inverse assembler, 4
elastomeric probe adapter
 definition, 189
electrical characteristics
 analysis probe, 169
Emulation Control Interface, 58,
 142, 147, 149, 150, 153, 154
emulation migration
 definition, 189
emulation module, 2, 3, 142, 149,
 153
 connected to Port Out, 151
 definition, 189
 firmware, 58
 icon, 146
 Port In connected to, 151
 stop processor on trigger, 146
 target system design, 54
 trigger, 144
 trigger signal, 150
 triggering logic analyzers, 150
emulation modules
 connected in series, 151
emulation probe
 definition, 189
emulation solution, 2
enable debug mode, 96
environmental characteristics
 analysis probe, 169

equipment, 15
 protecting, 36
equipment and software
 supplied, 17
erratic trace measurements, 180
Ethernet networks, 139
event wasn't captured, 183
examples of measurements, 143
exception decoding, 104
extender, 32, 40, 41, 189
extenders, 180
extension words, 136
extractor tool, 40

F

false trigger, 152
filter library code execution, 130
filters, inverse assembler, 135
firmware
 emulation module, 58
fixed code offsets, 114
FLASH/ROM triggers, 128
flexible adapter
 definition, 190
floating-point registers, 134
foam wafer, 36
frequencies greater than 50 MHz,
 180
ftp, 139
function calls, 149
FUNCTIONS in GPA format, 163

G

gateway address
 definition, 190
General-Purpose ASCII (GPA)
 symbol file, 111
General-Purpose ASCII format, 158
 address format, 158
 comments, 166
 FUNCTIONS, 163
 record format summary, 160

General-Purpose ASCII format
(continued)
 record headers, 158
 SECTIONS, 162
 simple form, 158
 SOURCE LINES, 165
 START ADDRESS, 166
 VARIABLES, 164
general-purpose flexible adapter
 definition, 190
general-purpose registers, 134
glitches, 143
ground returns, 45
Group Run, 151, 153, 154

H

halt the processor, 145
hardware breakpoint registers, 154
height of components, 32
high-density adapter cable
 definition, 190
high-density connector
 mechanical specifications, 44
 pin assignment, 44
high-density connectors, 29, 43
high-density termination adapter
 definition, 190
high-density termination cables, 64
high-level source code, 137, 142
holes, mounting, 34
HRESET signal, 54
humidity, analysis probe, 169

I

idle states, 128
idle/wait states, 136
illegal instruction, 134, 154
In Monitor, 151, 152
In Monitor signal, 150
incorrect inverse assembly, 181
incorrect signal levels, 177
information sources, 6

inline pins, 45
input voltage, maximum, 30
installation, overview of, 17
installing logic analyzer modules,
 57
installing software, 58
instruction fetches, triggering on,
 128
instruction word, 134
intermittent data errors, 177
intermodule measurement, 147,
 150
 problems, 183
 trigger, 149
Intermodule window, 144, 146,
 151, 153, 154
internal analyzer delays, 183
internal data cache, 30
internal instruction cache, 30
internal sample rate clock, 119
invalid opcode, 134
Invasm menu, 133
inverse assemble
 configuration file names, 62
inverse assembled data, 134
inverse assembler, 2, 29, 58, 85,
 133, 138, 182
 definition, 190
 equipment supplied, 21
 file name, 184
 filters, 135
 loading files, 86, 87
 not found, 184
 preferences, 95, 128, 129
 problems, 181
 requirements for, 89
 software, 21
 will not load, 182
inverse assembly, 89, 118, 182
 cache-on, 93
 incorrect, 181
 traditional, 89, 93

