User's Guide

Publication number E8136-97000
July 2001

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

Logic Analysis Support for the Motorola M•CORE Rainbow with Nexus3 Port

Logic Analysis Support for the Motorola M•CORE with Nexus3 Port— At a Glance

This book documents the Agilent Technologies inverse assembler for Motorola M•CORE Rainbow processor with Nexus3 port.

Inverse Assembler Software

The Agilent Technologies E8136A inverse assembler, in conjunction with an Agilent Technologies logic analyzer, allows you to view M•CORE assembly instructions which are executing in your target system. The inverse assembler can be configured to work with signals that are available for probing.

The inverse assembler uses the target processor's Nexus3 port to reconstruct the full software trace. The target processor can run full speed without being interrupted from internal flash or cache while processor execution trace is captured.

The inverse assembler model number is Agilent Technologies E9616A Option 001 when ordered alone. It is identified as "E8136A" in the Setup Assistant.

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.

Additional Information Sources

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

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

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

In This Book

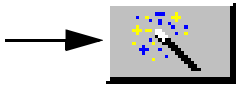
This book documents the following products:

Inverse Assembler Software		
Processors supported	Product ordered	Includes
Motorola M•CORE Rainbow with Nexus3 Port	Agilent Technologies E9616A Option #001 inverse assembler	E8136A inverse assembler

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 17.



Use the appropriate Run button

To prepare the logic analyzer to take a measurement, use the 

button on the logic analysis system.

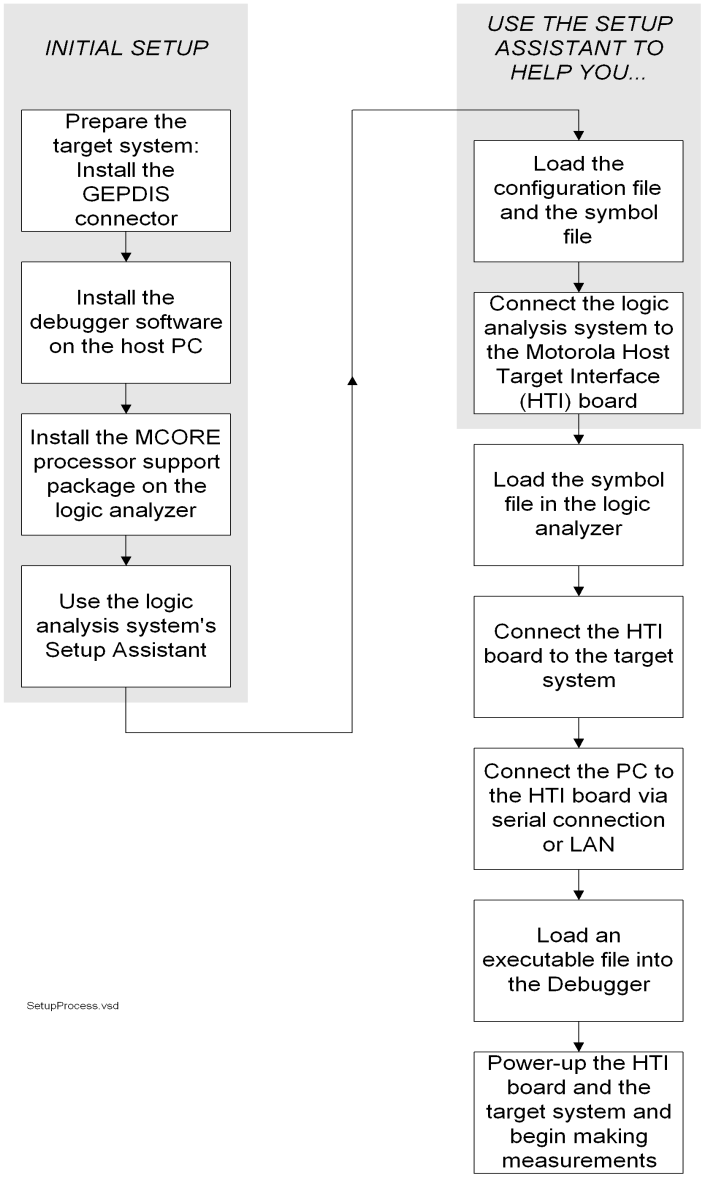
To run the target system, use the run button in the Metrowerks Code Warrior interface.

1

Overview

Setup Checklist

Follow these steps to connect and configure your equipment:



Setup Checklist References

- Prepare the target system: Install the GEPDIS connector—
  See Chapter 2, “Preparing the Target System,” beginning on page 23.
- Install the debugger software on the host PC—
  See the Metrowerks Code Warrior documentation.
- Install the M•CORE processor support package—
  See “To install the software from CD-ROM” on page 32.
- Use the logic analysis system’s setup assistant—
  See “Setup Assistant” on page 17.
- Connect the logic analysis system to the HTI board—
  Follow the instructions in the Setup Assistant.
- Load the symbol file in the logic analyzer—
  See “To load object file symbols” on page 51.
- Connect the HTI board to the target system—
  See the documentation provided with the Metrowerks Code Warrior debugger board.
- Connect the PC to the HTI board via a serial connection or LAN—
  See the documentation provided with the Metrowerks HTI board.
- Load and executable file into the debugger—
  See the documentation provided with the Metrowerks HTI board.
- Power up the target system and begin making measurements.
  See Chapter 6, “Capturing Processor Execution,” beginning on page 61 for information on setting up a trigger.
  For measurement examples, choose Help —> On Main System —>Measurement Examples in the logic analysis system online help.

System Overview

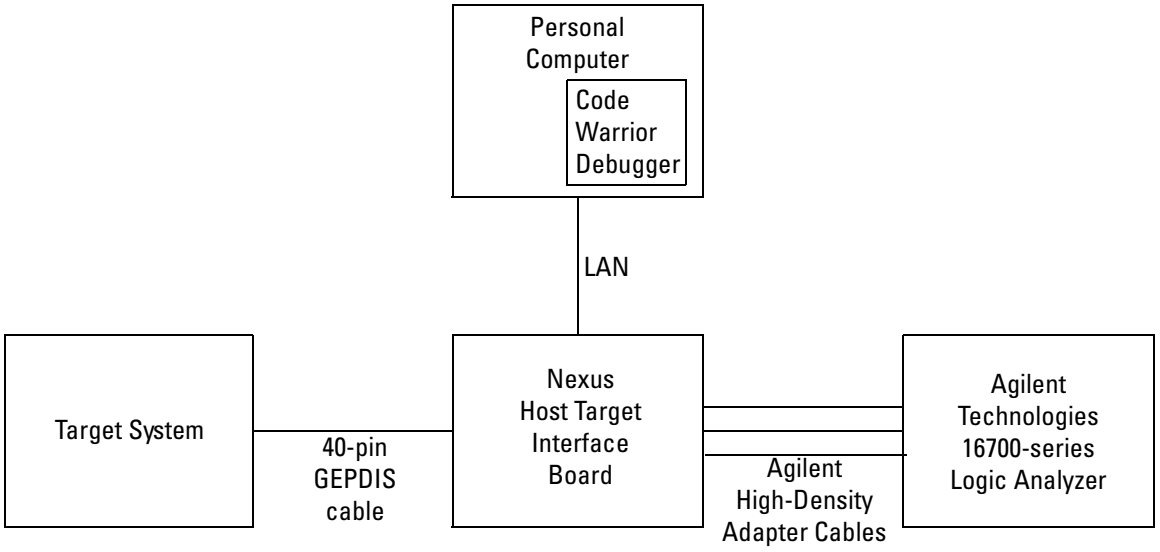
The Agilent Technologies E8136A inverse assembler works in conjunction with the Nexus3 Host Target Interface (HTI) board, which is manufactured by Metrowerks Corporation, a Motorola Company.

Metrowerks Code Warrior provides run control and a debugger interface.

When source code is compiled, an image file is generated. The inverse assembler uses the image file (in ELF format) combined with captured data from the HTI to reconstruct program trace. The Code Warrior debugger running on a PC provides run control.

Address, data and status information are conditioned by the HTI board, and fed to the logic analyzer where the program trace is reconstructed.

System Block Diagram

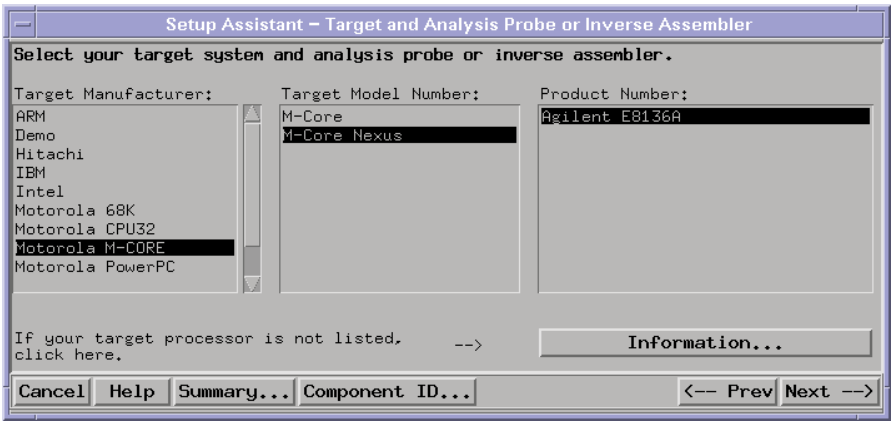


Setup Assistant

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

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

Start the Setup Assistant by selecting  in the system window.



If you ordered this inverse assembler software with your 16700-series logic analysis system, the logic analysis system has the latest software installed, including support for this product. If you received this product after you received your logic analysis system, see “Installing and loading software” on page 31.

Inverse assembler software

This section lists equipment supplied with the inverse assembler software and equipment requirements for using the inverse assembler software.

Equipment supplied

The E8136A consists of the following:

- Logic analyzer configuration files and the inverse assembler on a CD-ROM (for 16700 series logic analysis systems)
- This User's Guide

Minimum equipment required

For state and timing analysis of an M•CORE Rainbow with Nexus3 port target system, you need all of the following items:

- The E8136A M•CORE Rainbow with Nexus3 Inverse Assembler
- The Motorola Host-Target Interface (HTI) board for M•CORE Rainbow
- Three Agilent Technologies E5346A High Density Adapter Cables
- The Metrowerks Code Warrior debugger and a personal computer
- A LAN cable
- The GEPDIS connector on the target system
- An Agilent 16700-series logic analysis system with software version listed on page 20

Additional equipment supported

B4620B Source Correlation Tool Set

The inverse assembler software may be used with the B4620B Source Correlation Tool Set on an 16700-series logic analysis system.

Compatible Logic Analyzers

The table below lists the logic analyzers that can be used with the E8136A inverse assembler.

The E8136A inverse assembler requires six logic analyzer pods (102 channels) for inverse assembly with data trace, or four logic analyzer pods (68 channels) for inverse assembly without data trace (instruction and ownership trace only).

Logic Analyzers Supported

Logic Analyzer	Pods per card	Number of cards required for inverse assembly (NO DATA TRACE)	Number of cards required for inverse assembly (WITH DATA TRACE)
16752A	4	1	2
16751A	4	1	2
16750A	4	1	2
16719A	4	1	2
16718A	4	1	2
16717A	4	1	2
16716A	4	1	2
16715A	4	1	2
16712A	6	1	1
16711A	6	1	1
16710A	6	1	1

Logic analyzer software version requirements

The logic analyzers must have software with a version number greater than or equal to those listed below to make a measurement with the E8136A. You can obtain the latest software at the following web site:

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

If your software version is older than those listed, load new system software with the higher version numbers before loading the E8136A software.

