

---

# Help Volume

© 1998-2001 Agilent Technologies. All rights reserved.

---

## Display: Listing Display Tool

---

## Using the Listing Tool



The Listing tool displays state data in a listing format in the same order it was captured and placed into memory. Use the Listing tool to find and view data patterns and sequence of events.

- “Inverse Assemble State Listings” on page 45
- “Go to an Exact State” on page 41
- “Go to an Exact Pattern” on page 42
- Searching for a Single Data Value or Pattern (see the *Markers* help volume)
- Searching for Complex Patterns and Ranges (see the *Markers* help volume)
- Tracking a Sequence of Events - Grouping Markers (see the *Markers* help volume)
- “Loading and Saving Listing Configurations” on page 40
- “Printing the Listing to a File” on page 47
- “Printing the Listing Window” on page 55

### See Also

Working with Markers (see the *Markers* help volume)

“Working with Labels” on page 10

“Working with Bookmarks” on page 50

“Working with Inverse Assemblers” on page 17

“Using the Source Viewer” on page 24

“Seeing Measurement Results - Popup on Run” on page 85

“Changing the Trigger Reference” on page 49

“Selecting the Display Reference” on page 54

“Setting the Numeric Base” on page 14

“Turning Tabs On or Off in the Display Window” on page 86

Using the Analysis Tab (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Using the Mixed Signal Tab (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Main System Help (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Glossary of Terms (see page 89)



## Using the Listing Tool

### 1 Using the Listing Tool

|   |    |
|---|----|
| Working with Labels                               | 10 |
| Adjusting the Label Name Width                    | 10 |
| Setting the Label Font Size                       | 11 |
| Inserting, Replacing, or Deleting Labels          | 11 |
| Rearranging the Label Order                       | 12 |
| Adjusting Column Width                            | 12 |
| Setting Column Color                              | 13 |
| Setting the Numeric Base                          | 14 |
| Finding a label                                   | 14 |
| Working with Inverse Assemblers                   | 17 |
| Using Filter Options                              | 17 |
| Setting the Address Base in Disassembly Column    | 18 |
| Setting Symbol Width                              | 19 |
| Choosing an Alignment Starting Point              | 19 |
| Align the Display                                 | 21 |
| Loading and Unloading Inverse Assembler Files     | 22 |
| Using the Source Viewer                           | 24 |
| To open the Source Viewer display                 | 25 |
| Viewing Source Code Associated with Captured Data | 26 |
| Setting Up Triggers Based on Source Code          | 29 |
| Setting Source Viewer Options                     | 35 |
| Loading and Saving Listing Configurations         | 40 |
| Go to an Exact State                              | 41 |
| Go to an Exact Pattern                            | 42 |
| Defining the Search Term                          | 43 |

---

# Contents

|                                      |    |
|--------------------------------------|----|
| Inverse Assemble State Listings      | 45 |
| Printing the Listing to a File       | 47 |
| Changing the Trigger Reference       | 49 |
| Working with Bookmarks               | 50 |
| Placing Temporary Bookmarks          | 50 |
| Goto Temporary Bookmarks             | 51 |
| Placing Permanent Bookmarks          | 51 |
| Goto Permanent Bookmarks             | 52 |
| Selecting the Display Reference      | 54 |
| Printing the Listing Window          | 55 |
| Run/Group Run Function               | 56 |
| Checking Run Status                  | 57 |
| Demand Driven Data                   | 58 |
| Adding and Deleting Tools            | 59 |
| Help - How to Navigate Quickly       | 60 |
| Connecting Tools Together            | 61 |
| Clearing the Workspace               | 62 |
| Repositioning Tools in the Workspace | 63 |
| The Symbols Tab                      | 64 |
| Displaying Data in Symbolic Form     | 65 |

---

# Contents

|  |    |
|--|----|
| Setting Up Object File Symbols                               | 66 |
| To Load Object File Symbols                                  | 66 |
| Relocating Sections of Code                                  | 68 |
| To Delete Object File Symbol Files                           | 69 |
| Symbol File Formats  | 69 |
| Creating ASCII Symbol Files                                  | 70 |
| Creating a readers.ini File                                  | 75 |
| <br>   |    |
| Using Symbols In The Logic Analyzer                          | 78 |
| Using Symbols As Trigger Terms                               | 78 |
| Using Symbols as Search Patterns in Listing Displays         | 79 |
| Using Symbols as Trigger Terms in the Source Viewer          | 79 |
| Using Symbols as Pattern Filter Terms                        | 79 |
| Using Symbols as Ranges in the Software Performance Analyzer | 80 |
| <br>   |    |
| User-Defined Symbols   | 83 |
| To Create User-Defined Symbols                               | 83 |
| To Replace User-Defined Symbols                              | 83 |
| To Delete User-Defined Symbols                               | 84 |
| To Load User-Defined Symbols                                 | 84 |
| <br>   |    |
| Seeing Measurement Results - Popup on Run                    | 85 |
| <br>   |    |
| Turning Tabs On or Off in the Display Window                 | 86 |
| <br>   |    |
| Including Comments on Screen Prints                          | 87 |
| <br>   |    |
| Editing Colors   | 88 |

## **Glossary**

## **Index**

---

# Contents



---

Using the Listing Tool

## Working with Labels

---

**NOTE:**

---

Labels are created in the Format window of the instrument tool you are using. If you need to create new labels or modify existing label names, or change the polarity of a label, go to the *Format* window.

The following label operations can be performed in the listing display:

- “Inserting, Replacing, or Deleting Labels” on page 11
- “Setting the Label Font Size” on page 11
- “Adjusting the Label Name Width” on page 10
- “Rearranging the Label Order” on page 12
- “Adjusting Column Width” on page 12
- “Setting Column Color” on page 13
- “Setting the Numeric Base” on page 14
- “Finding a label” on page 14

---

## Adjusting the Label Name Width

The label name can be set to the following types:

- *Short* - only the name of the label name.
  - *Unique* - the name of the Analyzer and label. If there are duplicate label names, then the Analyzer name is added to the front of the label name.
  - *Full* - the name of the Frame, Slot, Analyzer, and label.
1. From the tool window, select the desired label, then select *Change Attributes*.
  2. From the Change attributes dialog, select the *Label format* field and select the desired label name type.
  3. Select *OK*.

---

## Setting the Label Font Size

Font settings are global. The font size you select is applied to all text in the display area.

1. From the tool's menu bar select *Options*, then select *Font*.
2. Select a font size from the list.

|        |                        |             |                  |
|--------|------------------------|-------------|------------------|
| Small  | StateNumber<br>Decimal | Lab1<br>Hex | Time<br>Relative |
| Medium | StateNumber<br>Decimal | Lab1<br>Hex | Time<br>Relative |
| Normal | StateNumber<br>Decimal | Lab1<br>Hex | Time<br>Relative |
| Bold   | StateNumber<br>Decimal | Lab1<br>Hex | Time<br>Relative |

### Example of Font Sizes

---

## Inserting, Replacing, or Deleting Labels

---

**NOTE:**

Labels that appear in the Label List are predefined in the *Format* window of the instrument tool you are using.

1. From the Listing display, select a label, then select a label function from the list.
2. In the Label Selection dialog, select a label from the List. If the selection list is long, use the scroll.
3. Select *Ok*. Use *Apply* if you want to repeat the operation without closing the dialog.

**Working with Labels**

You can also insert or delete labels through *Edit* in the display menu bar.

**Rearranging the Label Order**

1. *Point* to a label, then press and hold.
2. Drag the highlighted label to its new location, then release.

| StateNumber | Address | Time     |
|-------------|---------|----------|
| Decimal     | Hex     | Relative |
| -39         | 00      | 4,000 ns |
| -38         | 00      | 4,000 ns |
| -37         | 00      | 4,000 ns |
| -36         | 00      | 4,000 ns |

**Adjusting Column Width**

| State Number | Data_A | Time       |
|--------------|--------|------------|
| Decimal      | Hex    | Relative   |
| -8           | 0093   | -32,000 ns |
| -7           | 0093   | -28,000 ns |
| -6           | 0093   | -24,000 ns |
| -5           | 0093   | -20,000 ns |
| -4           | 0093   | -16,000 ns |
| -3           | 0093   | -12,000 ns |
| -2           | 0093   | -8,000 ns  |
| -1           | 0093   | -4,000 ns  |

1. *Point* to the right or left border line of the label box, then press and hold.

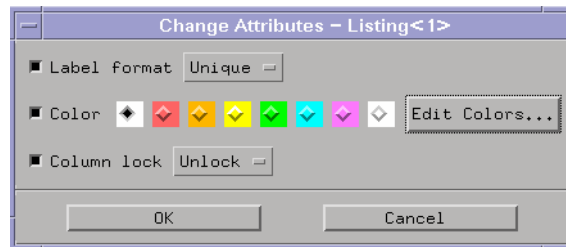
2. Drag the box edge to the desired width, then release.

You can also lock the column (see page 13).

---

## Setting Column Color

1. Select the desired label, then select *Change attributes*.
2. Select *Color* to enable, then select the desired color.
3. Select *Ok*.



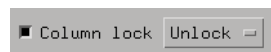
If you want to change the default colors for data columns, markers, and the window background color, use *Edit Colors...* (see page 88).

### See Also

“Locking the Column” on page 13

## Locking the Column

When a column is locked, it is highlighted and moved to the left side of the display. When locked, it cannot be deleted or moved.

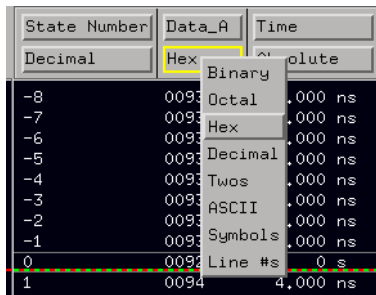


### Column Locking

1. In the *Change attributes* Dialog, select *Column lock*.
2. Select *Lock*, then select *Ok*.

---

## Setting the Numeric Base



1. Select the desired label's base field.
2. Select the desired base type.

### Symbols and Line #s

Software developers use *symbols* and *line numbers* (Line #s) to more easily find or view their code.

### Time Mode

In addition to the numeric base, when the *Time* label is displayed, you can set the column to display either relative, absolute time, or absolute picoseconds time.

### Polarity

Note that the polarity of a label, which is set in the *Format* window, affects the values displayed for the label. For example, the hexadecimal value "0000" will change to "FFFF" if the polarity of the label is changed from + to -.

---

## Finding a label

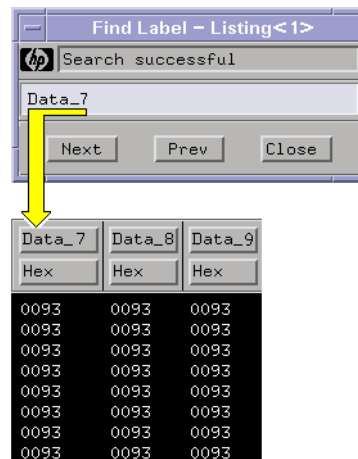
Access the *Find Label* feature under the *Edit* pick in the Listing tool menu bar. The *Find Label* feature locates specified labels, then displays them in the left-most column of the displayed listing.

The search functionality is similar to a text based keyword search. As you type the label name, the list of labels is rolled, exposing the label whose text characters, left to right, match the label name being typed.

You can search for a complete label name as shown below in example 1, or type just the base name and use the *Next* or *Prev* fields to roll through the list as in example 2.

### Example 1

You have a list of labels named *Data1* through *Data15*. Type the complete name *Data7* in the text entry field. The label named *Data7* appears on the left-most side of the displayed labels.



### Example 2

You have a list of labels named *Data1* through *Data15*. Type only the base name *Data* in the text entry field, then use the *Next* or *Prev* fields to roll the list of labels.

Chapter 1: Using the Listing Tool  
**Working with Labels**



| Data_1 | ... | Data_15 |
|--------|-----|---------|
| Hex    |     | Hex     |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |
| 0093   |     | 0093    |



## Working with Inverse Assemblers

Inverse assembly (IA) allows you to view state data as pseudocode mnemonics that resemble the actual program your system is running. To view inverse assembled data, you need a State Analyzer tool, a Listing tool, and the inverse assembler and configuration files for your microprocessor or bus.

You perform the following operations when working with inverse assemblers:

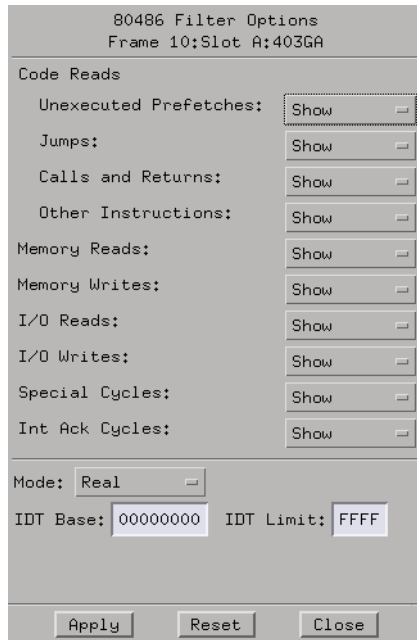
- Load a configuration file (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume) or set up a configuration within the analyzer
- “Align the Display” on page 21
- “Choosing an Alignment Starting Point” on page 19
- “Setting the Address Base in Disassembly Column” on page 18
- “Setting Symbol Width” on page 19
- “Using Filter Options” on page 17
- “Loading and Unloading Inverse Assembler Files” on page 22

---

## Using Filter Options

Some inverse assemblers have filter options. If filter options are available, there is a *Filter* selection under *Invasm* in the listing menu bar. Use filter options to select the information you want to show.