-
- IP address
 - definition, 190
 - J**
 - J clock, 118
 - JTAG control, 54
 - jumper, definition, 190
 - K**
 - keep-out area, 32
 - L**
 - labels
 - defining, 117
 - definition, 190
 - predefined, 115
 - LAN protocols, 139
 - LAN system administrators, 139
 - latency, 147
 - layout, 180
 - library code execution, 130
 - link-level address
 - definition, 191
 - Listing display window, 132
 - Listing tool, 142
 - Load menu, 94
 - load/store instructions, 136
 - loading configurations, vs.
 - installing, 86
 - loading object file symbols, 111
 - loading symbol information, 109
 - loading, minimum, 30
 - loading, processor signal line, 170
 - locked status line, 181
 - logic analysis
 - coordinating, 141
 - preparing target for, 29
 - logic analysis system
 - setting up, 55
 - software version, 23
 - logic analyzer, 176
 - configuration files, 88, 109
 - configuring, 83
 - connecting to target system, 62, 80
 - deep memory, 138
 - maximum input voltage, 30
 - messages, 184
 - modules, installing, 57
 - Pods, 181, 186
 - solving problems, 177
 - stopping, 147
 - storage qualification, 130
 - trigger setup, 123
 - triggers, 122
 - troubleshooting, 175
 - logic analyzer connectors, 28
 - logic analyzers
 - configuring, 86, 87
 - supported, 23, 168
 - M**
 - mainframe logic analyzer
 - definition, 191
 - male-to-male header
 - definition, 191
 - marginal timing, 177
 - master clock dialog, 118
 - master clock signal, 118
 - maximum input voltage, 30
 - measurement examples, 143
 - measurement initialization error, 184
 - measurement, viewing, 147
 - mechanical specifications
 - high-density connector, 44
 - memory, 142
 - memory bank, 101, 136
 - accesses, 136
 - memory banks, 94
 - Memory Disassembly window, 142
 - memory management units, 114
 - memory managers, 182
 - memory map, 96
 - memory map information, 96
 - messages
 - logic analyzer, 184
 - microprocessor bus cycles, 136
 - microprocessor caches, 182
 - microprocessor compatibility
 - analysis probe, 168
 - Mictor (Matched Impedance Connector) connectors, 43
 - MICTOR connector, definition, 191
 - minimum loading, 30
 - mnemonics, assembly language, 85
 - monitor, background debug, 189
 - mounting holes, 34
 - N**
 - network access to source files, 139
 - NFS client/server, 139
 - no configuration file loaded, 186
 - no inverse assembly, 181
 - no-connect, 45
 - no-data inverse assembly, 105
 - nostate, trigger on, 149
 - NULL pointer de-references, 143
 - numerical data, 134
 - O**
 - object file formats, 109
 - object file symbols, 122
 - loading, 111
 - occurrences remaining in level 1, 153, 155
 - on-chip hardware breakpoint
 - registers, 154
 - one-card 16550A installations, 185
 - opcode fetch, 187
 - opcode source, 105
 - operands, 134
 - operating altitude, analysis probe, 169