Logic Analyzer Software Version Requirements

Logic Analyzer	Minimum Logic Analyzer Software Version for use with E8170B
16700-series	The latest 16700-series logic analysis system software version is on the CD-ROM shipped with this product. Must be version 2.40 or higher.

Additional Information Sources

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

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

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

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

The **measurement examples** include valuable tips for making emulation and analysis measurements. You can find the measurement examples in the online help in your 16700 logic analysis system.

2

Preparing the Target System

Target System Requirements

Inverse assembly for the M•CORE Rainbow with Nexus3 Port requires connections as shown in the figure titled “System Block Diagram” on page 16.

The 40-pin GEPDIS connector is described in this chapter.

40-Pin GEPDIS connector on target system

The 40-pin Global Embedded Processor Debug Interface Standard (GEPDIS) connector must be fitted on the target system so a connection can be made between the target system and the Motorola Host Target Interface (HTI) board. Use a 40-conductor GEPDIS cable to make the connection.

The 40-pin GEPDIS connector is an AMP AMPMODU System 50 Header (surface mount), order code 104549-6. The pinout for the 40-pin GEPDIS connector is shown below.

See Also

See the documentation for the Motorola Host Target Interface board for more information.

40-Pin GEPDIS Connector Pinout

Signal Name IEEE 1149.1	Signal Name (Motorola)	Direction	Nexus Connector Pinout Diagram		Direction	Signal Name* (Motorola)
/RESET	/RESET	IN	1	2	-	VREF
/EVTI	/EVTI	IN	3	4	-	VALTREF
/TRST	/RSTI	IN	5	6	IN or OUT	VENDOR_IO1
TMS	/MSEI	IN	7	8	-	GND
TCK	MCKI	IN	9	10	-	GND
TDI	MDIO	IN	11	12	-	GND
TDO	RESERVED	OUT	13	14	-	GND
/RDY	RESERVED	OUT	15	16	IN or OUT	VENDOR_IO2
/EVTO	/EVTO	OUT	17	18	-	GND
/MSEO	/MSEO	OUT	19	20	-	GND
MCKO	MCKO	OUT	21	22	-	GND
MDO0	MDO0	OUT	23	24	-	GND
MDO1	MDO1	OUT	25	26	-	GND
MDO2	MDO2	OUT	27	28	-	GND
MDO3	MDO3	OUT	29	30	-	GND
MDO4	MDO4	OUT	31	32	-	GND
MDO5	MDO5	OUT	33	34	-	GND
MDO6	MDO6	OUT	35	36	-	GND
MDO7	MDO7	OUT	37	38	-	GND
MDI1	MDI1	IN	39	40	-	GND

* VREF: Voltage reference for tool signal level.
VALTREF: Alternative voltage reference for calibration tools, SRAM voltage, etc.
VENDOR_IOx: I/O signal defined by vendor (Motorola). Must be a mainly static signal (e.g. flash program enable).

Note: For reduced pin count use pins 1..26.

3

Setting Up the Logic Analysis System

Power-on/Power-off Sequence

Listed below are the sequences for powering on and off a fully connected system.

To power on

Ensure the target system and HTI board are 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 target system is connected to the logic analyzer, and everything is configured, turn on the Host Target Interface board.
3. Turn on the target system.

To power off

Turn off power to your system in the following order:

1. Turn off the target system.
2. Turn off the Host Target Interface board.
3. Turn off the 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 (ESD) can damage electronic components. Use appropriate ESD equipment (grounded wrist strap, etc.) and ESD-safe procedures when you handle and install modules.

Refer to your logic analysis system's *Installation Guide* for instructions on installing logic analyzer modules.

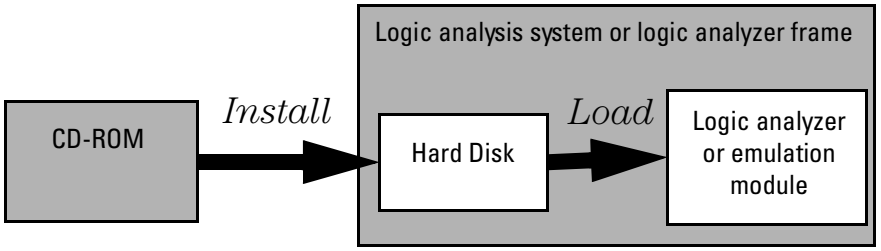
Software Requirements

The following software is required when using the inverse assembler:

- The Metrowerks Code Warrior debugger, for use with the Motorola Host Target Interface board.
- The Agilent Technologies 16700-series logic analysis system software version shown on page 20.

Installing and loading software

Installing the software will copy 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. (The easiest way to **load** files is using the Setup Assistant described on page 17.)



What needs to be installed

16700-series logic analysis systems

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

The B4620B Source Correlation Tool Set is installed with the logic analysis system's operating system. A password may be required to enable the tool set. Follow the instructions on the entitlement certificate.

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 16700 operating system, installation may take approximately 15 minutes.

NOTE: Some older systems use an external CD-ROM drive. If the CD-ROM drive is not connected, refer to the instructions printed on the CD-ROM package.

1. 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.

   If your system uses an external CD-ROM drive, switch it on first, then switch on 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.

NOTE: If your system requires an operating system update a dialog box will appear which instructs you how to update the operating system. Follow the directions in the dialog box.

7. From the list of types of packages, double-click "PROC-SUPPORT."

NOTE: For touch screen systems you must first un-select all of the package type lines, then 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 "mcore" package.

   If you are unsure whether this is the correct package, select Details for

information about the contents of the package.

9 Select Install.

The Continue dialog box will appear.

10 Select Continue.

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

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

The configuration files are stored in /logic/configs/hp/mcore_nexus. The inverse assemblers are stored in /logic/ia.

See Also

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

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

To list software packages which are installed

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

4

Connecting the Logic Analyzer to the HTI board

Connecting the Logic Analyzer to the Motorola HTI board

This section shows the connections between the logic analyzer pod cables and the target system. There are two types of analysis available:

32-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.

- 16715/16/17/18/19A logic analyzers - two cards (see page 40)
- 16710/11/12A logic analyzers - one card (see page 41)

No data

This type of analysis allows you to trace program flow only. Use the appropriate page listed below for your logic analyzer.

- 16715/16/17/18/19A logic analyzers - one card (see page 39)

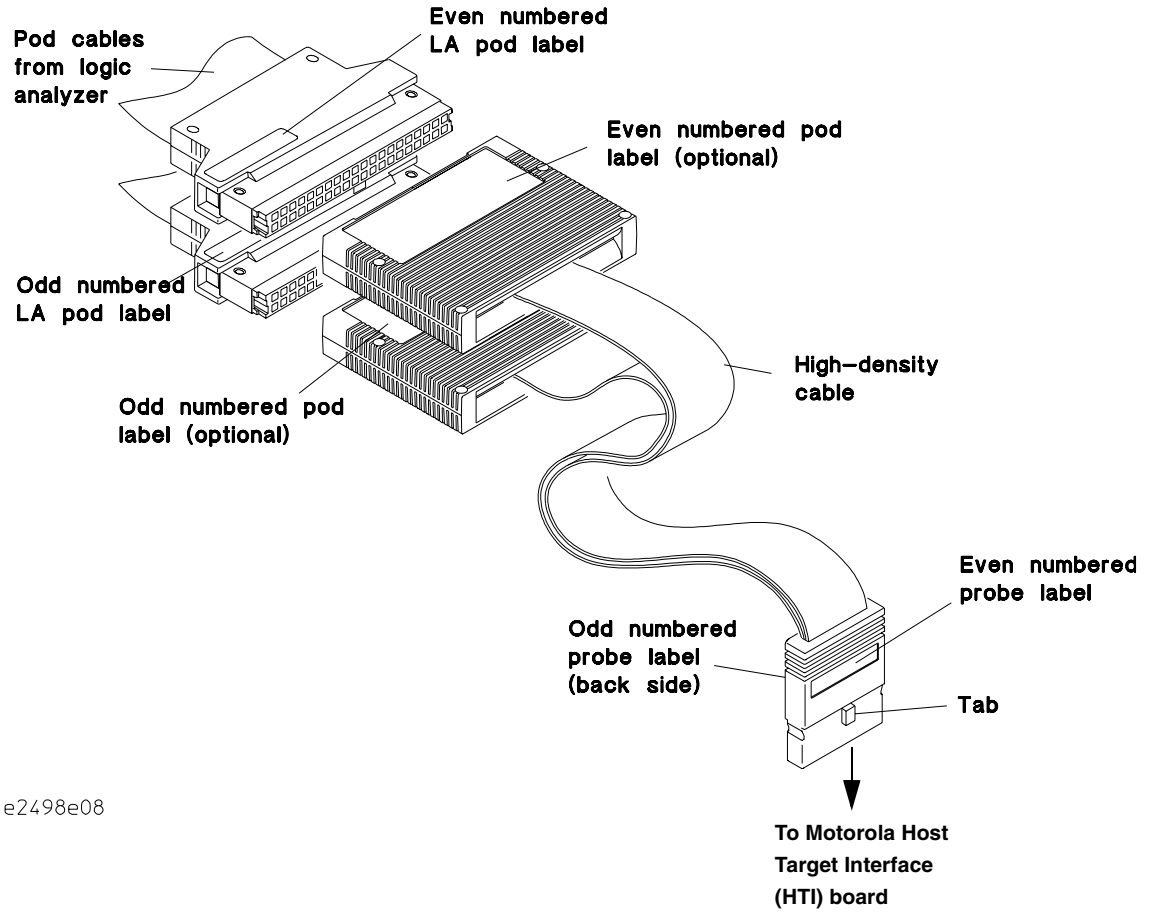
NOTE: To connect logic analyzers not listed here use the Setup Assistant, as described on page 17.

NOTE: A configuration file must be loaded to map logic analyzer channels to the MICTOR connectors on the HTI board. The configuration file names are listed on page 44.

AMP MICTOR (*M*atched *I*mpedance *C*onnec*TOR*) connectors are provided on the Motorola Host Target Interface (HTI) board for connection to the logic analysis system (reference "System Block Diagram" on page 16). Use three Agilent Technologies E5346A High Density Adapter Cables to connect the logic analysis system pods to the HTI board.

The Agilent E5346A 2x19 high-density termination cables include labels to identify them, as shown in the following illustration.

E5346A High-density Termination Cables



MICTOR Signal-to-connector mapping

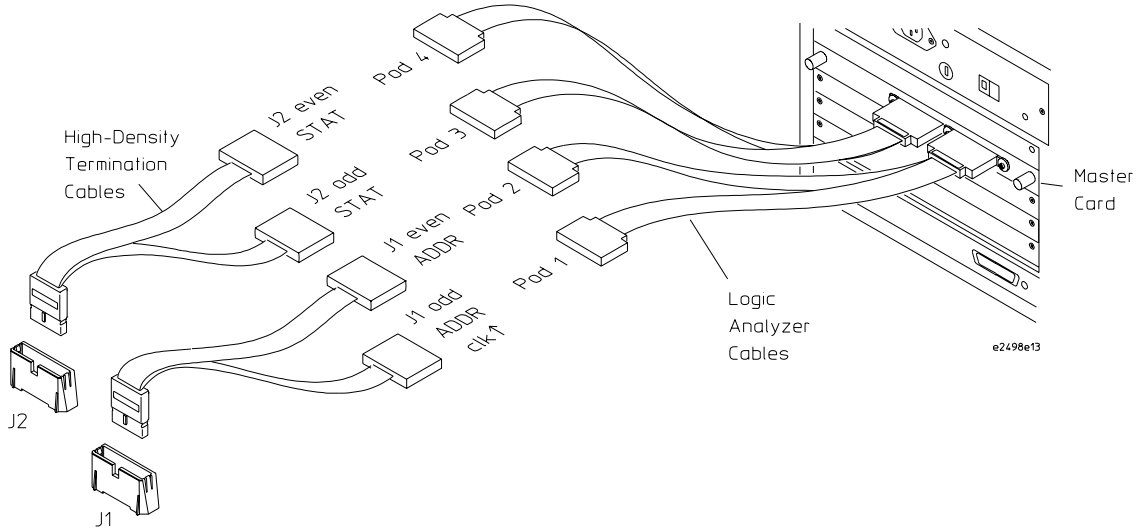
The following tables show the electrical signal-to-connector mapping required by the Motorola Host Target Interface board and the Agilent Technologies E8136A Inverse Assembler Software.