1. From the Listing menu bar, select *Invasm* then select *Filter*.
2. Set the desired filter choices.
3. Select *Apply* to use your choices or *Reset* to restore the former setup.
4. Select *Close*.



### **Mode and IDT Start/Size**

Some filter options include the Interrupt Descriptor Table (IDT) and the Mode selections. If you suspect the inverse assembler is disassembling improperly, make sure these settings match your target system. Refer to your analysis probe manual for more information. In most cases, the inverse assembler can determine the system settings, and will operate properly regardless of the settings entered here.

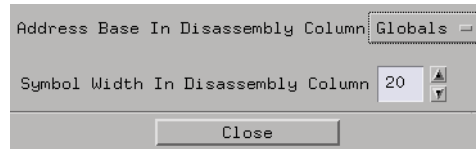
---

## **Setting the Address Base in Disassembly Column**

Changing the address base in inverse assembled listings allows you to see addresses expressed in hexadecimal values, global symbols, or source file line numbers.

1. From the Listing menu bar, select *Invasm* then select *Options*.

2. Select the *Address Base In Disassembly Column*, and select either *Hex*, *Globals*, or *Line #s*.
3. Select *Close*.

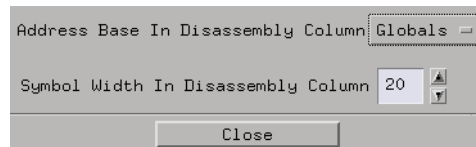


---

## Setting Symbol Width

When you specify the symbol width, you set the maximum number of characters to be seen in symbolic names that appear in inverse assembled listings. For example, a symbol might appear as the address of a jump instruction. This sets the width of that address symbol. Set a width large enough to allow easy interpretation of each symbol.

1. From the Listing menu bar, select *Invasm* then select *Options*.
2. Enter the symbol width.
3. Select *Close*.



---

### NOTE:

To widen or narrow a column in the Listing display, drag the width of the column in the column heading.

---

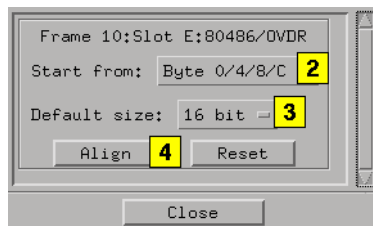
## Choosing an Alignment Starting Point

When you select a specific starting point (byte, word, etc.), the inverse assembler knows where in the first displayed line to begin inverse

---

assembly. Each time you assign a new alignment starting point, the IA re-aligns using the new starting point.

1. From the Listing menu bar, select *Invasm* then select *Align*.
2. From the Alignment dialog, select the *Start from* field, and select the desired starting point (see page 20) (byte, word, etc.).
3. If appropriate, toggle the *Default size* to either 16 bit or 32 bit. An equal sign (=) is placed in front of 32-bit instructions in the listing.
4. Select *Align* to start inverse assembly or *Reset* to restore the former display alignment.
5. Select *Apply*.

**See Also**

Refer to your analysis probe manual for more information.

**Byte Location**

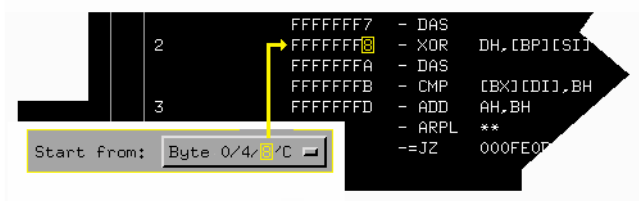
---

**NOTE:**

---

This is a 486 example

If you have an instruction that is displayed in multiple lines, you have to designate which byte the IA starts inverse assembling from. You have four choices, each containing four byte values. All total, you have 16 choices (4x4=16). As the example below shows, the first of three bytes in the instruction starts with an eight (8). You would use the Byte choice of 0/4/8/C because it contains eight (8).



**See Also** Refer to your analysis probe manual for more information.

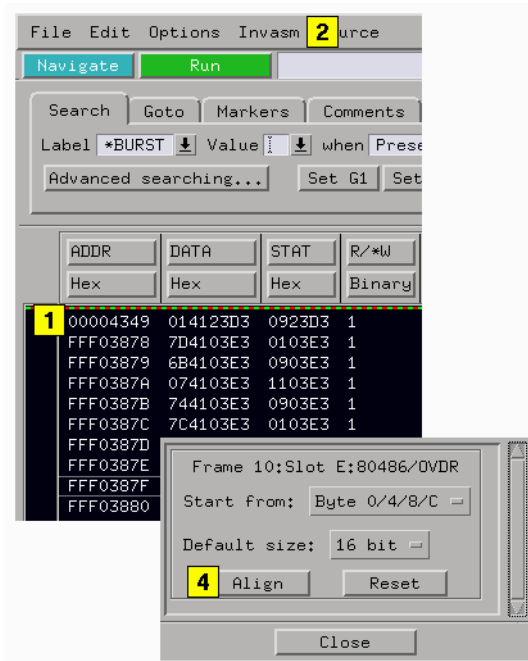
---

## Align the Display

Some inverse assemblers have alignment options. If alignment options are available, there is an *Align* selection under *Invasm* in the Listing menu bar.

Data is inverse assembled from the top line down. Any data before the top line of the display is left unchanged. Rolling the display up inversely assembles the lines as they appear on the bottom of the display. If you jump to another area in the display which is not inverse assembled, you will need to re-align by repeating steps 1 through 5 below. Once you inverse assemble a block of memory, the analyzer keeps it in the inverse assembled condition.

1. Scroll the listing so the first line you want inverse assembled is located at the top of the display.
2. From the Listing menu bar, select *Invasm* then select *Align*.
3. In the Alignment dialog, select any appropriate options.
4. Select *Align* to restart inverse assembly, or *Reset* to restore the former alignment of the listing.
5. Select *Close*.



---

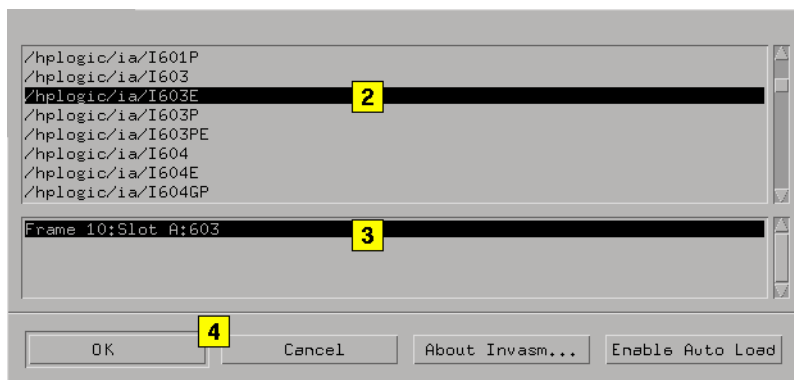
## Loading and Unloading Inverse Assembler Files

The *Load* operation loads IA files located in the logic analysis system, and adds inverse assembler information to the trace list.

1. From the Listing menu bar, select *File* then select *Load Invasm*, or select *Invasm* and then *Load*.
2. From the Load Invasm dialog, select the inverse assembly file.
3. If you have more than one analyzer module in your configuration, select the desired measurement module.
4. Select *OK*.

The *Unload* operation removes the inverse assembler information. If you have a trace list on screen when you select *Invasm -> Unload*,

your listing display will show a list of the values of the bits on the buses.



### Enable Auto Load

In some cases when loading IA files, the analyzer asks you to turn the Auto Load off. After such an event, the *Enable Auto Load* field allows you to reset the Auto Load function back to its default of ON. In most cases, you are loading IA configurations from the *Load Configuration* function, so resetting the Auto Load will not be necessary.

### About Invasm

Use the *About Invasm* for information on the IA software package, labels used, and any included components.

## Using the Source Viewer

The Source Viewer lets you:

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

Before you can use the Source Viewer, you must:

1. Obtain a license for the source correlation tool set. (see page 25)
2. Load symbols from your program code's object module file (see page 66).
3. Give the logic analysis system access to your program's source files (either by NFS mounting or by copying source files to the logic analyzer).
4. Open the Source Viewer display. (see page 25)

### **Viewing Source Code Associated with Captured Data**

- “To step through captured data by source lines” on page 26
- “To go to Listing reference points (Trigger, G1, G2)” on page 27
- “To go to captured data associated with a source line” on page 28

### **Setting Up Triggers Based on Source Code**

- “To trigger after, about, or before a source line” on page 29
- “To trace everything” on page 30
- “To trace until stop” on page 31
- “To trace about a variable, function, or line number” on page 32
- “To modify the trace setup” on page 32
- “To enable or disable emulator break on trigger” on page 33
- “If there are trigger setup problems” on page 34

### **Performing Other Source Viewer Tasks**

- “To browse source files” on page 35
- “To search for text in the source code” on page 35
- “To reload, or load different, symbols” on page 36
- “To edit the directory search list” on page 36

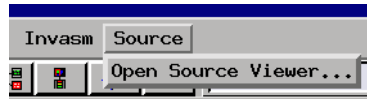


- “To specify the "Goto In Listing" alignment” on page 38
- “To change Source Viewer display options” on page 38
- “To display Source Viewer information” on page 39

---

## To open the Source Viewer display

You can open the Source Viewer from a logic analyzer icon or from the Listing display tool.



### See Also

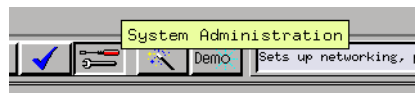
“To obtain a license for the source correlation tool set” on page 25

## To obtain a license for the source correlation tool set

The B4620B source correlation tool set is an add-on product to the Agilent Technologies 16700-series logic analysis system.

In order to use the Source Viewer display, you must obtain a license for the B4620B source correlation tool set.

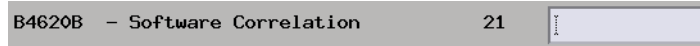
1. Follow the instructions on the Entitlement Certificate to obtain a password for the B4620B Source Correlation Tool Set. The password will be calculated based on your particular system ID and the information on the Entitlement Certificate.
2. In the main logic analysis system window, select the *System Admin* button.



3. In the System Administration Tools dialog, select the *Licensing...* button.



4. Enter the password.



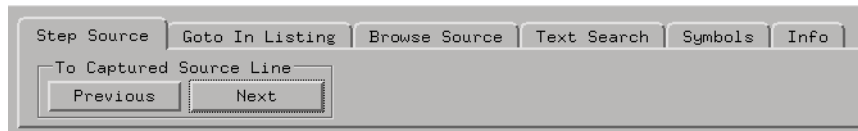
---

## Viewing Source Code Associated with Captured Data

- “To step through captured data by source lines” on page 26
- “To go to Listing reference points (Trigger, G1, G2)” on page 27
- “To go to captured data associated with a source line” on page 28

### To step through captured data by source lines

1. Select the Step Source tab.



2. Select *Previous* to go to the previous state in the captured data that is associated with a source line.

Or, select *Next* to go to the next state in the captured data that is associated with a source line.

```
69      update_sys.c: 62      lis      r3 0000
73      update_sys.c: 62      addi     r3 r3 4088
77      update_sys.c: 62      bl       update_s:get_targets
81      update_sys.c: 98      lis      r12 0000
85      update_sys.c: 98      li       r0 00000001
89      update_sys.c: 98      stb     r0 41C3(r12)
93      q.:ME_get_targets      write 01
```

```
Displayed File: /hplogic/demo/860_demo_board/source/update_sys.c
94 void
95 get_targets(INT_8 *temperature)
96 {
97
98     ME_get_targets = 1;
99
100    /* Ramp the temperature targets up and down */
101
102    if (*temperature == temp_target)
```

The source line associated with the captured data is highlighted.

## To go to Listing reference points (Trigger, G1, G2)

1. Select the Goto In Listing tab.



2. Select *Current* to highlight the source line associated with the captured data being viewed in the Listing display tool. *Current* is used to orient the Source Viewer right after it is opened or after browsing a source file.

Or, select *Trigger*, *G1 Marker*, or *G2 Marker* to go to that reference point in the Listing display tool.

## Chapter 1: Using the Listing Tool

### Using the Source Viewer

|       |                   |       |                      |          |
|-------|-------------------|-------|----------------------|----------|
| 49992 | :proc_spec.c: 101 | li    | r3 00000001          | FFF04170 |
| 49996 | :proc_spec.c: 101 | addi  | r4 r1 0008           | FFF04174 |
| 50000 | :proc_spec.c: 101 | bl    | lcd:lcd_write_string | FFF04178 |
| 50004 | /q.elf:lcd.c: 117 | mfscr | r0 lr                | FFF03060 |
| 50008 | /q.elf:lcd.c: 117 | mr    | r11 r1               | FFF03064 |
| 50012 | /q.elf:lcd.c: 117 | stwu  | r1 FFE0(r1)          | FFF03068 |
| 50016 | /q.elf:lcd.c: 117 | bl    | rce/q.elf:;text+4A04 | FFF0306C |

```

Displayed File: /hplgic/demo/860_demo_board/source/lcd.c
113
114 /*****
115 void
116 lcd_write_string(int line, char *str)
117 {
118     /* This routine will write the first 16 characters of str to the LCD */
119     /* At either line #1 or line #2, depending on the value of line */
120     /* The cursor will be set to the beginning of the line, and placed in */
121     /* overwrite mode */
122

```

The source line associated with the reference point is highlighted.