- operating temperature, analysis
 - probe, 169
 - optional accessories, analysis
 - probe, 168
 - optional instructions, 134
 - Options menu, 94
 - oscilloscope, 153, 155, 177, 183
 - measurement, 149
 - modules, installing, 57
 - other instructions, 136
 - other options, 102, 103
 - overview of installation, 17
- P**
- package supported, analysis probe, 168
 - pattern generator modules, installing, 57
 - patterns, 123
 - PCI analysis, 23, 29
 - PCI bus, 45
 - signal levels, 31
 - performance, profile system, 143
 - personality files, 58
 - pin protector, 36
 - pin protectors, 180
 - plastic pin protector, 36
 - plastic shroud, 43
 - Pods, logic analyzer, 181, 186, 191
 - poor connections, 177
 - Port In, 151
 - Port Out, 151
 - power requirements, analysis
 - probe, 169
 - power-on sequence, 179
 - power-ON/OFF sequence, 56
 - predefined symbols, 109, 110
 - preferences, inverse assembler, 95, 128, 129
 - prefetch, unused, 134
 - prefetches, 177
 - preprocessor
 - See* analysis probe
 - preprocessor interface
 - See* *analysis probe*
 - probe fixtures, custom, 180
 - probed signal lines, 30
 - probes required, analysis probe, 168
 - probing the target system, 61
 - problems
 - intermodule measurement, 183
 - inverse assembler, 181
 - logic analyzer, 177
 - probing, 179
 - processor activity, 149
 - processor bus, 155
 - activity, 149
 - state analysis of, 149
 - processor clock speed, 147
 - processor execution
 - coordinating, 141
 - stopping on analyzer trigger, 144
 - stops trigger the analyzer, 150
 - processor halt, 153, 154, 155
 - trace before, 149
 - triggering on, 152
 - processor options, 102
 - processor signal line loading, 170
 - processor support package, 58, 59
 - profile system performance, 143
 - program counter, 142, 143, 152
 - program stack, 152
 - protect your equipment, 36
 - pull-up resistors, 30
 - pulse shape requirements, 177
- Q**
- qualified processor clock, 153, 155
- R**
- ranges, 123
 - Recall button, 128
 - recommended circuit board routing, 44
 - recommended configuration
 - connection notes, 45
 - recommended connector layout, 45, 46
 - recommended signal routing, 45, 47
 - record format, General-Purpose ASCII, 160
 - record headers, 158
 - references, 6
 - registers, 142
 - relocated code, compensating for, 114
 - removing the analysis probe, 40
 - requirements, 15
 - electrical, 31
 - mechanical, 32
 - target system, 54
 - resources, 123
 - run control tool
 - See* emulation control interface
 - Run Control window, 145, 147, 153, 154
 - Running, emulation module state, 151
- S**
- sample period, 119
 - sample rate clock, internal, 119
 - saved trigger specifications, 128
 - SDMA signals, naming, 47
 - SDRAM address trigger, 128
 - search path, source code, 130, 139
 - sect1 title, 89
 - Section Format, 158
 - SECTIONS in GPA format, 162
 - selected file is incompatible, 186
 - setting breakpoints, 154
 - setup and hold time requirements, 177
-

- Setup Assistant, 18, 56, 58
 - setup assistant
 - definition, 191
 - Setup window, 109
 - setup, overview of, 17
 - shift counts, 134
 - shroud, 43, 44
 - signal ground returns, 45
 - signal integrity, 43, 46, 177
 - signal line loading, 170
 - signal line loading, analysis probe, 169
 - signal routing
 - recommended, 47
 - simplified mnemonic decoding, 103
 - simplified mnemonics, 103
 - skew, 183
 - skid effect, 147
 - slow clock error message, 153, 155
 - slow or missing clock, 186
 - socket
 - BGA, 32
 - socket, removing, 40
 - software
 - list of installed, 88
 - software addresses, 85, 138
 - software breakpoints, 154
 - software supplied, 20
 - software version, 23
 - software, installing, 58
 - solution
 - definition, 191
 - source code, 137, 142
 - associated with captured data, 137
 - search path, 130
 - triggering on, 130
 - source correlation tool set, 3, 137, 139
 - source files
 - network access to, 139
 - search path, 139
 - version control, 139
 - source line trigger, stopping
 - processor execution, 144
 - SOURCE LINES in GPA format, 165
 - Source Viewer window, 143, 144, 146
 - Source window, 144
 - source-level listing, 143
 - special-purpose registers, 134
 - specifications, 167
 - speed, processor clock, 147
 - S-Record files, 105
 - SRESET signal, 54
 - stack, program, 152
 - stand-alone logic analyzer
 - definition, 192
 - START ADDRESS in GPA format, 166
 - STAT label, 181
 - state analysis, 149, 192
 - definition, 192
 - state analyzer, 155
 - state mode, 118, 140
 - changing to, 118
 - state of the processor, 142
 - status display, 153
 - status lines, locked, 181
 - stop on any trigger, 146
 - stop the analyzer, 147
 - storage qualification, 130, 178, 182
 - subnet mask
 - definition, 192
 - supplied equipment and software, 20
 - support shroud, 44
 - supported logic analyzers, 23
 - suppressing all other operations, 136
 - surface mount connector, 44
 - SW_ADDR label, 85, 138
 - symbol file, 137
 - symbol files
 - creating, 158
 - symbol information
 - loading, 109
 - Symbol Selector dialog, 113
 - symbols
 - definition, 192
 - predefined, 110
 - Symbols tab, 109
 - symbols, displaying, 133
 - synchronization, 181
 - system administrators, 139
 - system crash, 149
 - System Performance Analyzer tool, 142
 - system performance, profile, 143
- T**
- target board adapter
 - definition, 192
 - target control port, 192
 - target interface module (TIM)
 - definition, 192
 - target system, 142, 176, 181
 - connecting logic analyzer to, 62
 - preparing, 27
 - probing, 61
 - requirements for emulation, 54
 - won't boot up, 179
 - TCP/IP protocol, 139
 - telnet, 139
 - temporary breaks, 152
 - threshold level, 177
 - time from arm greater than 41.93 ms, 187
 - timing analysis, 149, 192
 - definition, 192
 - timing analysis mode, 118
 - changing to, 119
 - data, displaying, 140
 - timing analyzer, 153, 155
 - timing requirements, 177, 180
 - tools, 142
 - trace about this line, 144