When you use the Setup Assistant to connect and configure your system, the configuration file will be loaded. The configuration file contains this signal-to-connector mapping information.

TCODE/PKT1 HTI MICTOR				ADDR/PKT2 HTI MICTOR				DATA/PKT3 HTI MICTOR			
Odd		Even		Odd		Even		Odd		Even	
Pin	M•CORE Signal	Pin	M•CORE Signal	Pin	M•CORE Signal	Pin	M•CORE Signal	Pin	M•CORE Signal	Pin	M•CORE Signal
2	NC	1	NC	2	NC	1	NC	2	NC	1	NC
4	NC	3	NC	4	NC	3	NC	4	NC	3	NC
6	NC	5	EVTO	6	NC	5	MSG_STB	6	NC	5	NC
8	NC	7	TCODE[0]	8	A[16]	7	A[0]	8	D[16]	7	D[0]
10	NC	9	TCODE[1]	10	A[17]	9	A[1]	10	D[17]	9	D[1]
12	NC	11	TCODE[2]	12	A[18]	11	A[2]	12	D[18]	11	D[2]
14	NC	13	TCODE[3]	14	A[19]	13	A[3]	14	D[19]	13	D[3]
16	NC	15	TCODE[4]	16	A[20]	15	A[4]	16	D[20]	15	D[4]
18	NC	17	TCODE[5]	18	A[21]	17	A[5]	18	D[21]	17	D[5]
20	NC	19	NC	20	A[22]	19	A[6]	20	D[22]	19	D[6]
22	NC	21	NC	22	A[23]	21	A[7]	22	D[23]	21	D[7]
24	NC	23	COUNT[0]	24	A[24]	23	A[8]	24	D[24]	23	D[8]
26	NC	25	COUNT[1]	26	A[25]	25	A[9]	26	D[25]	25	D[9]
28	NC	27	COUNT[2]	28	A[26]	27	A[10]	28	D[26]	27	D[10]
30	NC	29	COUNT[3]	30	A[27]	29	A[11]	30	D[27]	29	D[11]
32	NC	31	COUNT[4]	32	A[28]	31	A[12]	32	D[28]	31	D[12]
34	NC	33	COUNT[5]	34	A[29]	33	A[13]	34	D[29]	33	D[13]
36	NC	35	COUNT[6]	36	A[30]	35	A[14]	36	D[30]	35	D[14]
38	NC	37	COUNT[7]	38	A[31]	37	A[15]	38	D[31]	37	D[15]
Logic Analyzer Pod 3		Logic Analyzer Pod 4		Logic Analyzer Pod 1		Logic Analyzer Pod 2		Logic Analyzer Pod 5		Logic Analyzer Pod 6	

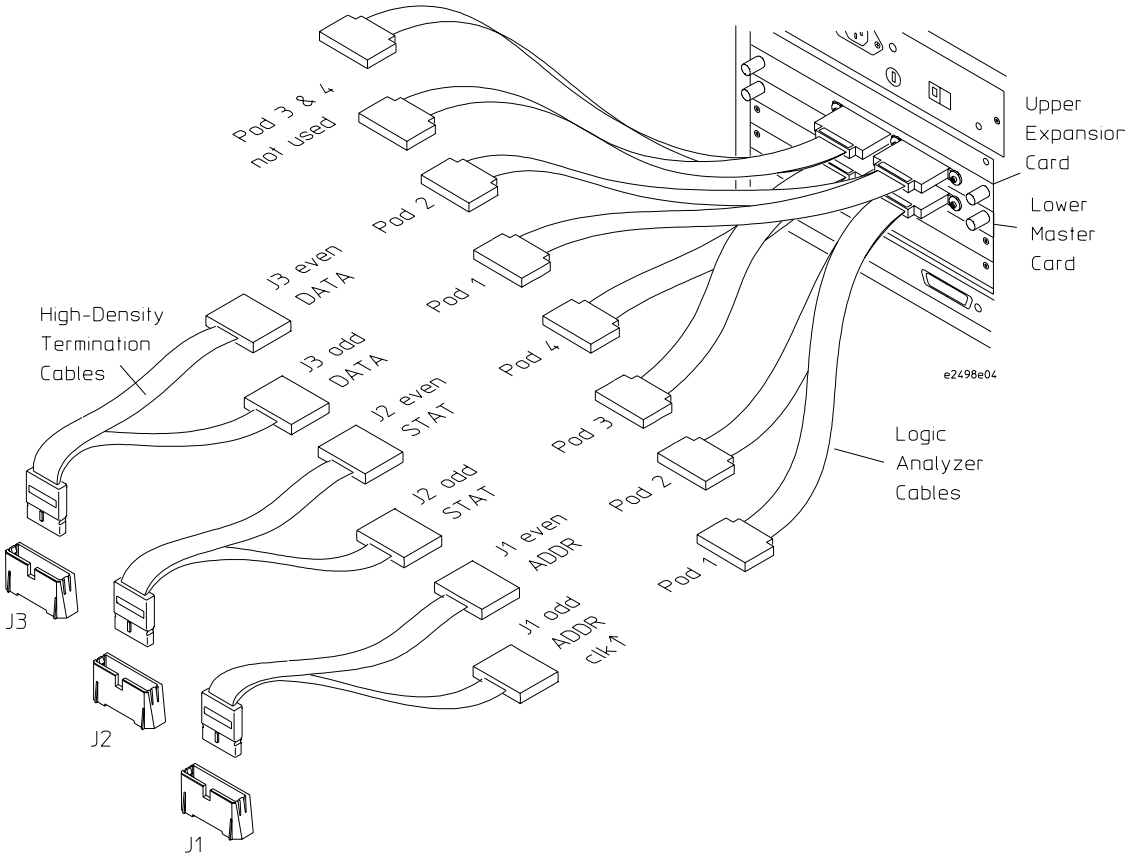
To connect to the 16715/16/17/18/19/50/51/52A logic analyzer (one card)

Use the figure below to connect the target system to the 16715/16/17/18/19A logic analyzer. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



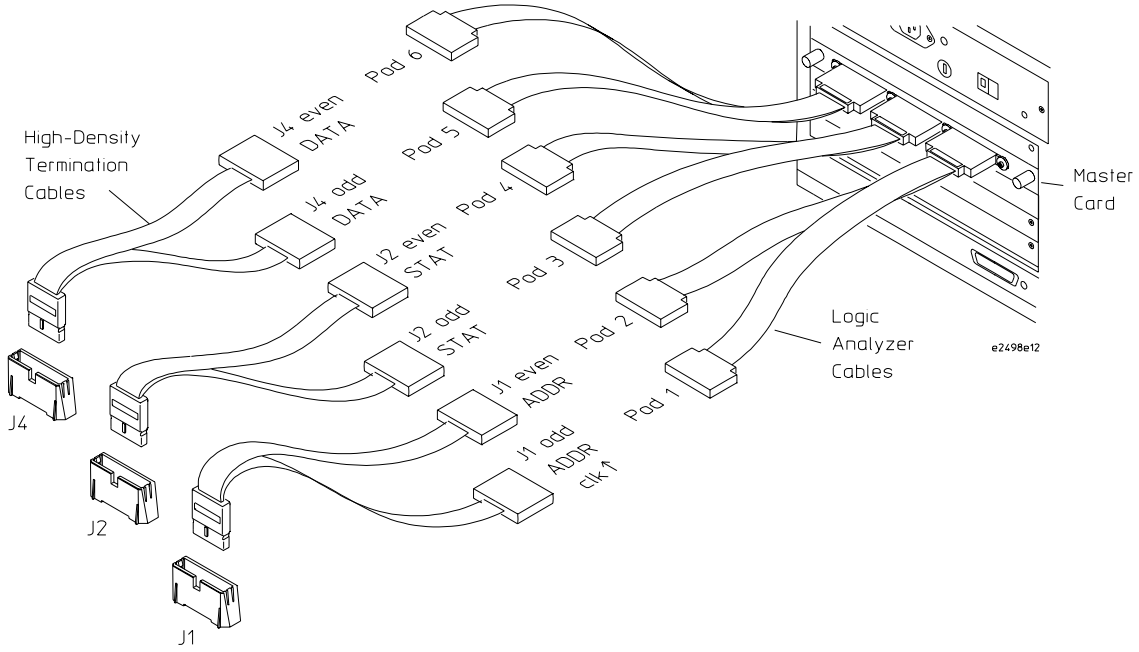
To connect to the 16715/16/17/18/19/50/51/52A logic analyzer (two cards)

Use the figure below to connect the target system to the 16715/16/17/18/19A logic analyzer. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



To connect to the 16710/11/12A logic analyzer (one card)

Use the figure below to connect the target system to the 16710/11/12A logic analyzer. Find the labels that were shipped with the high-density cables and use them to help identify the connections.



5

Configuring the 16700-series Logic Analyzer

Loading Configuration and Inverse Assembler Files from the System Hard Disk

The easiest way to load configuration and inverse assembler files is by using the Setup Assistant. If you choose to use Setup Assistant instead, it will load the configuration file and inverse assembler for you. See page 17.

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

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

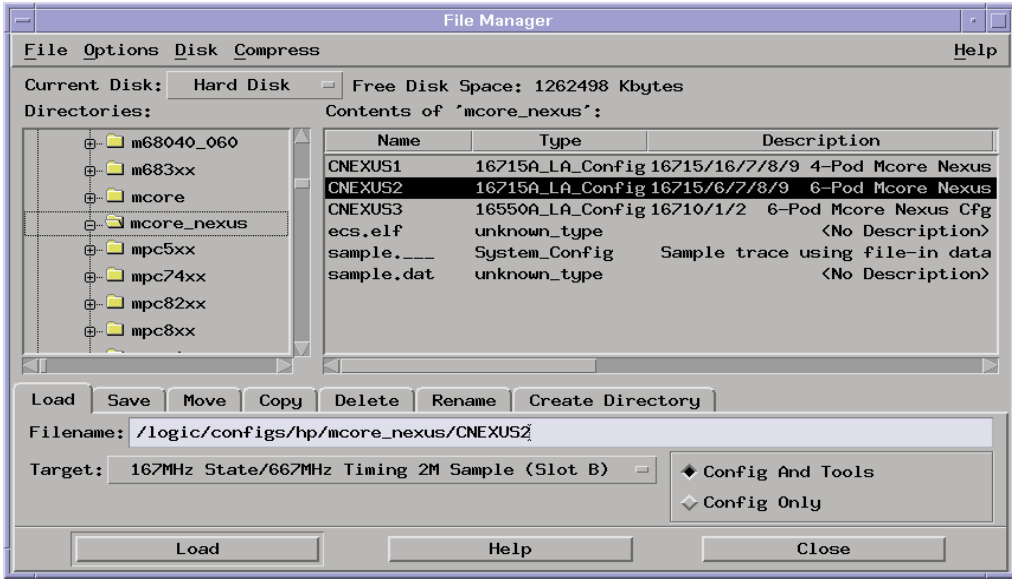
If the above directory does not exist, you need to install the MCORE Rainbow Processor Support Package. Close File Manager, then use the procedure on page 32 to install the MCORE Rainbow Processor Support Package before you continue.