### To go to captured data associated with a source line

1. Select the source line.
2. Choose either the *Goto this line in listing before current state* or *Goto this line in listing after current state* menu item.

```

Displayed File: /hplgic/demo/860_demo_board/source/ecs2.c
139
140 main()
141 {
142     boot_q();
143
144     init_system();
145     proc_spec_init();
146
147     for (;;)
148     {
149         update_system(num_checks);
150         num_checks++;
151         update_display(num_checks);
152         proc_specific();
153     }
154 }
155
156 /*****
157 * FUNCTION: update_disp
158 * PARMS:   counter --
159 * DESCRIPTION:
160 *   clear out the histo
161 *   of operating data (
162 *

```

|       |                   |       |                      |          |
|-------|-------------------|-------|----------------------|----------|
| 45963 | /q.elf:;text+4A70 | mtspr | lr r0                | FFF06A70 |
| 45967 | /q.elf:;text+4A74 | ori   | r1 r11 0000          | FFF06A74 |
| 45971 | /q.elf:;text+4A78 | blr   |                      | FFF06A78 |
| 45975 | q.elf:ecs2.c: 152 | bl    | proc_s:proc_specific | FFF031A8 |
| 45979 | :proc_spec.c: 69  | mfscr | r0 lr                | FFF040CC |
| 45983 | :proc_spec.c: 69  | stw   | r0 0004(r1)          | FFF040D0 |
| 45987 | :proc_spec.c: 69  | stwu  | r1 FFE0(r1)          | FFF040D4 |

The Listing display will search for the captured data associated with the source line.

---

**NOTE:**

The *Goto this line in listing* commands perform a pattern search on the PC or IP 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.

---

**See Also**

“To browse source files” on page 35

“To specify the "Goto In Listing" alignment” on page 38

---

## Setting Up Triggers Based on Source Code

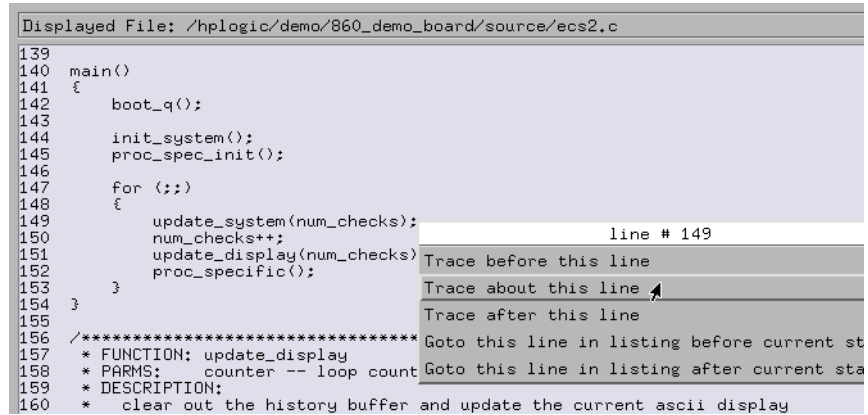
- “To trigger after, about, or before a source line” on page 29
- “To trace everything” on page 30
- “To trace until stop” on page 31
- “To trace about a variable, function, or line number” on page 32
- “To modify the trace setup” on page 32
- “To enable or disable emulator break on trigger” on page 33
- “If there are trigger setup problems” on page 34

### **To trigger after, about, or before a source line**

1. Select the source line you wish to trigger on.
2. Choose either *Trace before this line*, *Trace about this line*, or *Trace after this line* to set up a trace where the trigger point is at the end, middle, or start of the captured and stored data.

## Chapter 1: Using the Listing Tool

### Using the Source Viewer



```
Displayed File: /hplogic/demo/860_demo_board/source/ecs2.c
139
140 main()
141 {
142     boot_q();
143
144     init_system();
145     proc_spec_init();
146
147     for (;;)
148     {
149         update_system(num_checks);
150         num_checks++;
151         update_display(num_checks);
152         proc_specific();
153     }
154 }
155
156 /*****
157 * FUNCTION: update_display
158 * PARMs:   counter -- loop count
159 * DESCRIPTION:
160 *   clear out the history buffer and update the current ascii display
```

line # 149

Trace before this line

Trace about this line

Trace after this line

Goto this line in listing before current sta

Goto this line in listing after current sta

You are notified when the trigger has been set.

3. Select Run to start the measurement.

---

#### NOTE:

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

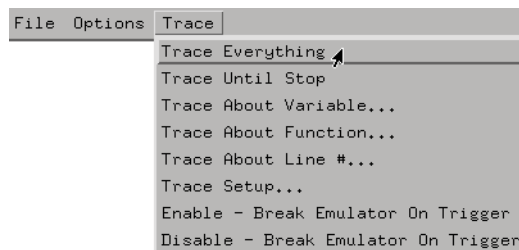
#### See Also

“To modify the trace setup” on page 32

“If there are trigger setup problems” on page 34

### To trace everything

1. Choose the Trace->Trace Everything command.



The logic analyzer will be set up to trigger on any state.

You are notified when the trigger is set.

2. Select Run to start the measurement.

---

**NOTE:**

---

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

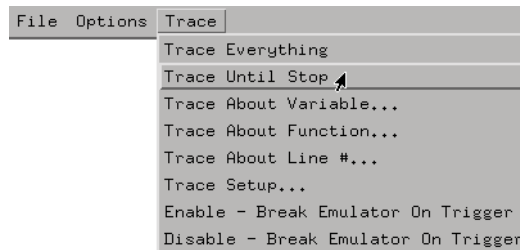
**See Also**

“To modify the trace setup” on page 32

“If there are trigger setup problems” on page 34

### To trace until stop

1. Choose the Trace->Trace Until Stop command.



The logic analyzer will be set up to never trigger.

This command is useful to capture what leads up to a target system halt.

You are notified when the trigger is set.

2. Select Run to start the measurement.
3. When you are ready to view the data that has been captured (for example, when the target system halts), select Stop to stop the measurement.

The Listing display and Source Viewer will show the data captured before the measurement was stopped.

---

**NOTE:**

---

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

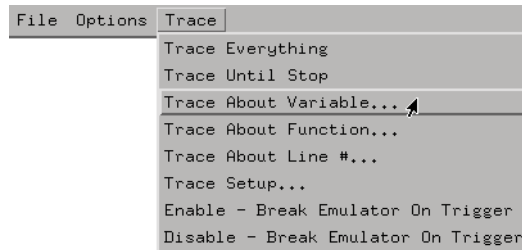
**See Also**

“To modify the trace setup” on page 32

“If there are trigger setup problems” on page 34

**To trace about a variable, function, or line number**

1. Choose the Trace->Trace About Variable..., Trace->Trace About Function..., or Trace->Trace About Line#... command.



2. Select the variable, function, or line number symbol to trigger about, and select OK.

The logic analyzer will be set up to trigger on the specified address.

You are notified when the trigger is set.

3. Select Run to start the measurement.

**NOTE:**

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

**See Also**

“To modify the trace setup” on page 32

“If there are trigger setup problems” on page 34

**To modify the trace setup**

1. Choose the Trace->Trace Setup... command.
2. Make your trace setup changes in the Source Line Trigger window.

You can change the *Trigger Position* to the start, center, or end of the captured data.

You can logically AND other label qualifiers to the ADDR label qualifier to, for example, trace before a write of a particular data value to a variable instead of just tracing about the variable.

You can insert, replace, or delete labels, and you can change how the label



appears in the dialog.

You can use the *Save* and *Recall* buttons to save and restore particular trace setups.

3. Select OK when you are done making trace setup changes.

You are notified when the trigger is set.

4. Select Run to start the measurement.

---

**NOTE:**

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

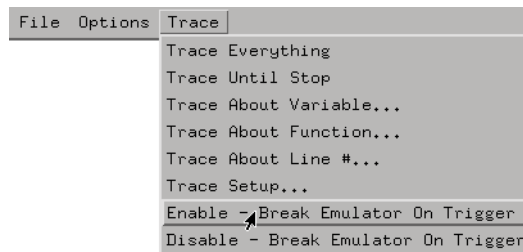
---

**See Also**

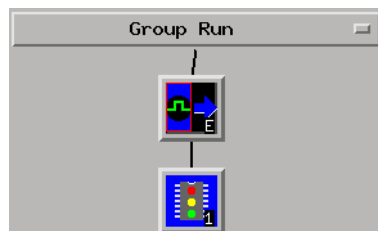
“If there are trigger setup problems” on page 34

### To enable or disable emulator break on trigger

1. Choose the Trace->Enable - Break Emulator On Trigger or Trace->Disable - Break Emulator On Trigger command.



When you enable the emulator break, the Intermodule window is set up so that the emulation module's break input is driven by the logic analyzer's trigger output.



When you disable the emulator break, the Intermodule window is set up so that the emulation module's break input is not driven by the logic analyzer.

You are notified when the trigger is set and the changes to the Intermodule window are made.

2. Select Run to start the measurement.

### **If there are trigger setup problems**

---

**NOTE:**

---

Source Viewer commands that set up triggers only modify the trigger condition. They do not modify the trigger position, storage qualifiers, "else" branch conditions, or other levels in the trigger sequence.

When you set up a trigger using the Source Viewer, it tries to use the logic analyzer's pattern and range resources in order. Trigger setup problems can occur when these resources are being used or aren't available. Common trigger setup error messages are described below.

**Resource in use by other machine**

This message occurs when the logic analyzer is configured as two machines and the Source Viewer wants one of the pattern or range resources used by the other machine.

To work around this problem, you can modify the trigger setup in the other machine so that it uses resources at the end of the available list, or you can turn off the other machine's use of the resource.

**Trigger specification exceeds resources available**

This message typically occurs when you use the Trace->Modify... command to set up trigger specifications that require more pattern or range resources than are available in the logic analyzer.

In this case, you must simplify the trigger setup.

**Unable to communicate with machine**

This message can occur when the Listing and Source Viewer input data comes from the File In tool instead of a logic analyzer.

In this case, you cannot set up triggers using the Source Viewer.

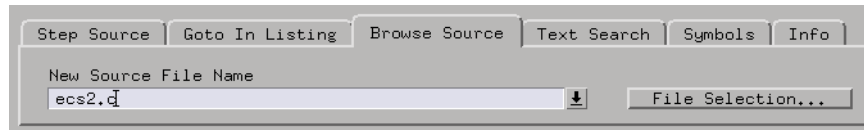
---

## Setting Source Viewer Options

- “To browse source files” on page 35
- “To search for text in the source code” on page 35
- “To reload, or load different, symbols” on page 36
- “To edit the directory search list” on page 36
- “To specify the "Goto In Listing" alignment” on page 38
- “To change Source Viewer display options” on page 38
- “To display Source Viewer information” on page 39

### To browse source files

1. Select the Browse Source tab.



2. Either enter the name of the source file you want to look at, select a source file from the drop-down list, or select the *File Selection...* button to open the file selection dialog.

Browsing source files is sometimes necessary before choosing commands for a particular source line.

### See Also

“To go to captured data associated with a source line” on page 28

“To trigger after, about, or before a source line” on page 29

### To search for text in the source code

1. Select the Text Search tab.



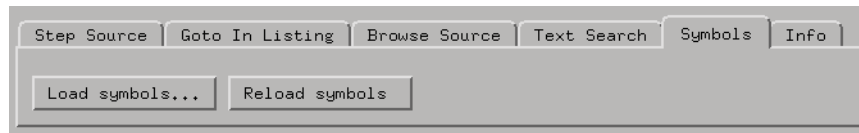
2. In the *Search String* field, enter the string you want to search for.
3. Select the *Search* button to start the search.

**See Also**

“To browse source files” on page 35

**To reload, or load different, symbols**

1. Select the Symbols tab.

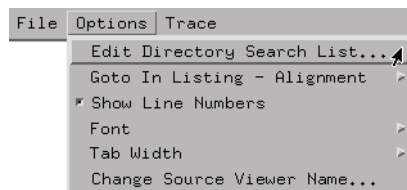


2. Either select the *Load symbols...* button to open the logic analyzer setup window with the Symbols tab selected.

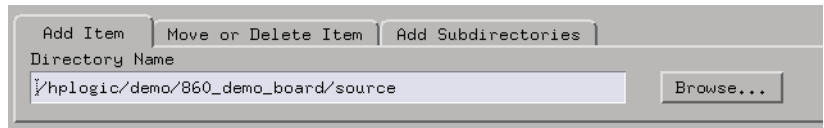
Or, select the *Reload symbols* button to reload the current symbol file. This is typically used after program code has been re-compiled and downloaded to the target system so that symbols match the new code.

**To edit the directory search list**

1. In the Source Viewer, choose the Options->Edit Directory Search List... command.

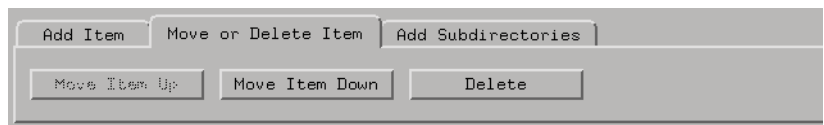


- To add a directory to the list, select the Add Item tab.



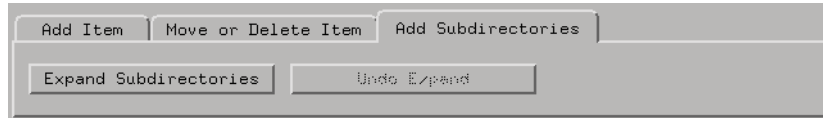
Either enter the directory name (starting with a "/) or select *Browse...* to open the directory selection dialog.

- To change the order of directories in the search list, or to remove a directory from the list, select the Move or Delete Item tab.



Select the directory in the list then select the *Move Item Up*, *Move Item Down*, or *Delete* buttons.

- To add subdirectories to the list, select the Add Subdirectories tab.



The *Expand Subdirectories* button will add all the subdirectories found under the selected directory. (The directories are recursively generated, including symbolically linked subdirectories.) The subdirectories will be listed below the selected directory.