trace until processor halt, 150
transition board
 definition, 193
trigger condition, 178
trigger on anystate, 154
trigger on nostate, 149
trigger pattern, 187
trigger point, 149, 152, 154
trigger position, 149
trigger sequence, 124
trigger sequencer specification,
 178
trigger specification
 definition, 193
trigger specifications, saved, 128
trigger tool, 109, 128, 129
triggering on FLASH/ROM
 addresses, 128
triggering on instruction fetches,
 128
triggering on SDRAM addresses, 95
triggering on source code, 130
troubleshooting, 175
two-card 16550A installations, 185

U

undefined opcode, 134
unnneeded information, 136
unused prefetch, 178
unwanted triggers, 177
user defined signals, 45
User Defined Symbols tab, 109
user-defined symbols, 109

V

valid opcode, 134
VARIABLES in GPA format, 164
version control, source file, 139
version, software, 23
view a measurement, 147

W

waiting for trigger, 187
Waveform display, 140
web sites
 logic analyzers, 6
 See Also under debugger names
word-aligned addresses, 187

X

X-Window client/server, 139

Y

Y-cable, 43

© Copyright Agilent Technologies
1994-2000
All Rights Reserved.

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

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in subparagraph (C) (1) (ii) of the Rights in Technical Data and Computer Software Clause in DFARS 252.227-7013. 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 or fire hazard.

- Service instructions are for trained service personnel. To avoid dangerous electric shock, do not perform any service unless qualified to do so. Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

- If you energize this instrument by an auto transformer (for voltage reduction), make sure the common terminal is connected to the earth terminal of the power source.

- Whenever it is likely that the ground protection is impaired, you must make the instrument inoperative and secure it against any unintended operation.

- Do not operate the instrument in the presence of flammable gasses or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.

- Do not install substitute parts or perform any unauthorized modification to the instrument.

- Capacitors inside the instrument may retain a charge even if the instrument is disconnected from its source of supply.

Safety Symbols



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



Hazardous voltage symbol.



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

WARNING

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

CAUTION

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

Product Warranty

This 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 Company will, at its option, either repair or replace products that prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by 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 *Solutions for the Motorola MPC8240 User's Guide*.

Publication number
E8128-97005, August 2000
Printed in USA.

Print history is as follows:
E8128-97000, March 1999
E8128-97001, July 1999
E8128-97002, October 1999
E8128-97003, December 1999
E8128-97004, May 2000

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.

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

UNIX is a registered trademark of The Open Group.

Windows, MS Windows, Windows NT, and MS-DOS are U.S. registered trademarks of Microsoft Corporation.

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

MPC8240 microprocessors are products of Motorola, Inc.