2 Using File Manager, select the configuration file you want to load in the /logic/configs/hp/mcore_nexus/ 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.

Logic Analyzer Configuration Files

Configuration File	Logic Analyzer Model	Logic Analyzer Cards Available
CNEXUS1	16715/16/17/18/19/50/51/52A	1*
CNEXUS2	16715/16/17/18/19/50/51/52A	2 or more
CNEXUS3	16710/11/12A	1 or more*

*Note: When using a single 16715/16/17/18/19/50/51/52A logic analyzer module (4 pods), data trace is not supported. Only instruction trace is supported with 4 pods.



The logic analyzer is configured for M•CORE Rainbow with Nexus3 analysis by loading the appropriate M•CORE Rainbow Nexus3 configuration file. Loading the indicated file also automatically loads the correct inverse assembler.

3 Close File Manager.

A sample M•CORE Rainbow with Nexus3 workspace (named sample._ _ _) is available in the directory /logic/configs/hp/mcore_nexus. This sample workspace will load a sample data file and the M•CORE Rainbow with Nexus3 Tool Development Kit for demonstration purposes.

Logic Analyzer Configuration

The following sections describe the logic analyzer configuration as set up by the configuration files. If you use the Setup Assistant, it will load the correct configuration file for you.

It is strongly recommended that you do not change the setup related to the sampling, format, pod assignment or configuration dialogs. The configuration file (loaded by the Setup Assistant in 16700-series logic analysis systems) will configure the logic analyzer for making measurements.

Configuring the Logic Analysis System

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

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

The configuration file you use is determined by the logic analyzer you are using. The configuration file names are listed in the table on page 44.

Using the Format menu

Labels can be created using the Format menu. A label consists of a name and an associated bit or group of bits. Labels are used to group and identify logic analyzer channels.

This section describes the organization of M•CORE Rainbow signals in the logic analyzer's Format menu.

The configuration files contain predefined format specifications. These format specifications include all labels for monitoring the microprocessor.

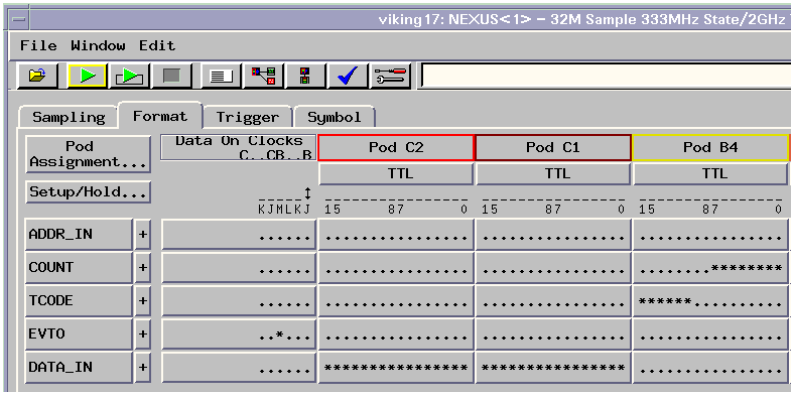
Do not modify the labels in the format specification if you want inverse assembly. Changes to these labels may cause incorrect or incomplete inverse assembly.

Bit ordering conventions

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

Most Significant	Least Significant	
A0	**A31**	***M•CORE Rainbow***
ADDR31	**ADDR0**	***Logic Analyzer***
This may cause confusion in the waveform window when using Channel Mode Sequential or Individual.		

Loading the configuration file sets up the analyzer format dialog to display the correct number of pods of data, depending on the analyzer.



Labels for Status Bits

The logic analyzer sets up the following labels when the configuration file is loaded. The STAT label appears in the Listing window.

Signals within the STAT label

Status Bit	Description
ADDR_IN	This is the raw address capture from the HTI board. The reconstruction tool generates ADDR from this. The ADDR label does not appear in the Format dialog; it appears in the Listing Window.
COUNT	The number of instructions executed since the last branch was taken. Used to generate STAT, which appears in the listing window.
TCODE	Transfer format, number or size of packets to be transferred, and purpose of each packet. See Motorola's Nexus3 Specification for details. Used to generate STAT, which appears in the listing window.
$\overline{\text{EVTO}}$	Event Out. Asserted low by the CPU when a software breakpoint is encountered. Used to generate STAT, which appears in the listing window.
DATA_IN	This is the raw data capture from the HTI board. The reconstruction tool generates DATA from this. The DATA label does not appear in the Format dialog; it appears in the Listing Window.

Symbols

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

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

You must download symbols into the logic analyzer in ELF object file format in order to use the M•CORE Rainbow with Nexus3 inverse assembler.

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

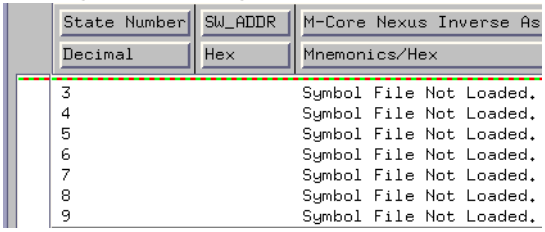
The B4620B Source Correlation Tool Set lets you display the high-level source code associated with captured data.

An ELF file is required for inverse assembly/trace reconstruction.

The compiler you use must be capable of generating an object file in ELF format. The ELF file contains symbols associated with the source code.

You should load the ELF file before attempting to use the M•CORE Rainbow with Nexus3 inverse assembler/trace reconstruction tool. The message "Symbol File Not Loaded" will appear in the Listing window if you attempt to use the inverse assembler without the executable's associated ELF file.

Message in the Listing Window



See "To load object file symbols" on page 51.

Types of symbols

Three symbol sources may be used in the logic analyzer:

- Object-file symbols
- Predefined M•CORE Rainbow symbols
- User-defined symbols

Object file symbols

The most common way to load program symbols into the logic analyzer is from an object file that is created when the program is compiled. The ELF file, which is required for trace reconstruction, contains the symbols associated with the source code. See “To load object file symbols” on page 51.

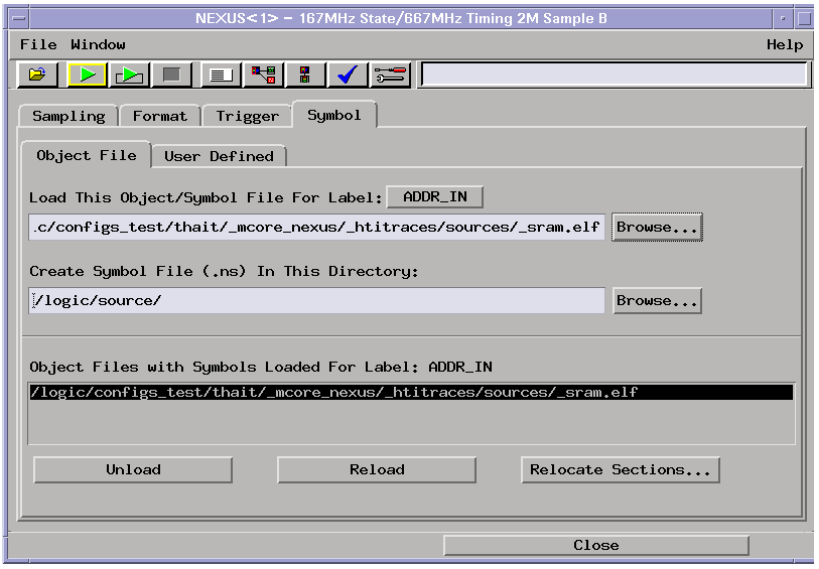
To load object file symbols

To load object file symbols for address values in the 16700-series logic analysis system:

1. Open the logic analyzer module's Setup window.
2. Select the Symbol tab.
3. Select the Object File tab.
4. Make sure the label is ADDR_IN.

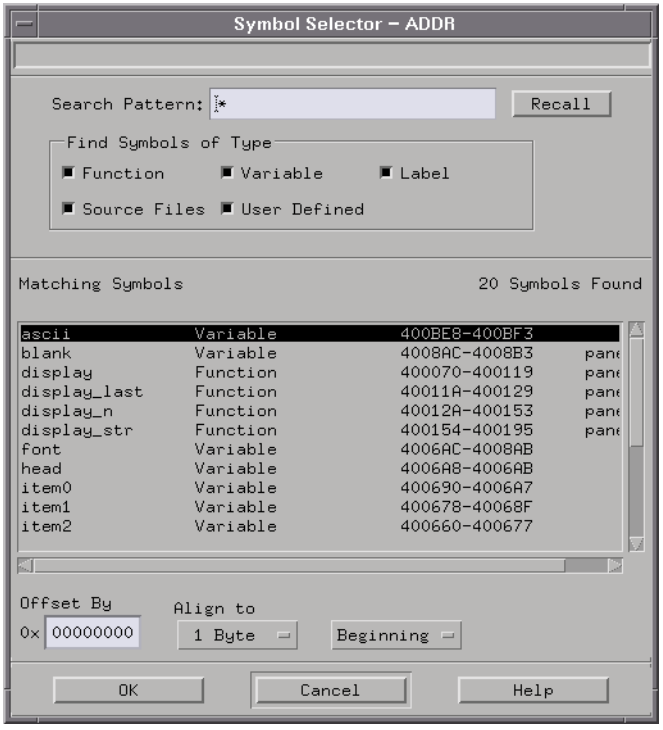
   If the ADDR_IN label is not available, you must first load the configuration file. See "Loading Configuration and Inverse Assembler Files from the System Hard Disk" on page 44 or "Setup Assistant" on page 17.
5. From this dialog you can select object files and load their symbol information. There are two Browse buttons. Click the Browse button for Load This Object/Symbol File For Label, and select and load the ELF file you want to load.
6. Close the logic analyzer module Setup window.
7. Open the Workspace window. Select the M•CORE Nexus icon, then select Display....
8. Select the Execute button, then select Close. The new ELF file is now used in the Listing window.

NOTE: The HTI board only generates ADDR_IN for indirect branches. Therefore, symbols will only appear when indirect branches occur, and you will only be able to trigger on indirect branches. Symbols will be displayed for all software addresses generated by the inverse assembler under the SW_ADDR label.



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

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.



See Also

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

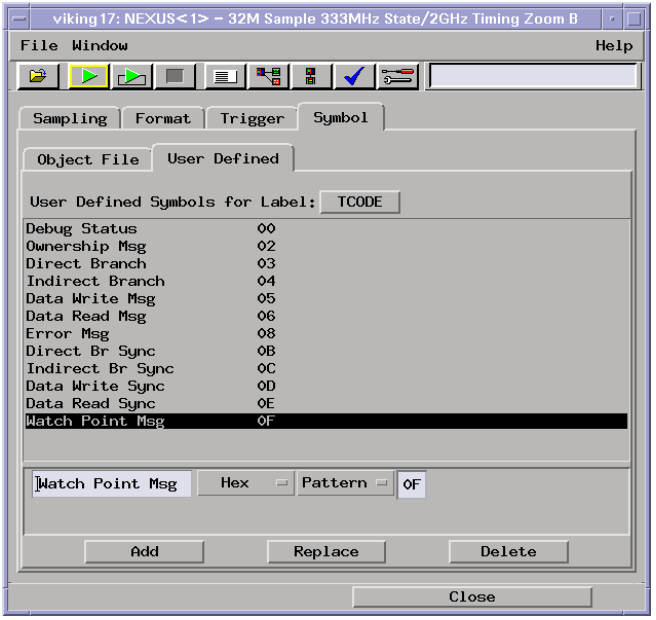
- If you have a 16700-series logic analysis system, refer to the online help.
- If you have another logic analyzer, refer to your logic analyzer documentation.