The *Undo Expand* button will remove directories that were previously added by an Expand Subdirectories command. If any other addition or deletion is done from the list after the expand, the *Undo Expand* button will not be accessible. If you merely move items in the list after an *Expand Subdirectories*, the previous expand can be undone.

**What is the directory search list?**

The directory search list is used by parts of the logic analysis system to locate symbol, source, and executable files.

**How is the directory search list used?**

When you type the name of a file, without its full directory path, in dialogs such as the symbol load dialog or in the Source Viewer browse dialog, the directory search list is used to find the file.

**Using the Source Viewer**

Source files referenced by NFS-mounted executable files are searched for under the same NFS mount point as the executable file.

**How is the list of directories built?**

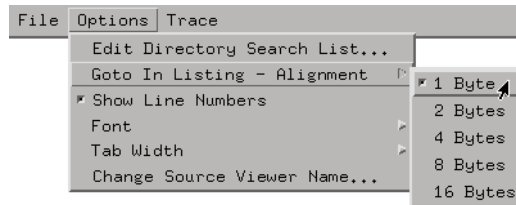
Directories are automatically placed in the directory search list when:

1. NFS directories are mounted.
2. Symbol files (with directory paths) are loaded.
3. Source files (with directory paths) are viewed from the Source Viewer.

Additionally, a "default" source directory (`/logic/source`) is placed in the list upon startup.

**To specify the "Goto In Listing" alignment**

1. Choose the Options->Goto In Listing - Alignment->N Bytes command.



When going to captured data in the Listing display that is associated with a source line, this option specifies whether to search for the exact address of the line number symbol or for the address aligned to some byte boundary.

This option is used when microprocessors fetch blocks of instructions at a time (from block boundary addresses only).

**See Also**

“To go to captured data associated with a source line” on page 28

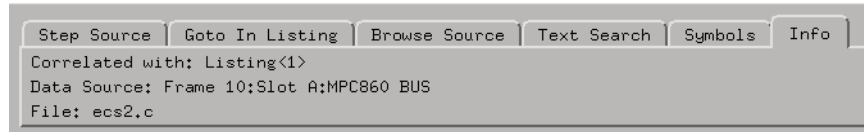
**To change Source Viewer display options**

- To specify whether line numbers are displayed, choose the Options->Show Line Numbers toggle.
- To specify the font size used when displaying a source file, choose the Options->Font->(Size) command.
- To specify the number of characters to use for tab stops, choose the Options->Tab Width->N spaces command.

- To change the name of the source viewer display, choose the Options->Change Source Viewer Name... command and enter the new name in the resulting dialog.

## To display Source Viewer information

1. Select the Info tab.



*Correlated with:* shows the Listing display associated with this source viewer.

*Data Source:* shows the logic analyzer whose captured data is being displayed.

*File:* shows the name of the source file being displayed.

## Loading and Saving Listing Configurations

Listing Tool settings can be saved to a configuration file along with the tools connected to it, and loaded from a previously saved configuration file.

- Loading Configuration Files (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)
- Saving Configuration Files (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

---

**NOTE:**

The *Load Configuration* window can be accessed via File->Load Configuration.

The *Save Configuration* window can be accessed via File->Save Configuration.

---



## Go to an Exact State

1. *Point* anywhere in the display area, then select *Goto state*.
2. Type in the desired state number.
3. Select the desired analyzer (data set).
4. Select *Goto*.
5. Optional - Simply select a previously visited state from the list. As you define states, they are added to the list.



---

## Go to an Exact Pattern

1. Point to any data column in the display area, then select *Goto pattern*.
2. Enter the desired pattern occurrence to find.
3. Select the desired reference point.

Beginning - from beginning of data set.

End - from end of data set.

Trigger - from the trigger point.

Display reference (see page 54) - from the state that is presently marked by the midscreen reference line.

4. Define (see page 43) the desired pattern.
5. Select *Goto*.




---

### NOTE:

Once you have a pattern search configured, to increment or decrement the occurrences, use the *Next* and *Prev* fields.

---

### Using the Goto Tab

You can quickly go to a single pattern or value by using the Goto tab as follows.

1. Select the Goto Tab.
2. Select Time or Samples.
3. Enter a time or sample value.
4. Select the *Goto* field. When the pattern or value is found, it is placed at center screen

### See Also

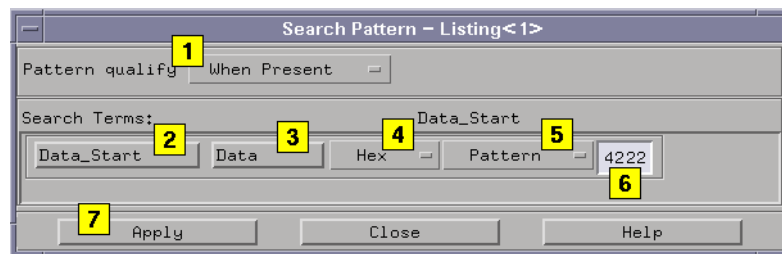
“What are Search Terms” on page 44

---

---

## Defining the Search Term

1. Select the pattern qualify option (see page 43).
2. Optional - assign a name to the search term.
3. Identify which data to search by selecting the desired data label.
4. Select the numeric base, symbols, or line # for the searched data.
5. Define the search term type to either Pattern or Range.
6. Enter the desired data pattern or range start/end patterns.
7. Select *Close*.



### See Also

“What are Search Terms” on page 44

## Pattern Qualify Options

The pattern qualify options determine where the search begins within the designated pattern or range.

- *When Entering*; the search begins at the beginning of the pattern or range.
- *When Exiting*; the search begins at the end of the pattern or range.
- *When Present*; the search begins at the beginning of the designated pattern or range occurrence when the greater-than and less-than criteria is matched. The pattern or range is considered present when the criteria is matched.
- *When Present >* (greater than); the search begins at the beginning of the designated pattern or range occurrence, and the designated > time value

**Go to an Exact Pattern**

was satisfied. The Time mode in the instrument tool must be on before this option is available.

- *When Present <* (less than); the search begins at the beginning of the designated pattern or range occurrence, and the designated < time value was satisfied. The Time mode in the instrument tool must be on before this option is available.
- *When Present <>* (less than - greater than); the search begins at the beginning of the designated pattern or range occurrence when the greater-than and less-than criteria is matched.

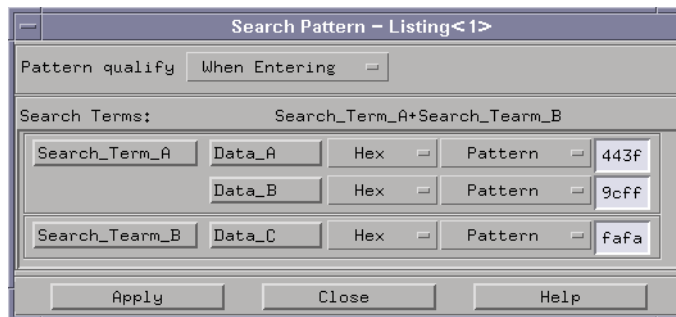
**What are Search Terms**

The concept of using *Terms* is the same for either *Search* terms or *Filter* terms. They are nothing more than constructed expressions that qualify what data is searched or filtered. An expression can be made up of a single term or multiple terms.

Each single term is made up of a data pattern or a range of data patterns for one or more labeled data sets. All labels within a single search term are logically *AND'ed* together. All search terms are logically *OR'ed* together.

**Example**

In the example below, the data set where either the binary data patterns of both labels *Data\_A* and *Data\_B* are found, or, the hex data pattern of label *Data\_C* is found.



## Inverse Assemble State Listings

Inverse assembly (IA) allows you to view state data as pseudocode mnemonics that resemble the program your system is running. To view inverse assembled data, you need a State Analyzer tool, a Listing tool, and the inverse assembler and configuration files for your microprocessor or bus.

1. There are three cases when you may wish to load full configuration files or inverse assembly files:
  - If you have a full configuration (tool configurations + IA) in the logic analysis system, simply load the configuration file (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume).
  - If you loaded a configuration file, but wish to load a different IA file, select here. (see page 22)
  - If you bought a new analysis probe, it came with a CD-ROM that contains the appropriate configuration file and IA file. Follow the instructions on the CD-ROM jacket to install the *Processor Support Package*. Then load the configuration using the Setup Assistant or the File Manager.
2. If the Listing window is not already on screen, select *Navigate*, the name of your state analyzer, and *Listing<1>*.
3. Select *Run* to acquire data.
4. If necessary, align the display (see page 21) by placing the first line of data you want inverse assembled, at the top of the display.
5. If necessary, choose an alignment start point (see page 19) (byte, word, etc.) by selecting *Invasm* in the Listing menu bar, selecting *Align*, then selecting the appropriate options in the Alignment dialog.
6. If desired, set up filters (see page 17) for the inverse assembled data.

### Multiple Display Configurations

To view multiple listings displays with the same IA, or a different IA, simply *add a listing tool* to the configuration, then *Load Invasm* from the new listing window.

### **Multiple Instrument Configurations**

To inverse assemble multiple data sets in the same listing, simply *connect another instrument tool* to the configuration, *load the appropriate IA configuration*, configure any necessary *time correlation*, then *Load Invasm* to the new instrument.

## Printing the Listing to a File

This operation prints the state listing to an ASCII file. You have the options of printing all labels or user-selected labels. You can also select how many states or samples to print.

### To Print Selected Labels

This operation only works with a keyboard.

1. While holding down the *Ctrl* key on the keyboard, select all desired labels. All selected labels will have a yellow box around them. Individually selected labels will remain selected until you either select them again while pressing the *Ctrl* key, or, by simply selecting any label.
2. From the Listing tool's menu bar select *File*, then select *Print to file*.
3. Select *Selected* columns to print.
4. Select either *All* or *Partial* as the range to print. If you select *Partial*, select *Samples* or *Time*, then set the *from* and *to* range fields.
5. Select either *Hard Disk* or *Flexible Disk*.
6. Select the path and filename to print the listing to. Use either the *Browse* field to build the directory path and filename, or type the path and filename directly into the text entry field.
7. Select *Save*. Filenames can only include the following alphanumeric characters:  
0-9, a-z, A-Z, (-), (\_), (.), (/), (\$), (:), and (+).

### To Print All Labels

1. From the Listing tool's menu bar select *File*, then select *Print to file*.
2. Select *All* columns to print.
3. Select either *All* or *Partial* as the range to print. If you select *Partial*, select *Samples* or *Time*, then set the *from* and *to* range fields.
4. Select either *Hard Disk* or *Flexible Disk*.
5. Select the path and filename to print the listing to. Use either the *Browse* field to build the directory path and filename, or type the path and filename directly into the text entry field.
6. Select *Save*. Filenames can only include the following alphanumeric

**Printing the Listing to a File**

characters:

0-9, a-z, A-Z, (-), ( \_), (.), (/), (\$), (:), and (+).

The following is an example of a stored print file:

Listing<1>

| State Number<br>Decimal | First_Label<br>Hex | Second_Label<br>Hex | Time<br>Relative |
|-------------------------|--------------------|---------------------|------------------|
| 0                       | 007E               | 00                  | 4.000 ns         |
| 1                       | 007E               | 00                  | 4.000 ns         |
| 2                       | 007E               | 00                  | 4.000 ns         |
| 3                       | 007E               | 00                  | 4.000 ns         |
| 4                       | 007E               | 00                  | 4.000 ns         |
| 5                       | 007E               | 00                  | 4.000 ns         |
| 6                       | 007E               | 00                  | 4.000 ns         |
| 7                       | 007E               | 00                  | 4.000 ns         |
| 8                       | 007E               | 00                  | 4.000 ns         |
| 9                       | 007E               | 00                  | 4.000 ns         |
| 10                      | 007E               | 00                  | 4.000 ns         |

Printed: 23 October 1998 (12:52:11)



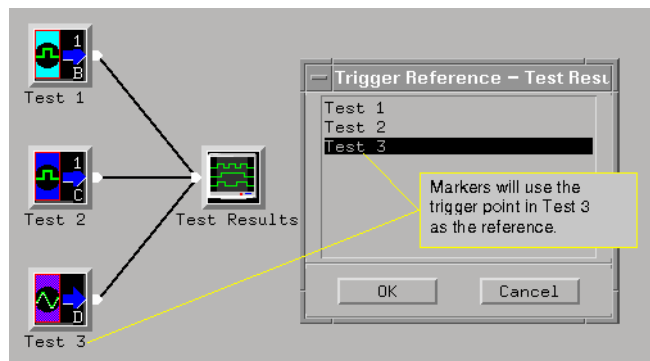
---

## Changing the Trigger Reference

When you have a configuration with more than one Instrument tool, you generate more than one data set. The *Reference trigger* function changes the focus of the display from the trigger point of one data set to another.

1. From the tool's menu bar select *Options*, then select *Reference trigger*.
2. From the Reference trigger dialog, select the desired data set.
3. Select *Ok*.

The following example shows a multiple Instrument configuration and the selection of a data set. The trigger point of the selected data set is used by the markers as the trigger reference.



## Working with Bookmarks

### **Temporary Bookmarks**

Temporary bookmarks are used to mark and find states in the current acquisition memory.

---

#### **NOTE:**

Temporary bookmarks go away when the next *Run* occurs.