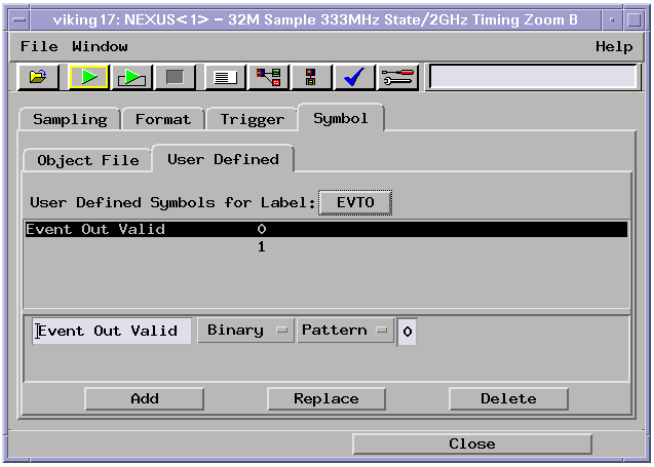
Predefined M•CORE Rainbow symbols

If you're using an Agilent inverse assembler for the M•CORE Rainbow microprocessor, the logic analyzer configuration files include predefined symbols. These symbols appear along with the user-defined symbols in the logic analyzer.

Predefined Logic Analyzer Symbols

The configuration software sets up symbol tables on the logic analyzer. The tables define a number of symbols which make several of the STAT fields easier to interpret. The following two illustrations show the predefined symbols, which appear under the User Defined tab in the Symbol dialog.




viking17: NEXUS<1> - 32M Sample 333MHz State/2GHz Timing Zoom B
File Window
Help
Sampling
Format
Trigger
Symbol
Object File
User Defined
User Defined Symbols for Label:
EVTO
Event Out Valid
0
1
Event Out Valid
Binary
Pattern
0
Add
Replace
Delete
Close

To create user-defined symbols

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

To view pre-defined and user-defined symbols for the M•CORE Rainbow

Both user defined and predefined symbols are shown in the User Defined symbol dialog.

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 other label values as well.

User-defined and pre-defined symbols are saved with logic analyzer configurations. The logic analyzer configuration files included with the M•CORE Rainbow inverse assembler contain predefined symbols for logic analyzer labels.

To display the predefined and user-defined symbols for the M•CORE Rainbow:

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

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

Using the Inverse Assembler

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

The E8136A inverse assembler for M•CORE Rainbow with Nexus3 uses the 32-bit address, 32-bit data (optional), TCODE and TCOUNT to reconstruct instruction flow.

To use the Invasm menu

You can access the Invasm menu in the listing window. The Invasm menu provides five choices: Load..., Unload..., Filter..., Preferences..., and Options.... The Filter... choice is discussed in “Display Filtering” on page 76.

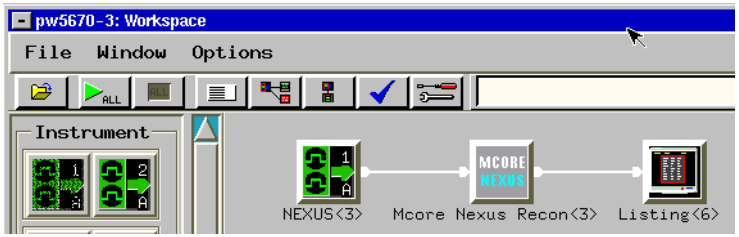
You can use the Preferences dialog to change the color of lines in the listing window that show access to particular memory regions. See “To set the memory map preferences” on page 59.

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 an 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. The E8136A inverse assembler's name is IAMCNEXUS.

This inverse assembler requires the M*CORE Nexus3 Reconstruction Tool to process the raw data before the data is sent to the Listing window. The M*CORE Nexus3 Reconstruction Tool is automatically added to the Workspace (as shown below) when you use the Setup Assistant.



The Unload... option is primarily used for diagnostic purposes. This option is not generally used during development and debug of a target system. Loading an inverse assembler effectively unloads the previous one.

To set the memory map preferences

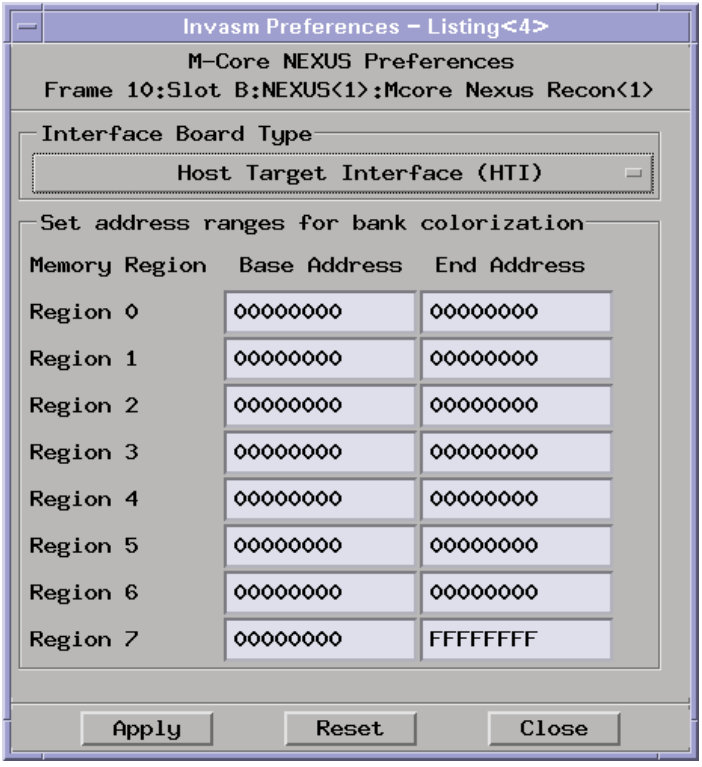
Interface Board Type. Set the Interface Board Type to Host Target Interface (HTI). The EBDI option is provided for backward compatibility with older target systems.

Memory Region. Enter the memory region information, which is used in conjunction with the Filter Dialog (see page 76) to make the listing easier to interpret by displaying memory ranges in a particular color. If the first address of a state falls within one of the ranges, this state including the state's sub-rows will be given the color defined in the Filter dialog.

Base Address. Defines the starting address of the region.

End Address. Defines the ending address of the region.

Inverse Assembler Preferences Dialog



Compilers for the M•CORE Rainbow

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

The Metrowerks Code Warrior compiler is compatible with the Agilent M•CORE Rainbow with Nexus3 port inverse assembler.

See Also

Contact your Agilent Technologies sales engineer to find out whether other compilers for the M•CORE Rainbow microprocessor can be used with Agilent logic analysis systems.

6

Capturing Processor Execution

The normal steps in using the logic analyzer are:

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

The logic analyzer is configured and labels are created (formatted) for the logic analysis channels when configuration files are loaded. See "Loading Configuration and Inverse Assembler Files from the System Hard Disk" on page 44.

You can load program object file symbols into the logic analyzer when configuring it. See "To load object file symbols" on page 51.

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

See Chapter 7, "Displaying Processor Execution," beginning on page 71 for information on displaying captured data.

Trigger sequence

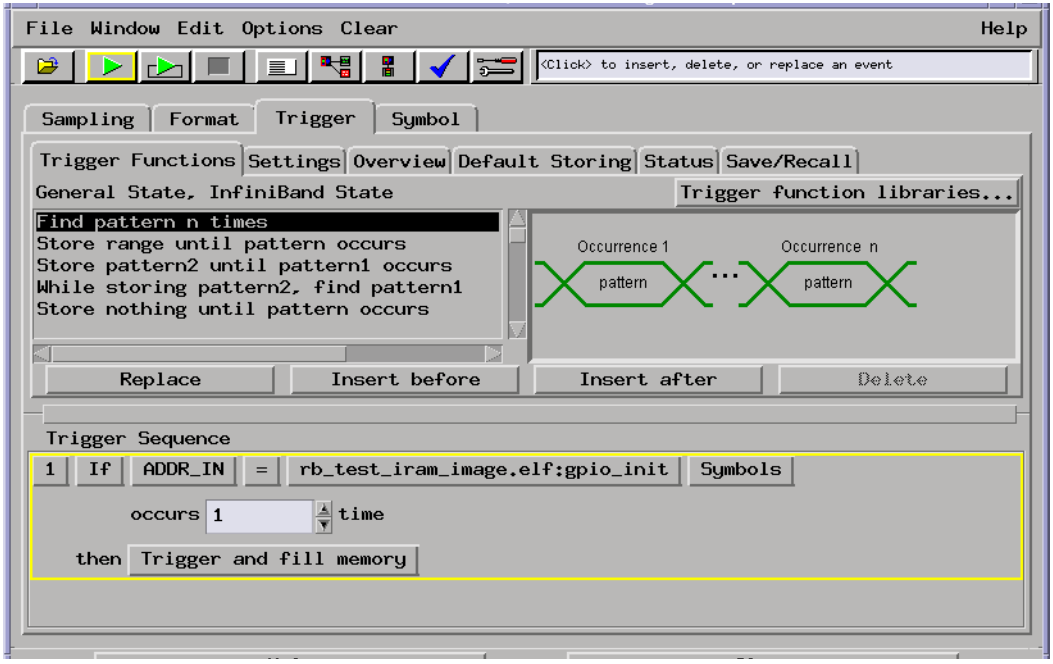
The Trigger sequence (also called trigger specification or trigger setup) is set up by the software to store all states.

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

Setting up a trigger

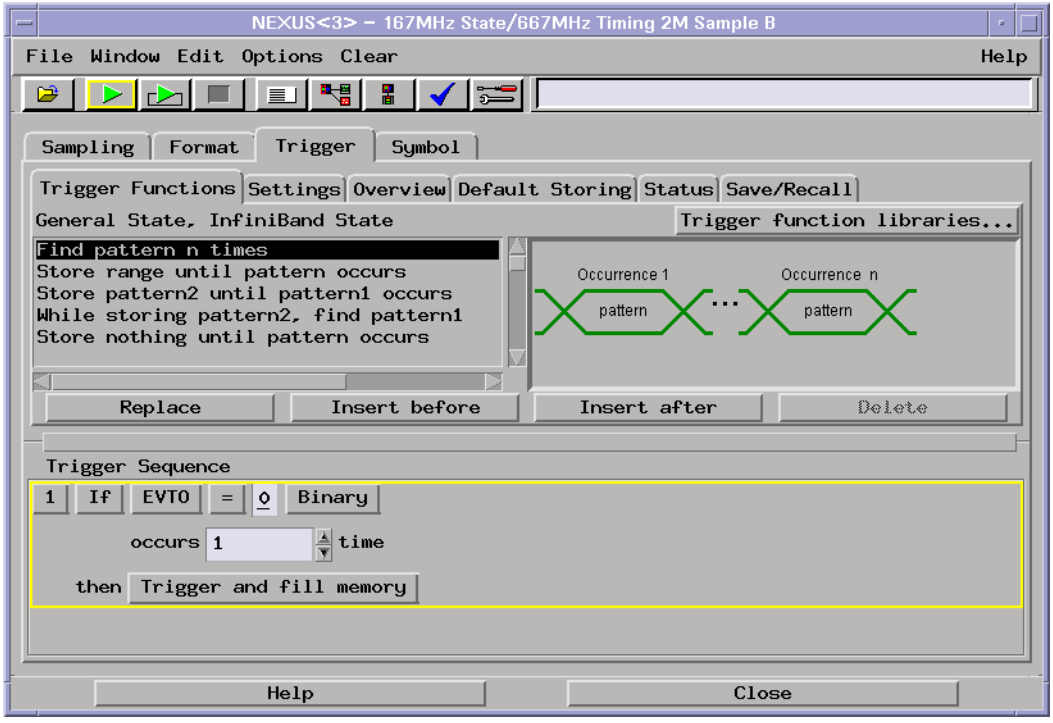
Triggering on an indirect branch address

The HTI board generates a hardware address when there is an indirect branch or a sync. The trigger condition can be set to trigger on the destination address of the indirect branch. The sequential instruction, direct branch address and the opcode can not be used as trigger points because they are generated by the inverse assembler.