- “Placing Temporary Bookmarks” on page 50
- “Goto Temporary Bookmarks” on page 51

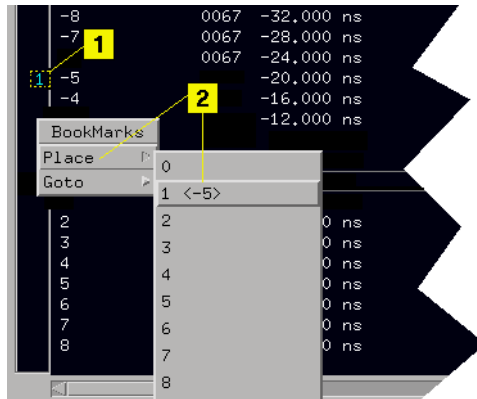
### **Permanent User-defined Bookmarks**

Permanent bookmarks are used to mark and find states that occur repeatedly through multiple acquisitions. When a new acquisition occurs, the bookmark is placed at the same state number in the new listing.

- “Placing Permanent Bookmarks” on page 51
  - “Goto Permanent Bookmarks” on page 52
- 

## Placing Temporary Bookmarks

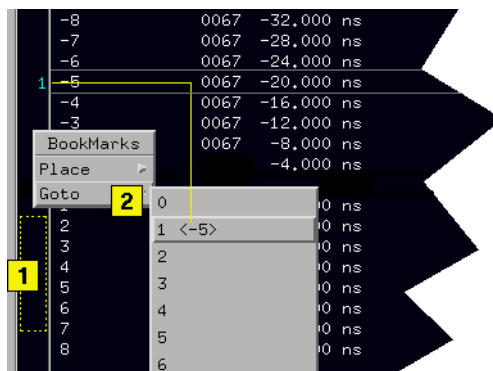
1. Point next to the desired state number in the left-most margin of the listing display.
2. Select *Place*, then select the desired bookmark number. The bookmark number appears at the left-most side of the listing.



---

## Goto Temporary Bookmarks

1. Point in the left-most margin of the listing display.
2. Select *Goto*, then select the desired bookmark number.



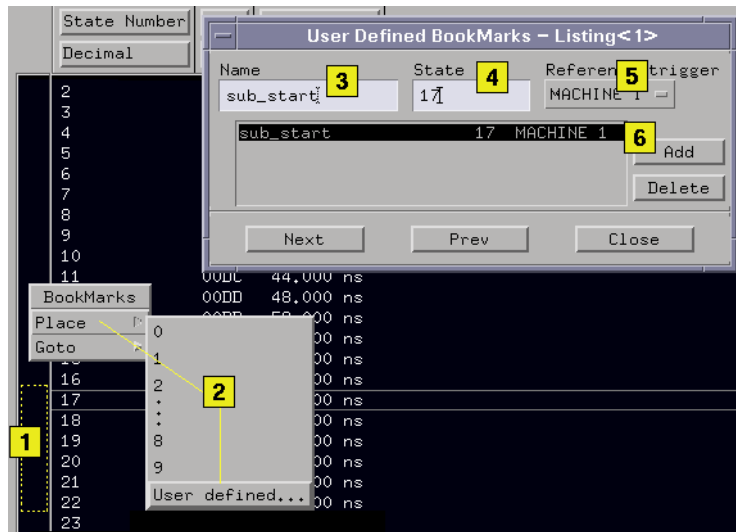
---

## Placing Permanent Bookmarks

1. Point next to the desired state number in the left-most margin of the listing display.
2. Select *Place*, then *User defined*.

**Working with Bookmarks**

3. Type in a bookmark name.
4. Type in the desired state number.
5. If you have multiple analyzers in the configuration, select the desired data set.
6. Select *Add*.

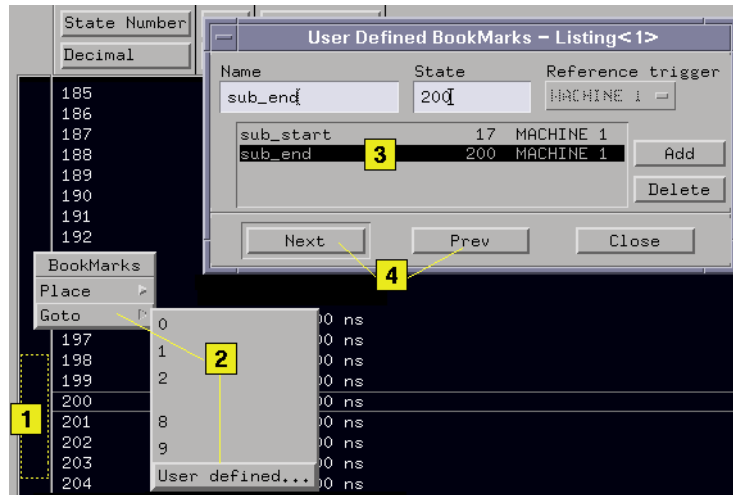


---

## Goto Permanent Bookmarks

1. *Point* to the left-most side of the listing.
2. Select *Goto*, then *User defined*.
3. Select the bookmark name to goto.
4. Select *Next* or *Prev*.

The highlighted bookmark is displayed at center screen as you increment/decrement the highlighter in the bookmark list.



Permanent user-defined bookmarks do not have numbers like temporary bookmarks. They simply appear at mid-screen when called for.

---

**NOTE:**

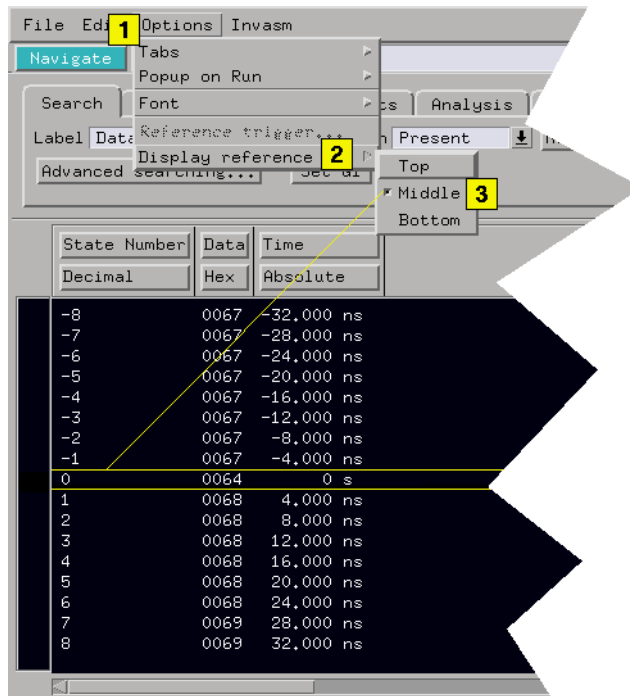
If the state you mark does not appear in subsequent acquisitions, the bookmark is placed at either the beginning or end of the current listing depending on whether the state number is less than or greater than the listing boundaries.

---

## Selecting the Display Reference

The display reference allows you to set where the trigger point (state 0) is referenced in the display.

1. From the tool's menu bar select *Options*.
2. Select *Display reference*.
3. Select the desired reference location.



## Printing the Listing Window

The print windows operation enables you to print just the Listing tool window. Use this operation if you want a hardcopy or electronic record of configurations and data currently displayed in the viewing area of the Listing window.

---

**NOTE:**

---

Only the currently displayed viewing area of the Listing window is printed. If any data or configuration fields appear offscreen, scroll the desired data or configuration fields into the window's viewing area before printing.

1. Optional - configure the Print Options (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume) if desired. Print Options include print destination, file format type, filename autoincrement, and color/b&w; pixel mapping.
2. In the Listing tool menu bar, select *File*, then select *Print This Window*. The print output will be as configured in the Print Options in step 1.

**See Also**

Print Options (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Printer Setup (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

“Including Comments on Screen Prints” on page 87

## Run/Group Run Function

### Using Run - Run All - Group Run

The Run/Stop functions are initiated by selecting icons in the icon bar at the top of the tool windows. All instrument, display, and analysis tool windows will have one of the Run icons shown below to initiate the run function.

When two or more instrument tools are configured, they can be run either independently or as a group. If run in a group, it is called an Intermodule measurement. Use the Intermodule Window (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume) to coordinate the arming in a "Group Run". A common "Group Run" configuration is to configure one instrument to trigger and then arm another instrument to start evaluation of its own trigger condition.



The Run Single icon appears if you have a single instrument configured in your measurement and you want to run a single acquisition.



The Run All icon always appears in the System, Workspace and Run Status windows. Also appears in instrument and display windows when you are using multiple instruments in your measurement and these instruments ARE NOT configured in an intermodule measurement (Group Run). This choice runs a single acquisition on all instruments in the configuration.



The Group Run icon appears in all windows when you are using multiple instruments, and these instruments are configured into a Group Run. This choice runs a single acquisition on all instruments in the Group Run configuration.



The Run Repetitive icon appears in all windows. It is used to run a *Run Single*, *Run All*, and a *Group Run* acquisition repetitively. The current run mode will continue to run until *Cancel* is selected.



## Using Stop



The Stop icon terminates all of the run functions shown above.

- Stops a single instrument running a measurement (perhaps waiting for a trigger condition).
- Stops all instruments running separate measurements (easily viewed from the Workspace window).
- Stops all instruments running in a Group Run configuration.

## See Also

“Demand Driven Data” on page 58

“Checking Run Status” on page 57

---

## Checking Run Status

The *Run Status* dialog provides status information about the currently configured instruments, and the status of the run with respect to the trigger specification.

To access the *Run Status* dialog, select the *Run Status* icon in the System Window, or, select *Window -> System -> Run Status*



## Demand Driven Data

When an analyzer measurement occurs, acquisition memory is filled with data that is then transferred to the display memory of the analysis or display tools you are using, as needed by those tools. In normal use, this *demand driven data* approach saves time by not transferring unnecessary data.

Since acquisition memory is cleared at the beginning of a measurement, stopping a run may create a discrepancy between acquisition memory and the memory buffer of connected tools. Without a complete trace of acquisition memory, the display memory will appear to have 'holes' in it which appear as filtered data.

This situation will occur in these cases:

- If you stop a repetitive measurement after analyzer data has been cleared and before the measurement is complete.
- If a trigger is not found by the analyzer and the run must be stopped to regain control.

To make sure all of the data in a repetitive run is available for viewing:

- In the workspace, attach a Filter tool to the output of the analyzer.
- In the Filter, select "Pass Matching Data"
- In the filter terms, assure the default pattern of all "Don't Care" (Xs).

This configuration will always transfer all data from acquisition memory. While this configuration will increase the time of each run, it will guarantee that repetitive run data is available regardless of when it is stopped.

## Adding and Deleting Tools

---

**NOTE:**

The add operation is also done for you automatically when you select an Instrument Icon (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume) from the System window. For an overview on the automatic configuration of measurements, refer to Overview - Starting a New Measurement (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume).

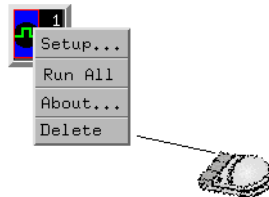
---

**To Add a Tool**

1. *Point* to the new tool in the toolbox, then *drag and drop* the new tool on top of any current tool in the measurement setup.  
New tools that are dropped on top of a current tool are automatically connected to the measurement.

**To Delete a Tool**

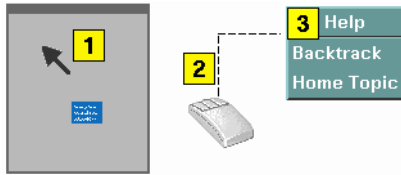
1. Point to the tool you want to delete.
2. Select *Delete*.

**See Also**

- “Clearing the Workspace” on page 62
- “Connecting Tools Together” on page 61
- “Repositioning Tools in the Workspace” on page 63

## Help - How to Navigate Quickly

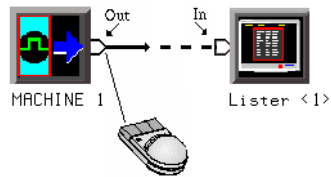
1. Place mouse cursor anywhere in a help window.
2. Press the right mouse button.
3. Select desired destination.



You can also access all navigation and search commands from the help window menu bar.

## Connecting Tools Together

If you *drag and drop* tools into open space in the workspace, you must create a data path between the tools by connecting their output and input ports.



### **To Connect Output and Input Ports**

1. *Point* to the tool output port.
2. Press and hold, then move the cursor over to a tool input port, then release.  
You should now have a line, representing a data path, drawn between tool data ports.

## Clearing the Workspace

Use the *Clear Workspace* option to remove all Instrument, Display, Analysis, and Utility tools from the workspace.

1. In the main window menu bar, select *File*, then select *Clear Workspace*.

---

**NOTE:**

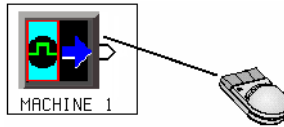
The *Clear Workspace* option does not clear parameter settings within the tools. To clear the Workspace and reset all parameters within the tools back to the defaults, exit out of the system and start a new session.

---

## Repositioning Tools in the Workspace

Once tools are placed on the workspace (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume), you can reposition them to help you visualize your measurement, or to reveal their input and output ports for connection into the measurement.

### To Reposition a Single Tool



1. *Point* to the tool to move.
2. *Drag and drop* the tool on its new location.

## The Symbols Tab

The Symbols tab offers control of the *symbols* capabilities. Symbols represent patterns and ranges of values found on labeled sets of bits. Two kinds of symbols are available:

- Object File Symbols. These are symbols from your source code and symbols generated by your compiler.
- User-Defined Symbols. These are symbols you create.

To load symbols, see:

- “To Load Object File Symbols” on page 66
- “To Load User-Defined Symbols” on page 84

Symbols are available for all state and timing analyzers. Each label listed in the Format menu can have its own group of symbols associated with it.

- “User-Defined Symbols” on page 83
- “Setting Up Object File Symbols” on page 66
- “Using Symbols In The Logic Analyzer” on page 78
- “Displaying Data in Symbolic Form” on page 65



## Displaying Data in Symbolic Form

You can display data in symbolic form in some of the display tools, such as the Listing display and the Waveform display.