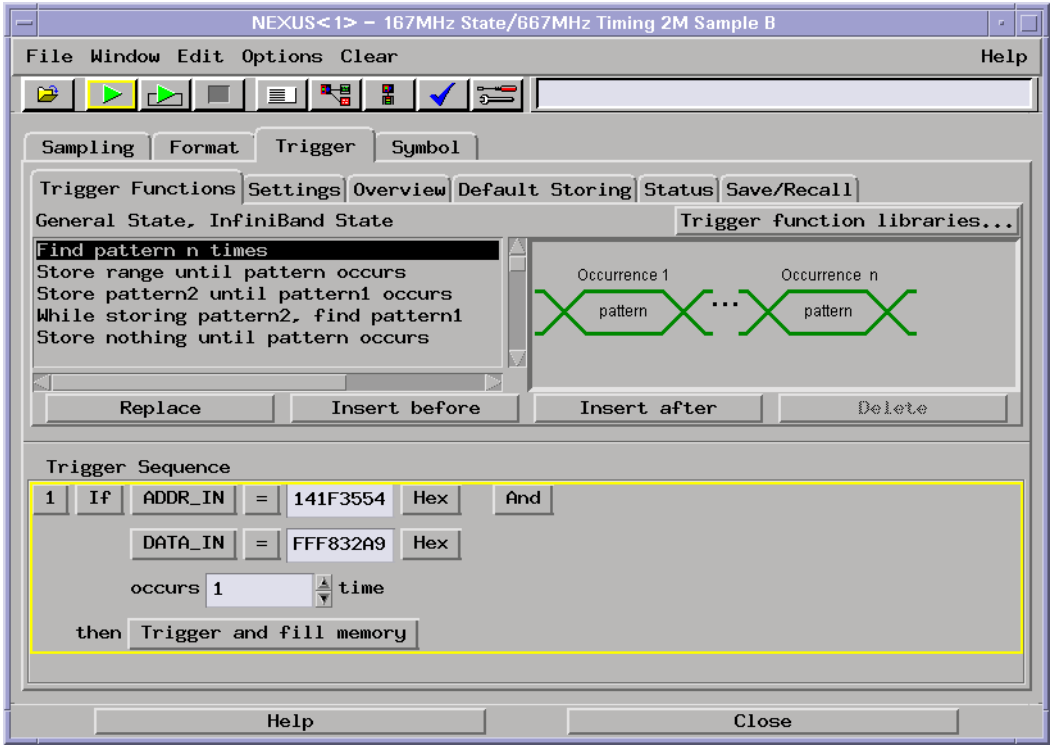
Triggering on the EVTO signal.

EVTO is asserted low when either of the two watchpoint addresses occur. The watchpoint addresses can be set in the Code Warrior run control. For more information on how to do this, please refer to Metrowerks' documentation. The trigger sequence for triggering on EVTO is shown below.



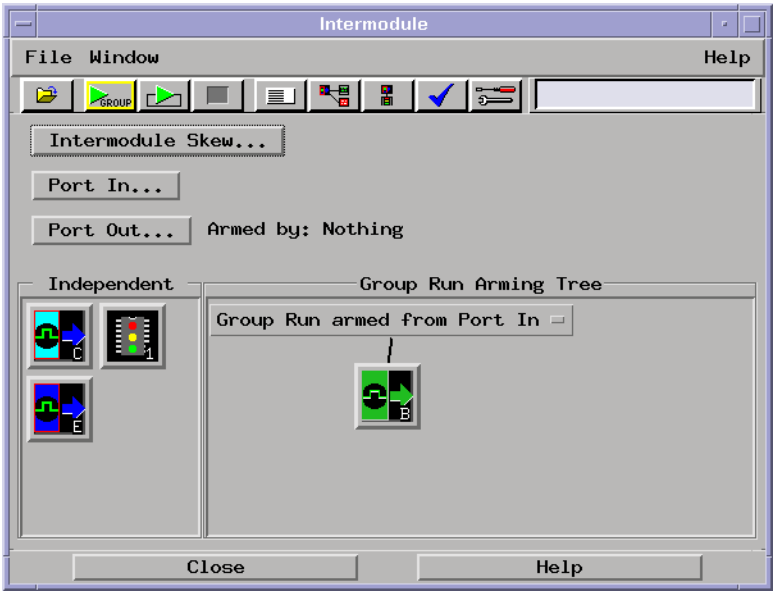
Triggering on addresses or data

The Nexus3 port can be configured to provide instruction trace, data trace, or both. When configured to trigger on data trace, the logic analyzer can trigger on either an address, data, or both. This is because the HTI board generates addresses and I/O data for all data states. The trigger sequence for triggering on an address is shown below.



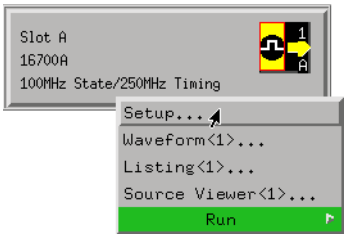
Triggering from an external signal

If your target system is capable of generating a trigger signal, you can connect the logic analyzer Port In connector for custom triggering. The trigger setup for triggering on an external signal is shown in the Intermodule window below.



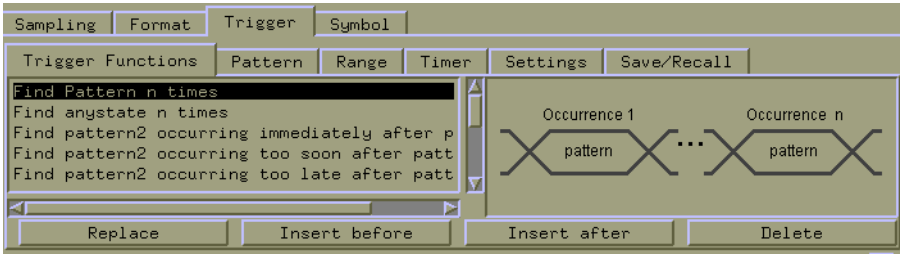
To Set Up Logic Analyzer Triggers

1 Open the logic analyzer's Setup window.

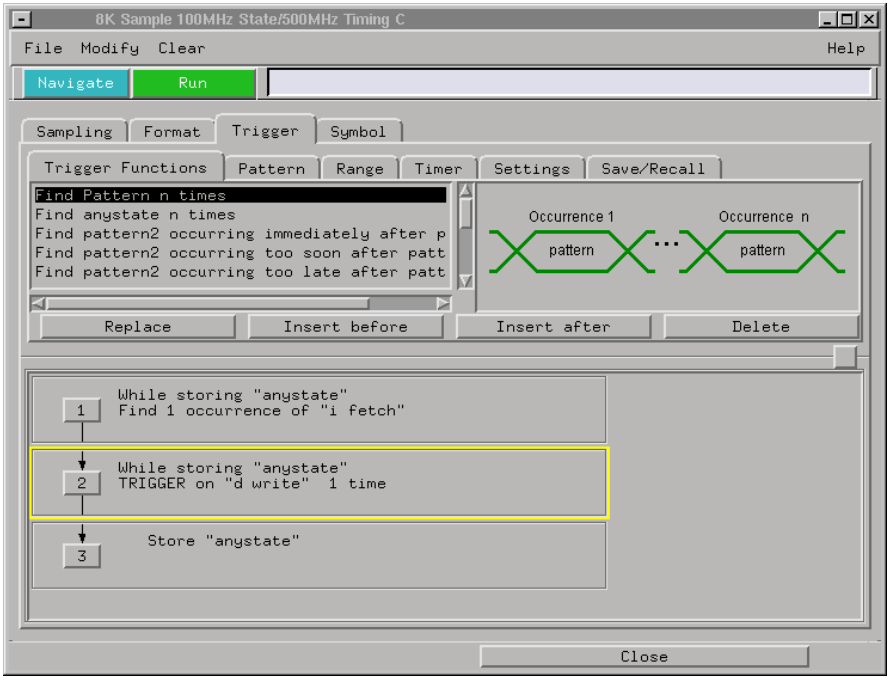


2 Select the Trigger tab.

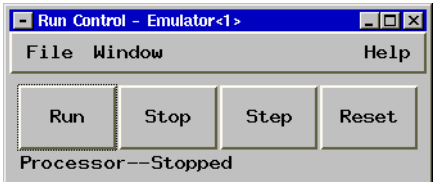
3 Select the trigger function that will be used in the logic analysis measurement.



4 Set up the trigger sequence.



5 Run the measurement.



See Also

See the Agilent 16700-series logic analysis system's on-line help for more information on setting up logic analyzer triggers.

Triggering on Symbols and Source Code

When setting up trigger specifications to capture M•CORE execution:

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

To correlate relocatable code using the address offset

You need to adjust the source correlation tool set to compensate for relocatable code segments or memory management units that produce fixed code offsets. The offset field in the trigger menu allows you to offset the symbol address. Entering the appropriate address offset will cause the source correlation tool set to reference the correct symbol information for the relocatable or offset code.

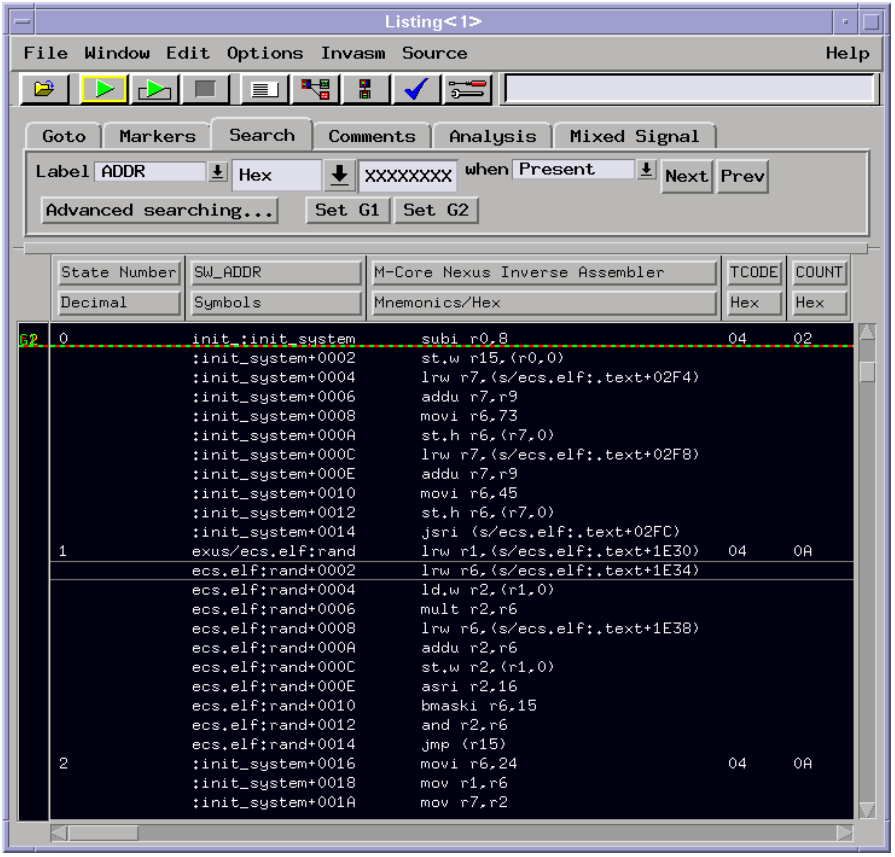
7

Displaying Processor Execution

Viewing trace data

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

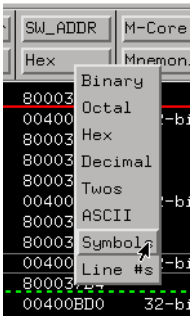
Listing Window.