### To View Symbolic Values in a Waveform Display

1. Select the label name where you want to display symbolic values.
2. Choose *Properties...*
3. In the Properties dialog:
  - Set ShowValue to *On*.
  - Set Base to *Symbols* or *Line #s*.
  - Select the *OK* button.

The symbolic names for the values now appear in the overlaid bus waveform.

### To View Symbolic Values in a Listing Display

1. Select the numeric base of the label where you want to display symbolic values.
2. Set the numeric base to *Symbols* or *Line #s*.  
The symbolic names for the values now appear instead of numeric data.

## Setting Up Object File Symbols

Object file symbols can include variable names, procedure or function names, and source file names with line numbers. The linkage between symbol names and address or data values comes from one of two sources:

- Object files that are created by your compiler/linker.
- ASCII symbol files you create with a text editor.

### To use object file symbols

1. Generate an object file with symbolic information using your software development tools.
2. If your language tools cannot generate object file formats that are supported by the logic analyzer, create an ASCII symbol file (see page 70).
3. Load the object file (see page 66) or ASCII symbol file into the logic analyzer.
4. If necessary, relocate sections of your code (see page 68).

### See Also

“Using Symbols In The Logic Analyzer” on page 78

“Symbol File Formats” on page 69

---

## To Load Object File Symbols

1. Select the *Symbol* tab and then the *Object File* tab.
2. Select the label name you want to load object file symbols for.  
In most cases you will select the label representing the address bus of the processor you are analyzing.
3. Specify the directory to contain the symbol database file (*.ns*) in the field under, *Create Symbol File (.ns) in This Directory*. Select the *Browse...* button if you wish to find an existing directory name.
4. In the *Load This Object/Symbol File For Label* field, enter the object file

name containing the symbols. Select the *Browse...* button to find the object file and select the *Load* button in the Browser dialog.

If your logic analyzer is NFS mounted to a network, you can select object files from other servers.

### **To reload object file symbols**

1. Select the object file/symbol file to reload from the *Object Files with Symbols Loaded For Label* field.
2. Select the *Reload* button.

### **Value update**

The values of the object file symbols being used as terms or as SPA state-interval ranges will be updated automatically each time the object file symbols are reloaded.

### **Configuration file save**

The name of the current object file is saved when a configuration file is saved. The object file will be reloaded when the configuration is loaded.

### **Multiple files**

You can load the same symbol file into several different analyzers, and you can load multiple symbol files into one analyzer. Symbols from all the files you load will appear together in the object file symbol selector that you use to set up resource terms.

### **Object file versions**

During the load process, a symbol database file with a *.ns* extension will be created by the system. One *.ns* database file will be created for each symbol file you load. Once the *.ns* file is created, the Symbol Utility will use this file as its working symbol database. The next time you need to load symbols into the system, you can load the *.ns* file explicitly, by placing the *.ns* file name in the *Load This Object/Symbol File For Label* field.

If you load an object file that has been loaded previously, the system will compare the time stamps on the *.ns* file and the object file. If the object file is newer, the *.ns* file will be created. If the object file has not

been updated since it was last loaded, the existing *.ns* file will be used.

**See Also**

“Using Symbols In The Logic Analyzer” on page 78

“Symbol File Formats” on page 69

---

## Relocating Sections of Code

Use this option to add offset values to the symbols in an object file. You will need this if some of the sections or segments of your code are relocated in memory at run-time. This can occur if your system dynamically loads parts of your code so that the memory addresses that the code is loaded into are not fixed.

### To Relocate a Single Section of Code

1. Select the *Symbol* tab and then the *Object File* subtab.
2. Select the *Relocate Sections...* button.
3. In the Address column, select the address you wish to relocate.
4. In the *Edit selected section* field, enter the new address.
5. Select *Apply Edit*.
6. Repeat steps 3 through 5 above for any other sections to be relocated.
7. Select the *Close* button.

### To Relocate All Sections of Code

1. Select the *Symbol* tab and then the *Object File* subtab.
2. Select the *Relocate Sections...* button.
3. Enter the value you wish to use in the *Offset all selections by* field.
4. Select the *Apply Offset* button.
5. Select the *Close* button.

---

## To Delete Object File Symbol Files

1. Select the *Symbol* tab, and then the *Object File* subtab.
2. Select the file name you want to delete in the text box labeled, *Object Files with Symbols Loaded For Label*.
3. Select the *Unload* button.

---

## Symbol File Formats

The logic analysis system can read symbol files in the following formats:

- OMF96
- OMFx86
- IEEE-695
- ELF/stabs
- TI COFF

For ELF/stabs, and ELF/stabs/Mdebug files, C++ symbols are demangled so that they can be displayed in the original C++ notation. To improve performance for these ELF symbol files, type information is not associated with variables. Hence, some variables (typically a few local static variables) may not have the proper size associated with them. They may show a size of 1 byte and not the correct size of 4 bytes or even more. All other information function ranges, line numbers, global variables and filenames will be accurate. These behaviors may be changed by creating a readers.ini (see page 75) file.

### See Also

“Creating ASCII Symbol Files” on page 70

“Creating a readers.ini File” on page 75

## Creating ASCII Symbol Files

If your language tool chain does not produce object files in a supported format, you can create an ASCII symbol file to define symbols. You can also use an ASCII symbol file to define symbols that are not included in your object file.

You can create an ASCII symbol file using any text editor that supports ASCII format text. Each entry in the file you create must be a string of ASCII characters consisting of a symbol name followed by an address or address range. The address or address range must be a hexadecimal number. It must appear on the same line of the text file as the symbol name and it must be separated from the symbol name by one or more blank spaces or tabs. Address ranges must be in the following format:

```
beginning address..ending address
```

Two formats are available for creating ASCII symbol files:

“Simple Format” on page 70

“Record Header Format” on page 70

---

**NOTE:**

---

It is possible to generate ASCII symbol files from the symbol or load map output of most language tools.

### Simple Format

An ASCII symbol file can be a simple list of name/address pairs.

**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.

### Record Header Format

An ASCII symbol file can be divided into records using key words,

called *record headers*. The different records allow you to specify different kinds of symbols, with differing characteristics. An ASCII symbol file can contain any of the following kinds of records:

“Start Address” on page 72

“Sections” on page 72

“Functions” on page 72

“Variables” on page 74

“Source Line Numbers” on page 73

“Comments” on page 74

The record headers must be enclosed in square brackets, like this: [HEADER]. If no record header is specified, the lines following are assumed to be symbol definitions in one of the VARIABLES formats:

```
variable    address
variable    start..end
variable    start address    size
```

**Example**

Here is an ASCII symbol 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
```

## Chapter 1: Using the Listing Tool

### Setting Up Object File Symbols

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

#### Start Address . Format

```
[START ADDRESS]
address
```

*address* - The address of the program entry point, in hexadecimal.

#### Example

```
[START ADDRESS]
00001000
```

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

#### Format

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

*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
```

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

#### Format

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

*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* - (optional) Attribute 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. In other words, this is an absolute segment.

### Example

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

---

**NOTE:**

If Section definitions are used in an ASCII symbol file, any subsequent Function or Variable definitions must fall within the address ranges of one of the defined Sections. Those Functions and Variables that do not will be ignored by the Symbol Utility.

---

**Source Line Numbers** . Use SOURCE LINES to associate addresses with lines in your source files.

### Format

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

*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
```

## Chapter 1: Using the Listing Tool

### Setting Up Object File Symbols

```
14          0000100A
22          0000101E
```

#### See Also

Using the Source Viewer (see the *Listing Display Tool* help volume)

**Variables.** You can specify symbols for variables using:

- The address of the variable.
- The address and the size of the variable.
- The range of addresses occupied by the variable.

If you give only the address of a variable, the size is assumed to be 1 byte.

#### Format

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

*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* - (optional) The size of the variable, in bytes, in decimal.

#### Example

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

**Comments .** Any text following a # character is ignored by the Symbol Utility. The # can be used to comment a file. Comments can appear on a line by themselves, or on the same line, following a symbol entry.

#### Format

```
#comment text
```

### Example

```
#This is a comment
```

---

## Creating a readers.ini File

You can change how an ELF/Stabs, Tioff or Coff/Stabs symbol file is processed by creating a reader.ini file.

1. Create the reader.ini file on your workstation or PC.
2. Copy the file to /logic/symbols/readers.ini on the logic analysis system.

### Reader options

#### C++Demangle

```
1= Turn on C++ Demangling (Default)  
0= Turn off C++ Demangling
```

#### C++DemOptions

```
803= Standard Demangling  
203= GNU Demangling (Default Elf/Stabs)  
403= Lucid Demangling  
800= Standard Demangling without function parameters  
200= GNU Demangling without function parameters  
400= Lucid Demangling without function parameters
```

#### MaxSymbolWidth

```
80= Column width max of a function or variable symbol  
Wider symbols names will be truncated.  
(Default 80 columns)
```

#### OutSectionSymbolValid

```
0= Symbols whose addresses aren't within the  
defined sections are invalid (Default)  
1= Symbols whose addresses aren't within the  
defined sections are valid
```

This option must be specified in the Nsr section of the Readers.ini file:

```
[Nsr]  
OutSectionSymbolValid=1
```

#### ReadElfSection

```
2= Process all globals from ELF section (Default)  
Get size information of local variables
```

### Setting Up Object File Symbols

```
1= Get size information of global and local variables
   Symbols for functions will not be read, and
   only supplemental information for those symbols in
   the Dwarf or stabs section will be read.
0= Do not read the Elf Section
```

If a file only has an ELF section this will have no effect and the ELF section will be read completely. This can occur if the file was created without a "generate debugger information" flag (usually -g). Using the -g will create a Dwarf or Stabs debug section in addition to the ELF section.

### StabsType

```
StabsType=0 Reader will determine stabs type (Default)
StabsType=1 Older style stabs
              (Older style stabs have individual symbol
              tables for each file that was linked into
              the target executable, the indexes of each
              symbol table restart at 0 for each file.)
StabsType=2 Newer style stabs
              (New style stabs have a single symbol table
              where all symbols are merged into a large
              symbol array).
```

### ReadOnlyTicoffPage

ReadOnlyTicoffPage tells the ticoff reader to read only the symbols associated with the specified page (as an example 'ReadOnlyTicoffPage=0' reads only page 0 symbols). A value of -1 tells the ticoff readers to read symbols associated with all pages.

```
ReadOnlyTicoffPage=-1 Read all symbols associated with all
                      ticoff pages (Default)
ReadOnlyTicoffPage=p  Read only symbols associated with
                      page 'p' (where p is any integer
                      between 0 and n the last page of
                      the object file).
```

### AppendTicoffPage

AppendTicoffPage tells the ticoff reader to append the page number to the symbol value. This assumes that the symbol value is 16-bits wide and that that page number is a low positive number which can be ORed into the upper 16 bits of an address to create a new 32-bit symbol address. For example, if the page is 10 decimal and the symbol address is 0xF100 then the new symbol address will be 0xAF100.

```
AppendTicoffPage=1  Append the ticoff page to the symbol
                    address
AppendTicoffPage=0  Do not append the ticoff page to the
                    symbol address (Default)
```

## Examples

### Example for Elf/Stabs

```
[ReadersElf]  
C  
C  
MaxSymbolWidth=60  
StabsType=2
```

### Example for Coff/Stabs (using Ticoff reader)

```
[ReadersTicoff]  
C  
C  
MaxSymbolWidth=60  
StabsType=2
```

### Example for Ticoff