On the 16700-series logic analysis systems, the entire synthesized address appears under the label “SW_ADDR”. The actual address bits presented by the M•CORE may be observed under the ADDR label.

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 51.

Inverse assembler output format

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

Interpreting Data

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

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

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

The inverse assembler decodes the full M•CORE instruction set architecture. When unimplemented opcodes are encountered, the listing displays “illegal opcode.”

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

SW_ADDR Label

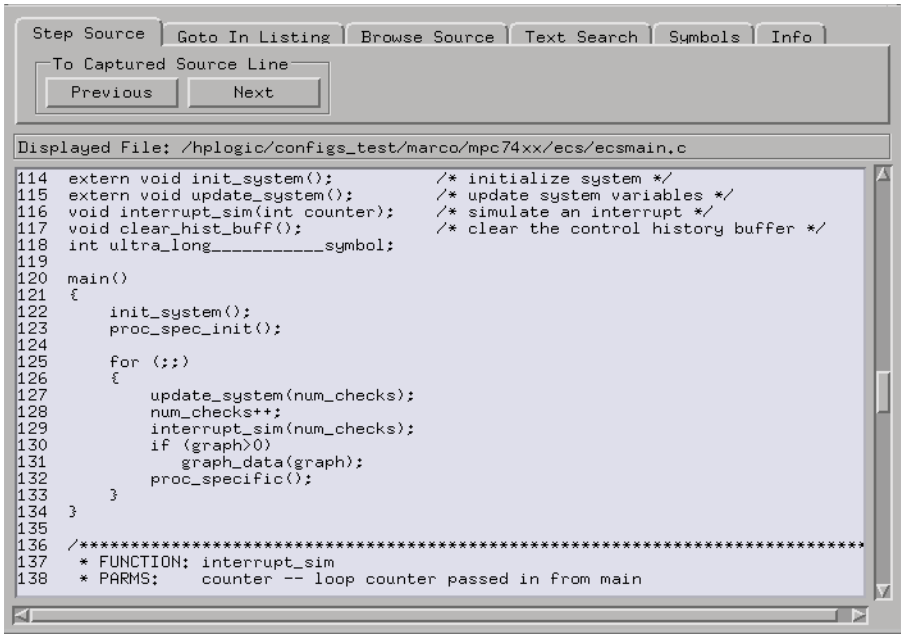
When a 16700-series logic analysis system is being used, the inverse assembler generates a “SW_ADDR” field. This field is the Software Address generated by the inverse assembler.

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

Displaying Data with the B4620B Source Correlation Tool Set

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

Source Correlation Tool Set Data



Display Filtering

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

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

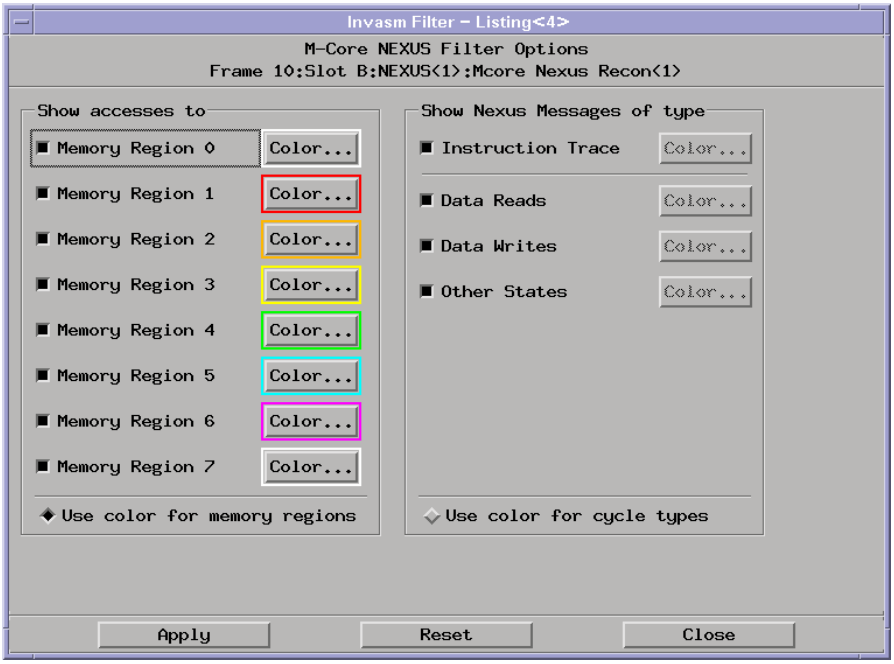
This dialog allows faster analysis in two ways. First, you can filter unneeded information out of the display. For example, suppressing idle states will show only states in which a transaction was completed.

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

16700-series logic analysis systems provide one additional feature for analyzing data. Instead of (or in addition to) showing or suppressing states, the selected states can also be shown in color.

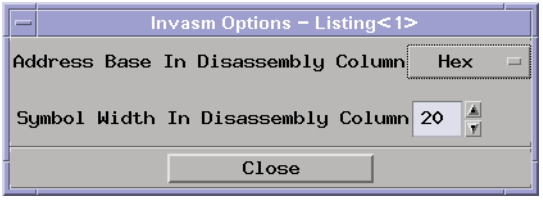
Color can only be used for distinguishing either memory region accesses or cycle types, but not both at the same time.

Inverse Assembler Filter Dialog



Options

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



Displaying Source Code

The B4620B Source Correlation Tool Set lets you:

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

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

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

See Also

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

Requirements for source correlation

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

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

- Proper probing and inverse assembly.

  All the information needed to reconstruct the complete address bus of the target system must be acquired by the logic analyzer. When the target system is properly connected to the logic analyzer, the E8136A inverse assembler meets this requirement.

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

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

- Object file symbols.

  The source correlation tool set requires that symbols be loaded into the logic analyzer (refer to page 50, in this chapter).

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

- Access to source code files.

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

Inverse assembler generated PC (software address) label

In the 16700-series logic analysis system, the M•CORE inverse assembler generates a "PC" label. The PC label is displayed as another column in the Listing tool. This label is also known as the Software Address generated by the inverse assembler.

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

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

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

Access to source code files

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

Source file search path

Verify that the correct file search paths for the source code have been entered into the source correlation tool set. The 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 logic analyzer networking must be compatible with the user's network environment. Agilent 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.

8

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

Format

```
[SECTIONS]
 section_name  start..end  attribute

 [FUNCTIONS]
 func_name  start..end

 [VARIABLES]
 var_name   start [size]
 var_name   start..end

 [SOURCE LINES]
 File: file_name
 line#  address

 [START ADDRESS]
 address

 #Comments
```

If no record header is specified, [VARIABLES] is assumed. Lines without a preceding header are assumed to be symbol definitions in one of the VARIABLES formats.

Example

This is an example GPA file that contains several different kinds of records:

```
[SECTIONS]
 prog      00001000..0000101F
 data      40002000..40009FFF
 common    FFFF0000..FFFF1000

 [FUNCTIONS]
 main      00001000..00001009
 test      00001010..0000101F

 [VARIABLES]
 total     40002000  4
 value     40008000  4

[SOURCE LINES]
 File: main.c
 10        00001000
 11        00001002
 14        0000100A
 22        0000101E

 File: test.c
  5        00001010
  7        00001012
 11        0000101A
```

SECTIONS

Format

```
[SECTIONS]
 section_name  start..end  attribute
```

Use SECTIONS to define symbols for regions of memory, such as sections, segments, or classes.

`section_name` A symbol representing the name of the section.

`start` The first address of the section, in hexadecimal.

`end` The last address of the section, in hexadecimal.

`attribute` This is optional, and may be one of the following:

NORMAL (default)—The section is a normal, relocatable section, such as code or data.

NONRELOC—The section contains variables or code that cannot be relocated; this is an absolute segment.

> **Enable Section Relocation**
>
> To enable section relocation, section definitions must appear before any other definitions in the file.

Example

```
[SECTIONS]
 prog           00001000..00001FFF
 data           00002000..00003FFF
 display_io     00008000..0000801F  NONRELOC
```

If you use section definitions in a GPA symbol file, any subsequent function or variable definitions must be within the address ranges of one of the defined sections. Functions and variables that are not within the range are ignored.

FUNCTIONS

Format

```
[FUNCTIONS]
 func_name  start..end
```

Use FUNCTIONS to define symbols for program functions, procedures, or subroutines.

`func_name` A symbol representing the function name.

`start` The first address of the function, in hexadecimal.

`end` The last address of the function, in hexadecimal.

Example

```
[FUNCTIONS]
 main     00001000..00001009
 test     00001010..0000101F
```

VARIABLES

Format

```
[VARIABLES]
 var_name   start [size]
 var_name   start..end
```

You can specify symbols for variables either by using the address of the variable, the address and the size of the variable, or a range of addresses occupied by the variable. If you specify only the address of a variable, the size is assumed to be one byte.

`var_name` A symbol representing the variable name.

`start` The first address of the variable, in hexadecimal.

`end` The last address of the variable, in hexadecimal.

`size` This is optional, and indicates the size of the variable, in bytes, in decimal.

Example

```
[VARIABLES]
 subtotal     40002000   4
 total        40002004   4
 data_array   40003000..4000302F
 status_char  40002345
```

SOURCE LINES

Format

```
[SOURCE LINES]
 File: file_name
 line#  address
```

Use SOURCE LINES to associate addresses with lines in your source files.

`file_name` The name of a file.

`line#` The number of a line in the file, in decimal.

`address` The address of the source line, in hexadecimal.

Example

```
[SOURCE LINES]
 File: main.c
 10       00001000
 11       00001002
 14       0000100A
 22       0000101E
```

START ADDRESS

Format

```
[START ADDRESS]
 address
```

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

Example

```
[START ADDRESS]
 00001000
```

Comments

Format

```
#comment text
```

Use the # character to include comments in a file. Any text following the # character is ignored. You can put comments on a line alone or on the same line following a symbol entry.

Example

```
 #This is a comment.
```

9

Troubleshooting the Inverse Assembler

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. Otherwise, you may damage circuitry in the analyzer or target system.

Logic Analyzer Problems

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

Intermittent data errors

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

- ❏ Remove and re-seat all cables and probes, ensuring that there are no bent pins or poor probe connections.
- ❏ Adjust the threshold level of the data pod to match the logic levels in the system under test.
- ❏ Use an oscilloscope to check the signal integrity of the data lines.

  Clock signals for the state analyzer must meet particular pulse shape and timing requirements. Data inputs for the analyzer must meet pulse shape and setup and hold time requirements.

See Also See “Capacitive loading” on page 98 for information on other sources of intermittent data errors.

Unwanted triggers

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

- ❏ Add the prefetch queue or pipeline depth to the trigger address to avoid this problem.

  The logic analyzer captures prefetches, even if they are not executed. When you are specifying a trigger condition or a storage qualification that follows an instruction that may cause branching, an unused prefetch may generate an unwanted trigger.

No activity on activity indicators

- ❏ Check for loose cables or board connections.
- ❏ Check for bent or damaged pins on the connectors.

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.

Analyzer won't power up

If logic analyzer power is cycled when the logic analyzer is connected to a target system or emulation probe that remains powered up, the logic analyzer may not be able to power up. Some logic analyzers are inhibited from powering up when they are connected to a target system or emulation probe that is already powered up.

- ❏ Remove power from the target system, then disconnect all logic analyzer cabling from the target system. This will allow the logic analyzer to power up. Reconnect logic analyzer cabling after power up.

Target System Problems

This section lists problems that you might encounter with the target system.

Target system will not boot up

If the target system will not boot up after connecting the logic analyzer, the microprocessor (if socketed) or the 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 analysis probe and target system.

a Power up the analyzer.

b Power up the target system.

❏ Verify that the microprocessor and the cables are securely inserted into their respective sockets.

❏ Verify that the logic analyzer cables are in the proper sockets of the target system and are firmly inserted.

Erratic trace measurements

- ❏ 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 probe connected.

  See “Capacitive loading” in this chapter. While logic analyzer loading is slight, pin protectors, extenders, and adapters may increase it to unacceptable levels. If the target system design has close timing margins, such loading may cause incorrect processor functioning and give erratic trace results.

- ❏ Ensure that you have sufficient cooling for the microprocessor.

  Ensure that you have ambient temperature conditions and air flow that meet or exceed the requirements of the microprocessor manufacturer.

Capacitive loading

Excessive capacitive loading can degrade signals, resulting in incorrect capture by the analysis probe interface, or system lockup in the microprocessor. All interfaces add additional capacitive loading, as can custom probe fixtures you design for your application.

Careful layout of your target system can minimize loading problems and result in better margins for your design. This is especially important for systems that are running at frequencies greater than 50 MHz.

- ❏ Remove as many pin protectors, extenders, and adapters as possible.

Inverse Assembler Problems

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

When you obtain incorrect inverse assembly results, it may be unclear whether the problem is in the connectors or in your target system. If you follow the suggestions in this section to ensure that you are using 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 the correct processor is selected in the processor options preferences menu. (This is not applicable to the M•CORE with Nexus3 inverse assembler.)
- ❏ 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 connector numbers. Target systems must supply address (ADDR), data (DATA), and status (STAT) information to the analyzer in a predefined order. The cable connections are often altered to support that need. Thus, one target system might require that you connect cable 2 to analyzer pod 2, while another will require you to connect cable 5 to analyzer pod 2. See Chapter 3 for connection information.
- ❏ Check the activity indicators for status lines locked in a high or low state.
- ❏ Verify that the STAT, DATA, and ADDR format labels have not been modified from their default values.

  These labels must remain as they are configured by the configuration file. Do

not change the names of these labels or the bit assignments within the labels. Some analysis probes also require other data labels. See“Configuring the 16700-series Logic Analyzer” on page 43 for more information.

- ❏ Verify that memory managers have been disabled (if present).

  In most cases, if the memory managers remain enabled you should still get inverse assembly. It may be incorrect because the logical address may not map to the physical address.

- ❏ Verify that the preferences are set correctly if you have not disabled the cache (not applicable to certain processor types).

  To determine if a cache is on or off, examine the most significant bit of the ICCST register (for the instruction cache) or the DCCST register (for data cache). If this bit is 1, the cache is on; if the bit is 0, the cache is off.

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

- ❏ Verify that the endian selection is correct (if present) in the processor options preferences menu.

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 Also

See “Installing and loading software” on page 31.

See “Loading Configuration and Inverse Assembler Files from the System Hard Disk” on page 44.

Intermodule Measurement Problems

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

An event wasn't captured by one of the modules

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

❏ Adjust the skew in the Intermodule menu.

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

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

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

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 16700-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.

See "To install the software from CD-ROM" on page 32 for details.

"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 analysis probe interface configuration files.

See Also

See "Loading Configuration and Inverse Assembler Files from the System Hard Disk" on page 44 for details on loading configuration files.

"Selected File is Incompatible"

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

"Slow or Missing Clock"

- ❏ This error message might occur if the logic analyzer cards are not firmly seated in the logic analysis system frame. Ensure that the cards are firmly seated.
- ❏ This error might occur if the target system is not running properly. Ensure that the target system is on and operating properly.
- ❏ If the error message persists, check that the logic analyzer pods are connected to the proper connectors on the analysis probe interface. See Chapter 3 to determine the proper connections.

"Time from Arm Greater Than 41.93 ms"

The 16550A state/timing analyzers have a counter to keep track of the time from when an analyzer is armed to when it triggers. The width and clock rate of this counter allow it to count for up to 41.93 ms before it overflows. Once the counter has overflowed, the system does not have the data it needs to calculate the time between module triggers. The system must know this time to be able to display data from multiple modules on a single screen.

"Waiting for Trigger"

If a trigger pattern is specified, this message indicates that the specified trigger pattern has not occurred. Verify that the triggering pattern is correctly set.

- When analyzing microprocessors that fetch only from word-aligned addresses, ensure that the trigger condition is set to look for an opcode fetch at an address corresponding to a word boundary.

“Searching for sync point”

When the trace reconstruction tool begins decoding the trace data, the system searches for a starting point in which TCODE equals 4, 11, or 12. The message “Searching for sync point” is displayed until the starting point is found. The Nexus HTI board provides a complete address for indirect branches, or for all direct and indirect sync points. The complete address is needed to determine the starting point for trace reconstruction.

"Symbol File Not Loaded."

This message is displayed when the symbol file is not loaded in the logic analyzer's setup menu. The symbol file must be in ELF format. See “An ELF file is required for inverse assembly/trace reconstruction.” on page 49 and “To load object file symbols” on page 51.

"Error. Did not end with DIRECT branch."

This message is displayed when the inverse assembler expects the last sequential instruction to be a direct branch and it is not. An early version of the EBDI (the predecessor to the HTI board) incorrectly computes the addresses in some cases, which leads to this error. Double click on the M•CORE Nexus icon in the Workspace and click on "Execute" to make sure that the latest symbol file has been read.

"Error. Indirect branch in mid-stream..."

The inverse assembler decodes a series of sequential instructions based on the COUNT signal provided by the Nexus bus. The “Error. Indirect branch in mid-stream...” message is displayed when an indirect branch opcode occurs when a sequential instruction is expected. An indirect branch instruction is expected as the last instruction in a series. Double click on the M•CORE Nexus icon in the Workspace and click on "Execute" to make sure that the latest symbol file has been read by the reconstruction tool.

10

Hardware Reference

Rainbow operating characteristics

The following operating characteristics are not specifications, but are typical operating characteristics for the E8136A Inverse Assembler.

Operating Characteristics	
Microprocessor Compatibility	M•CORE Rainbow with Nexus3 Port
Microprocessor Bus Speed	Microprocessor bus speed depends on the logic analyzer card used. See logic analyzer card specifications for details.
Signal Line Loading	Typically 100 kΩ plus 10 pF.
Setup/Hold Requirement	For all signals, the logic analyzers require a minimum combined setup/hold window. For 16710/11/12A logic analyzers, the combined window must be at least 4.0 ns. For 16715/16/17/18/19/50/51/52A logic analyzers the combined window must be at least 2.5 ns (1.25 ns using eye finder).

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."

Emulation Solution A set of tools for debugging your target system. A solution includes probing, inverse assembly, the B4620B Source Correlation Tool Set, and an emulation module.

Emulator An emulation module or an emulation probe.

Extender A part whose only function is to provide connections from one location to another. One or more extenders might be stacked to raise a probe above a target microprocessor to avoid mechanical contact with other components installed close to the target microprocessor. Sometimes called a "connector board."

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

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

General-Purpose Flexible Adapter A cable assembly that connects the signals from an elastomeric probe adapter to an analysis probe. Normally, a male-to-male header or transition board makes the connections from the general-purpose flexible adapter to the analysis probe.

High-Density Adapter Cable A cable assembly that delivers signals from an analysis probe hardware interface to the logic analyzer pod cables. A high-density adapter cable has a single *MICTOR connector* that is installed into the analysis probe, and two cables that are connected to corresponding odd and even logic analyzer pod cables.

High-Density Termination Adapter Cable Same as a High-Density Adapter Cable, except it has a termination in the *MICTOR connector*.

In Background, In Monitor See *Background Debug Monitor*.

Inverse Assembler Software that displays captured bus activity as assembly language mnemonics. In addition, inverse assemblers may show execution history or decode control busses.

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

Jumper Moveable direct electrical connection between two points.

JTAG (OnCE) port See *debug port*.

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

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

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

Male-to-male Header A board assembly that makes point-to-point connections between the female pins of a flexible adapter or transition board and the female pins of an analysis probe.

MICTOR Connector A high-density matched 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 Set, and (optionally) an emulation module.

Stand-Alone Logic Analyzer A standalone logic analyzer has a predefined set of hardware components which provide a specific set of capabilities. A standalone logic analyzer differs from a mainframe logic analyzer in that it does not offer card slots for installation of additional capabilities, and its specifications are not modified based upon selection from a set of optional hardware boards that may be installed within its frame.

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

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

Symbol Symbols represent patterns and ranges of values found on labeled sets of bits. Two kinds of symbols are available:
1) Object file symbols — Symbols from your source code, and symbols generated by your compiler. Object file symbols may represent global variables, functions, labels, and source line numbers.
2) User-defined symbols — Symbols you create.

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

Target Control Port An 8-bit, TTL port on a logic analysis system that you can use to send signals to your target system. It does not function like a pattern generator or emulation module, but more like a remote control for the target's switches.

Target Interface Module A small circuit board which connects the 50-pin cable from an E5901A emulation module or E5900A emulation probe

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

TIM See *Target Interface Module*.

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

Transition Board A board assembly that obtains signals connected to one side and rearranges them in a different order for delivery at the other side of the board.

Trigger Specification A set of conditions that must be true before the instrument triggers. See the printed or online documentation of your logic analyzer for details.

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

Index

Index

Index

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in subparagraph (C) (1) (ii) of the Rights in Technical Data and Computer Software Clause in DFARS 252.227-7013.
Agilent Technologies,
395 Page Mill Road,
Palo Alto, CA 94303-0870 U.S.A.
Rights for non-DOD U.S. Government Departments and Agencies are set forth in FAR 52.227-19 (c) (1,2).

Document Warranty

The information contained in this document is subject to change without notice.

Agilent Technologies makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

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

Safety

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

Warning

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

Safety Symbols

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

Hazardous voltage symbol.

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

WARNING

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

CAUTION

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

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

Product Warranty

This Agilent Technologies product has a warranty against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Agilent Technologies will, at its option, either repair or replace products that prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Agilent Technologies.

For products returned to Agilent Technologies for warranty service, the Buyer shall prepay shipping charges to Agilent Technologies and Agilent Technologies shall pay shipping charges to return the product to the Buyer. However, the Buyer shall pay all shipping charges, duties, and taxes for products returned to Agilent Technologies from another country.

Agilent Technologies warrants that its software and firmware designated by Agilent Technologies for use with an instrument will execute its programming instructions when properly installed on that instrument. Agilent Technologies does not warrant that the operation of the instrument software, or firmware will be uninterrupted or error free.

Limitation of Warranty

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by the Buyer, Buyer- supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

No other warranty is expressed or implied. Agilent Technologies specifically disclaims the implied warranties of merchantability or fitness for a particular purpose.

Exclusive Remedies

The remedies provided herein are the buyer's sole and exclusive remedies. Agilent Technologies shall not be liable for any direct, indirect, special, incidental, or consequential damages, whether based on contract, tort, or any other legal theory.

Assistance

Product maintenance agreements and other customer assistance agreements are available for Agilent Technologies products. For any assistance, contact your nearest Agilent Technologies Sales Office.

Certification

Agilent Technologies Company certifies that this product met its published specifications at the time of shipment from the factory. Agilent Technologies further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology, to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

About this edition

This is the *Logic Analysis Support for the Motorola M•CORE Rainbow with Nexus3 Port User's Guide*.

Publication number
E8136-97000, July 2001
Printed in USA.

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