```
[ReadersTicoff]  
C  
C  
MaxSymbolWidth=60  
ReadOnlyTicoffPage=4  
AppendTicoffPage=1
```

## Using Symbols In The Logic Analyzer

The ways symbols can be used in the logic analyzer are listed below:

- “Using Symbols As Trigger Terms” on page 78
  - “Using Symbols as Search Patterns in Listing Displays” on page 79
  - “Using Symbols as Trigger Terms in the Source Viewer” on page 79
  - “Using Symbols as Pattern Filter Terms” on page 79
  - “Using Symbols as Ranges in the Software Performance Analyzer” on page 80
  - “Displaying Data in Symbolic Form” on page 65
- 

### Using Symbols As Trigger Terms

You can use either one or both types of symbols as terms within your trigger sequence:

- *Object File Symbols.*
  - *User-Defined Symbols.*
1. At the bottom of the analyzer Trigger window, select the label button next to one of the resource terms, and choose *Replace*.
  2. In the Resource selection dialog, select a label to be used in your trigger sequence.  
Use a label that has *symbols* loaded.
  3. Set the numeric base of the trigger term to *Symbols* or *Line #s*.
  4. Select the button to the right of the numeric base field.
  5. In the *Symbol Selector* (see page 80) dialog, select the symbol you want to use.

---

**NOTE:**

The values of object file symbols used as trigger terms are automatically updated when the object file symbols are reloaded (see page 66).

---

---

## Using Symbols as Search Patterns in Listing Displays

1. Under the *Search* tab in the Listing display, select the *Advanced searching* button.
2. In the Goto Pattern dialog, select the *Define* button.
3. In the Search Pattern dialog, select the *Symbols* numeric base.
4. Select *Pattern*, *Range*, *Not Pattern*, or *Not Range*.
5. Select the button to the right of the numeric base field.
6. In the *Symbol Selector* (see page 80) dialog, select the symbol you want to use.

### See Also

Go to an Exact Pattern. (see the *Listing Display Tool* help volume)

---

## Using Symbols as Trigger Terms in the Source Viewer

1. In the Source Viewer menu bar, select *Trace*, and select *Trace Setup*.
2. In the Source Line Trigger dialog, select *Symbols* or *Line #s* in the numeric base field.
3. Select *Pattern*, *Range*, *Not Pattern*, or *Not Range*.
4. Select the button to the right of the numeric base field.
5. In the *Symbol Selector* (see page 80) dialog, select the symbol you want to use.

### See Also

To modify the trace setup. (see the *Listing Display Tool* help volume)

---

## Using Symbols as Pattern Filter Terms

1. Select the numeric base field beside the selected filter term, and select

*Symbols or Line #s.*

2. Select *Pattern, Range, Not Pattern, or Not Range.*
3. Select *Remove Matching Data or Pass Matching Data,* as desired.
4. Select the *Absolute XXXX* button.
5. In the *Symbol Selector* (see page 80) dialog, select the symbol you want to use.

---

## Using Symbols as Ranges in the Software Performance Analyzer

1. In the state interval SPA tool, select the *Symbols* button in the Define Ranges dialog.
2. In the Symbol Selector (see page 81) dialog, select the symbol or group of symbols you want to use as ranges in your measurement.

### See Also

Defining State Interval Ranges. (see the *System Performance Analyzer* help volume)

### Using the Symbol Selector Dialog

1. In the *Symbol Selector* dialog, select the symbol you want to use. All of your symbols for the current label, regardless of type, will be available in the dialog.
  - Use the Search Pattern (see page 81) field to filter the list of symbols by name. You can use the Recall button to recall a desired Search Pattern.
  - Use the Find Symbols of Type selections to filter the symbols by type.
2. Select the symbol you want to use in the list of *Matching Symbols.*
3. If you are using object file symbols, you may need to:
  - Set *Offset By* (see page 82) to compensate for microprocessor prefetches.
  - Set *Align to x Byte* (see page 82) to trigger on odd-byte boundaries.



4. Select the Beginning, End, or Range of the symbol.
5. Select the *OK* button.  
The name of your symbol now appears as the value of the resource term.
6. Select the *Cancel* button to exit the *Symbol Selector* dialog without selecting a symbol.

## Using the Symbol Selector Dialog

1. In the *Symbol Selector* dialog, select the symbol you want to use. All of your symbols, regardless of type, will be available in the dialog.
  - Use the Search Pattern (see page 81) field to filter the list of symbols by name. You can use the Recall button to recall a desired Search Pattern.
  - Use the Find Symbols of Type selections to filter the symbols by type.
2. Drag to select the symbols you want to use in the list of *Matching Symbols*.
  - Select the *Select All* button to select all symbols in the list.
  - Select the *Unselect All* button to unselect all symbols in the list.
3. Select the *Add Selected Symbols To Range List* button to place the selected symbols into the *Current ranges* list in the Define Ranges dialog.
4. Select the *Close* button to exit the *Symbol Selector* dialog.

## Search Pattern

Use this field to locate particular symbols in the symbol databases. To use this field, enter the name of a file or symbol. The system searches the symbol database for symbols that match this name. Symbols that match appear in the list of *Matching Symbols*. You can also use wildcard characters to find symbols.

### Asterisk wildcard (\*)

The asterisk wildcard represents "any characters." When you perform a search on the symbol database using just the asterisk, you will see a list of all symbols contained in the database. The asterisk can also be added to a search word to find all symbols that begin or end with the

same letters. For example, to find all of the symbols that begin with the letters "st", select the Search Pattern field and enter "st\*".

### **Align to x Byte Option**

Most processors do not fetch instructions from memory on byte boundaries. In order to trigger a logic analyzer on a symbol at an odd-numbered address, the address must be masked off. The "Align to x Byte" option allows you to mask off an address.

#### **Example**

Assume the symbol "main" occurs at address 100F. The processor being probed is a 68040, which fetches instructions on long-word (4-byte) boundaries. In order to trigger on address 100F, the Align to x Byte option sets the two least-significant address bits to "don't cares". This qualifies any address from 100C through 100F.

### **Offset By Option**

The Offset By option allows you to add an offset value to the starting point of the symbol that you want to use as a term. You might do this in order to trigger on a point in a function that is beyond the preamble of the function, or to trigger on a point that is past the prefetch depth of the processor. Setting an offset helps to avoid false triggers in these situations. The offset specified in the Offset By field is applied before the address masking is done by the "Align to x Byte" option.

#### **Example**

An 80386 processor has a prefetch depth of 16 bytes. Assume functions *func1* and *func2* are adjacent to each other in physical memory, with *func2* following *func1*. In order to trigger on *func2* without getting a false trigger from a prefetch beyond the end of *func1*, you need to add an offset value to your trigger term. The offset value must be equal to or greater than the prefetch depth of the processor. In this case, you would add an offset of 16 bytes to your trigger term. You would set the value of the "Offset By" field to 10 hex. Now, when you specify *func2* as your trigger term, the logic analyzer will trigger on address *func2*+10.

## User-Defined Symbols

“To Create User-Defined Symbols” on page 83

“To Replace User-Defined Symbols” on page 83

“To Delete User-Defined Symbols” on page 84

“To Load User-Defined Symbols” on page 84

---

## To Create User-Defined Symbols

1. Under the *Symbol* tab, select the *User Defined* tab.
2. Select the label name you want to define symbols for.
3. At the bottom of the *User Defined* tab, enter a symbol name in the entry field.
4. Select a numeric base.
5. Select *Pattern* or *Range* type for the symbol.
6. Enter values for the pattern or range the symbol will represent.
7. Select the *Add* button.
8. Repeat steps 3 through 7 for additional symbols.
9. You can edit your list of symbols by using *Replace* (see page 83) and *Delete* (see page 84), if desired.

### See Also

“Using Symbols In The Logic Analyzer” on page 78

---

## To Replace User-Defined Symbols

1. Under the *Symbol* tab, select the *User Defined* tab.
  2. Select the label you want to replace symbols for.
-

## User-Defined Symbols

3. Select the symbol to replace.
4. At the bottom of the *User Defined* tab, modify the symbol name, numeric base, Pattern/Range type, and value, as desired.
5. Select the *Replace* button.
6. Repeat steps 3 through 5 to replace other symbols, if desired.

---

## To Delete User-Defined Symbols

1. Under the *Symbol* tab, select the *User Defined* tab.
2. Select the label you want to delete symbols from.
3. Select the symbol to delete.
4. Select the *Delete* button.
5. Repeat steps 3 and 4 to delete other symbols, if desired.

---

## To Load User-Defined Symbols

If you have already saved a configuration file, and the configuration included user-defined symbols, load the file with its symbols, as follows:

1. In the menu bar of your analyzer window, select *File* and then *Load Configuration...*
2. In the Load Configuration dialog, select the directory and filename to be loaded.
3. Select the target of the load operation.
4. Select the *Load* button.  
User-defined symbols that were resident in the logic analyzer when the configuration was saved are now loaded and ready to use.

### See Also

“Using Symbols In The Logic Analyzer” on page 78

## Seeing Measurement Results - Popup on Run

By default, measurement results do not pop up automatically when an analyzer completes its run. If you wish to have a display of measurement results pop up when a run is completed, do the following:

- In the menu bar of the display tool, select *Options*, then select *Popup on Run* and set to *On*.

## Turning Tabs On or Off in the Display Window

The tabs in the display window give you quick access and visual reference to all the capabilities in the display window. You may wish to turn the tabs off to simplify display content.

- In the menu bar of the display tool, select *Options*, then select *Tabs* and set to *On* or *Off*.

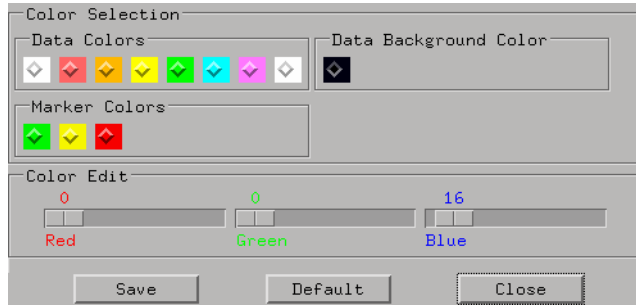
## Including Comments on Screen Prints

When you have tabs turned on, you can also enter your own comments on screen. This allows you to obtain a printed copy of your screen that includes your comments.

- Turn tabs on, (see page 86) and under the *Tabs On* option, select *Stack comments*.
- Enter the text you wish to see in the print of your display. Each print can include multiple comment lines. If you need more than the default two comment lines, resize the comment text window using the sash.

---

## Editing Colors



### To edit a color

1. Select a color to change.
2. Move the *Color Edit* sliders to obtain the desired colors.  
The colors change in the display to show the result of your modifications.
3. Select *Save* to accept and use the new colors, and save them as the powerup default colors.  
Optional - Select *Default* to restore the original colors and save them as the powerup default colors.
4. Select *Close*.



---

## Glossary

**absolute** Denotes the time period or count of states between a captured state and the trigger state. An absolute count of -10 indicates the state was captured ten states before the trigger state was captured.

**acquisition** Denotes one complete cycle of data gathering by a measurement module. For example, if you are using an analyzer with 128K memory depth, one complete acquisition will capture and store 128K states in acquisition memory.

**analysis probe** A probe connected to a microprocessor or standard bus in the device under test. An analysis probe provides an interface between the signals of the microprocessor or standard bus and the inputs of the logic analyzer. Also called a *preprocessor*.

**analyzer 1** In a logic analyzer with two *machines*, refers to the machine that is on by default. The default name is *Analyzer<N>*, where N is the slot letter.

**analyzer 2** In a logic analyzer with two *machines*, refers to the machine that is off by default. The default name is *Analyzer<N2>*, where N is the slot letter.

**arming** An instrument tool must be

armed before it can search for its trigger condition. Typically, instruments are armed immediately when *Run* or *Group Run* is selected. You can set up one instrument to arm another using the *Intermodule Window*. In these setups, the second instrument cannot search for its trigger condition until it receives the arming signal from the first instrument. In some analyzer instruments, you can set up one analyzer *machine* to arm the other analyzer machine in the *Trigger Window*.

**asterisk (\*)** See *edge terms*, *glitch*, and *labels*.

**bits** Bits represent the physical logic analyzer channels. A bit is a *channel* that has or can be assigned to a *label*. A bit is also a position in a label.

**card** This refers to a single instrument intended for use in the Agilent Technologies 16600A-series or 16700A/B-series mainframes. One card fills one slot in the mainframe. A module may comprise a single card or multiple cards cabled together.

**channel** The entire signal path from the probe tip, through the cable and module, up to the label grouping.

**click** When using a mouse as the

---

## Glossary

pointing device, to click an item, position the cursor over the item. Then quickly press and release the *left mouse button*.

**clock channel** A logic analyzer *channel* that can be used to carry the clock signal. When it is not needed for clock signals, it can be used as a *data channel*, except in the Agilent Technologies 16517A.

**context record** A context record is a small segment of analyzer memory that stores an event of interest along with the states that immediately preceded it and the states that immediately followed it.

**context store** If your analyzer can perform context store measurements, you will see a button labeled *Context Store* under the Trigger tab. Typical context store measurements are used to capture writes to a variable or calls to a subroutine, along with the activity preceding and following the events. A context store measurement divides analyzer memory into a series of context records. If you have a 64K analyzer memory and select a 16-state context, the analyzer memory is divided into 4K 16-state context records. If you have a 64K analyzer memory and select a 64-state context, the analyzer memory will be

divided into 1K 64-state records.

**count** The count function records periods of time or numbers of state transactions between states stored in memory. You can set up the analyzer count function to count occurrences of a selected event during the trace, such as counting how many times a variable is read between each of the writes to the variable. The analyzer can also be set up to count elapsed time, such as counting the time spent executing within a particular function during a run of your target program.

**cross triggering** Using intermodule capabilities to have measurement modules trigger each other. For example, you can have an external instrument arm a logic analyzer, which subsequently triggers an oscilloscope when it finds the trigger state.

**data channel** A *channel* that carries data. Data channels cannot be used to clock logic analyzers.

**data field** A data field in the pattern generator is the data value associated with a single label within a particular data vector.

**data set** A data set is made up of all labels and data stored in memory of any single analyzer machine or

---

## Glossary

instrument tool. Multiple data sets can be displayed together when sourced into a single display tool. The Filter tool is used to pass on partial data sets to analysis or display tools.

**debug mode** See *monitor*.

**delay** The delay function sets the horizontal position of the waveform on the screen for the oscilloscope and timing analyzer. Delay time is measured from the trigger point in seconds or states.

**demo mode** An emulation control session which is not connected to a real target system. All windows can be viewed, but the data displayed is simulated. To start demo mode, select *Start User Session* from the Emulation Control Interface and enter the demo name in the *Processor Probe LAN Name* field. Select the *Help* button in the *Start User Session* window for details.

**deskewing** To cancel or nullify the effects of differences between two different internal delay paths for a signal. Deskewing is normally done by routing a single test signal to the inputs of two different modules, then adjusting the Intermodule Skew so that both modules recognize the signal at the same time.

**device under test** The system under test, which contains the circuitry you are probing. Also known as a *target system*.

**don't care** For *terms*, a "don't care" means that the state of the signal (high or low) is not relevant to the measurement. The analyzer ignores the state of this signal when determining whether a match occurs on an input label. "Don't care" signals are still sampled and their values can be displayed with the rest of the data. Don't cares are represented by the *X* character in numeric values and the dot (.) in timing edge specifications.

**dot (.)** See *edge terms*, *glitch*, *labels*, and *don't care*.

**double-click** When using a mouse as the pointing device, to double-click an item, position the cursor over the item, and then quickly press and release the *left mouse button* twice.

**drag and drop** Using a Mouse: Position the cursor over the item, and then press and hold the *left mouse button*. While holding the left mouse button down, move the mouse to drag the item to a new location. When the item is positioned where you want it, release the mouse button.

---

## Glossary

Using the Touchscreen:  
Position your finger over the item, then press and hold finger to the screen. While holding the finger down, slide the finger along the screen dragging the item to a new location. When the item is positioned where you want it, release your finger.

**edge mode** In an oscilloscope, this is the trigger mode that causes a trigger based on a single channel edge, either rising or falling.

**edge terms** Logic analyzer trigger resources that allow detection of transitions on a signal. An edge term can be set to detect a rising edge, falling edge, or either edge. Some logic analyzers can also detect no edge or a *glitch* on an input signal. Edges are specified by selecting arrows. The dot (.) ignores the bit. The asterisk (\*) specifies a glitch on the bit.

**emulation module** A module within the logic analysis system mainframe that provides an emulation connection to the debug port of a microprocessor. An E5901A emulation module is used with a target interface module (TIM) or an analysis probe. An E5901B emulation module is used with an E5900A emulation probe.

**emulation probe** The stand-alone equivalent of an *emulation module*. Most of the tasks which can be performed using an emulation module can also be performed using an emulation probe connected to your logic analysis system via a LAN.

**emulator** An *emulation module* or an *emulation probe*.

**Ethernet address** See *link-level address*.

**events** Events are the things you are looking for in your target system. In the logic analyzer interface, they take a single line. Examples of events are *Label1 = XX* and *Timer 1 > 400 ns*.

**filter expression** The filter expression is the logical *OR* combination of all of the filter terms. States in your data that match the filter expression can be filtered out or passed through the Pattern Filter.

**filter term** A variable that you define in order to specify which states to filter out or pass through. Filter terms are logically *OR*'ed together to create the filter expression.

**Format** The selections under the logic analyzer *Format* tab tell the

---

## Glossary

logic analyzer what data you want to collect, such as which channels represent buses (labels) and what logic threshold your signals use.

**frame** The Agilent Technologies 16600A-series or 16700A/B-series logic analysis system mainframe. See also *logic analysis system*.

**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.

**glitch** A glitch occurs when two or more transitions cross the logic threshold between consecutive timing analyzer samples. You can specify glitch detection by choosing the asterisk (\*) for *edge terms* under the timing analyzer Trigger tab.

**grouped event** A grouped event is a list of *events* that you have grouped, and optionally named. It can be reused in other trigger sequence levels. Only available in Agilent Technologies 16715A, 16716A, and 16717A logic analyzers.

**held value** A value that is held until

the next sample. A held value can exist in multiple data sets.

**immediate mode** In an oscilloscope, the trigger mode that does not require a specific trigger condition such as an edge or a pattern. Use immediate mode when the oscilloscope is armed by another instrument.

**interconnect cable** Short name for *module/probe interconnect cable*.

**intermodule bus** The intermodule bus (IMB) is a bus in the frame that allows the measurement modules to communicate with each other. Using the IMB, you can set up one instrument to *arm* another. Data acquired by instruments using the IMB is time-correlated.

**intermodule** Intermodule is a term used when multiple instrument tools are connected together for the purpose of one instrument arming another. In such a configuration, an arming tree is developed and the group run function is designated to start all instrument tools. Multiple instrument configurations are done in the Intermodule window.

**internet address** Also called Internet Protocol address or IP address. A 32-bit network address. It

---

## Glossary

is usually represented as decimal numbers separated by periods; for example, 192.35.12.6. Ask your LAN administrator if you need an internet address.

**labels** Labels are used to group and identify logic analyzer channels. A label consists of a name and an associated bit or group of bits. Labels are created in the Format tab.

**line numbers** A line number (Line #s) is a special use of *symbols*. Line numbers represent lines in your source file, typically lines that have no unique symbols defined to represent them.

**link-level address** Also referred to as the Ethernet address, this is 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.

**local session** A local session is when you run the logic analysis system using the local display connected to the product hardware.

**logic analysis system** The Agilent Technologies 16600A-series or

16700A/B-series mainframes, and all tools designed to work with it.

Usually used to mean the specific system and tools you are working with right now.

**machine** Some logic analyzers allow you to set up two measurements at the same time. Each measurement is handled by a different machine. This is represented in the Workspace window by two icons, differentiated by a 1 and a 2 in the upper right-hand corner of the icon. Logic analyzer resources such as pods and trigger terms cannot be shared by the machines.

**markers** Markers are the green and yellow lines in the display that are labeled *x*, *o*, *G1*, and *G2*. Use them to measure time intervals or sample intervals. Markers are assigned to patterns in order to find patterns or track sequences of states in the data. The *x* and *o* markers are local to the immediate display, while *G1* and *G2* are global between time correlated displays.

**master card** In a module, the master card controls the data acquisition or output. The logic analysis system references the module by the slot in which the master card is plugged. For example, a 5-card Agilent Technologies 16555D

---

## Glossary

would be referred to as *Slot C: machine* because the master card is in slot C of the mainframe. The other cards of the module are called *expansion cards*.

**menu bar** The menu bar is located at the top of all windows. Use it to select *File* operations, tool or system *Options*, and tool or system level *Help*.

**message bar** The message bar displays mouse button functions for the window area or field directly beneath the mouse cursor. Use the mouse and message bar together to prompt yourself to functions and shortcuts.

### **module/probe interconnect cable**

The module/probe interconnect cable connects an E5901B emulation module to an E5900B emulation probe. It provides power and a serial connection. A LAN connection is also required to use the emulation probe.

**module** An instrument that uses a single timebase in its operation. Modules can have from one to five cards functioning as a single instrument. When a module has more than one card, system window will show the instrument icon in the slot of the *master card*.

**monitor** When using the Emulation Control Interface, running the monitor means the processor is in debug mode (that is, executing the debug exception) instead of executing the user program.

**panning** The action of moving the waveform along the timebase by varying the delay value in the Delay field. This action allows you to control the portion of acquisition memory that will be displayed on the screen.

**pattern mode** In an oscilloscope, the trigger mode that allows you to set the oscilloscope to trigger on a specified combination of input signal levels.

**pattern terms** Logic analyzer resources that represent single states to be found on labeled sets of bits; for example, an address on the address bus or a status on the status lines.

**period (.)** See *edge terms*, *glitch*, *labels*, and *don't care*.

**pod pair** A group of two pods containing 16 channels each, used to physically connect data and clock signals from the unit under test to the analyzer. Pods are assigned by pairs in the analyzer interface. The number of pod pairs available is determined

---

## Glossary

by the channel width of the instrument.

**pod** See *pod pair*

**point** To point to an item, move the mouse cursor over the item, or position your finger over the item.

**preprocessor** See *analysis probe*.

**primary branch** The primary branch is indicated in the *Trigger sequence step* dialog box as either the *Then find* or *Trigger on* selection. The destination of the primary branch is always the next state in the sequence, except for the Agilent Technologies 16517A. The primary branch has an optional occurrence count field that can be used to count a number of occurrences of the branch condition. See also *secondary branch*.

**probe** A device to connect the various instruments of the logic analysis system to the target system. There are many types of probes and the one you should use depends on the instrument and your data requirements. As a verb, "to probe" means to attach a probe to the target system.

**processor probe** See *emulation probe*.

**range terms** Logic analyzer resources that represent ranges of values to be found on labeled sets of bits. For example, range terms could identify a range of addresses to be found on the address bus or a range of data values to be found on the data bus. In the trigger sequence, range terms are considered to be true when any value within the range occurs.

**relative** Denotes time period or count of states between the current state and the previous state.

**remote display** A remote display is a display other than the one connected to the product hardware. Remote displays must be identified to the network through an address location.

**remote session** A remote session is when you run the logic analyzer using a display that is located away from the product hardware.

**right-click** When using a mouse for a pointing device, to right-click an item, position the cursor over the item, and then quickly press and release the *right mouse button*.

**sample** A data sample is a portion of a *data set*, sometimes just one point. When an instrument samples the target system, it is taking a single



---

## Glossary

measurement as part of its data acquisition cycle.

**Sampling** Use the selections under the logic analyzer Sampling tab to tell the logic analyzer how you want to make measurements, such as State vs. Timing.

**secondary branch** The secondary branch is indicated in the *Trigger sequence step* dialog box as the *Else on* selection. The destination of the secondary branch can be specified as any other active sequence state. See also *primary branch*.

**session** A session begins when you start a *local session* or *remote session* from the session manager, and ends when you select *Exit* from the main window. Exiting a session returns all tools to their initial configurations.

**skew** Skew is the difference in channel delays between measurement channels. Typically, skew between modules is caused by differences in designs of measurement channels, and differences in characteristics of the electronic components within those channels. You should adjust measurement modules to eliminate as much skew as possible so that it does not affect the accuracy of your

measurements.

**state measurement** In a state measurement, the logic analyzer is clocked by a signal from the system under test. Each time the clock signal becomes valid, the analyzer samples data from the system under test. Since the analyzer is clocked by the system, state measurements are *synchronous* with the test system.

**store qualification** Store qualification is only available in a *state measurement*, not *timing measurements*. Store qualification allows you to specify the type of information (all samples, no samples, or selected states) to be stored in memory. Use store qualification to prevent memory from being filled with unwanted activity such as no-ops or wait-loops. To set up store qualification, use the *While storing* field in a logic analyzer trigger sequence dialog.

**subnet mask** A subnet mask blocks out part of an IP address so that 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. Ask your LAN administrator if you need a the subnet mask for your network.

---

# Glossary

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

- 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.
- User-defined symbols - Symbols you create.

Symbols can be used as *pattern* and *range* terms for:

- Searches in the listing display.
- Triggering in logic analyzers and in the source correlation trigger setup.
- Qualifying data in the filter tool and system performance analysis tool set.

**system administrator** The system administrator is a person who manages your system, taking care of such tasks as adding peripheral devices, adding new users, and doing system backup. In general, the system administrator is the person you go to with questions about implementing your software.

**target system** The system under test, which contains the microprocessor you are probing.

**terms** Terms are variables that can be used in trigger sequences. A term can be a single value on a label or set of labels, any value within a range of values on a label or set of labels, or a glitch or edge transition on bits within a label or set of labels.

**TIM** A TIM (Target Interface Module) makes connections between the cable from the emulation module or emulation probe and the cable to the debug port on the system under test.

**time-correlated** Time correlated measurements are measurements involving more than one instrument in which all instruments have a common time or trigger reference.

**timer terms** Logic analyzer resources that are used to measure the time the trigger sequence remains within one sequence step, or a set of sequence steps. Timers can be used to detect when a condition lasts too long or not long enough. They can be used to measure pulse duration, or duration of a wait loop. A single timer term can be used to delay trigger until a period of time after detection of a significant event.

---

# Glossary

**timing measurement** In a timing measurement, the logic analyzer samples data at regular intervals according to a clock signal internal to the timing analyzer. Since the analyzer is clocked by a signal that is not related to the system under test, timing measurements capture traces of electrical activity over time. These measurements are *asynchronous* with the test system.

**tool icon** Tool icons that appear in the workspace are representations of the hardware and software tools selected from the toolbox. If they are placed directly over a current measurement, the tools automatically connect to that measurement. If they are placed on an open area of the main window, you must connect them to a measurement using the mouse.

**toolbox** The Toolbox is located on the left side of the main window. It is used to display the available hardware and software tools. As you add new tools to your system, their icons will appear in the Toolbox.

**tools** A tool is a stand-alone piece of functionality. A tool can be an instrument that acquires data, a display for viewing data, or a post-processing analysis helper. Tools are represented as icons in the main window of the interface.

**trace** See *acquisition*.

**trigger sequence** A trigger sequence is a sequence of events that you specify. The logic analyzer compares this sequence with the samples it is collecting to determine when to *trigger*.

**trigger specification** A trigger specification is a set of conditions that must be true before the instrument triggers.

**trigger** Trigger is an event that occurs immediately after the instrument recognizes a match between the incoming data and the trigger specification. Once trigger occurs, the instrument completes its *acquisition*, including any store qualification that may be specified.

**workspace** The workspace is the large area under the message bar and to the right of the toolbox. The workspace is where you place the different instrument, display, and analysis tools. Once in the workspace, the tool icons graphically represent a complete picture of the measurements.

**zooming** In the oscilloscope or timing analyzer, to expand and contract the waveform along the time base by varying the value in the s/Div

field. This action allows you to select specific portions of a particular waveform in acquisition memory that will be displayed on the screen. You can view any portion of the waveform record in acquisition memory.

## A

absolute, time mode, 14  
add, 59  
address base in inverse assembled listings, 18  
align display, inverse assembly, 21  
Align to x Byte option, 82  
Align to x Byte option for symbols, 82  
Alignment byte, inverse assembler, 19  
alignment, when searching for source code in Listing, 38  
ascii, 14  
ASCII format, 70  
ASCII format symbols, 70, 72, 73, 74  
ASCII symbol file, 73  
assembler, loading and unloading inverse assembler files, 22

## B

beyond prefetch depth, 82  
binary, 14  
block instruction fetches, 38  
bookmarks, 50, 51, 52  
break on trigger, enable/disable in Source Viewer, 33  
Browse Source tab (in Source Viewer), 35  
browser dialog, 78  
browser, symbol search, 78  
browsing, 81  
browsing the symbol database, 81

## C

cancel, 56  
captured data associated with source line, 28  
captured data, stepping through by source lines, 26  
clearing the workspace, 62

code (source), searching for text in, 35  
code directory search list (Source Viewer), 36  
code, assigning address offsets, 68  
COFF symbol reader options, 76  
color, editing in waveforms, 88  
color, label column, 13  
column, 13  
column width, adjust, 12  
comments, 74  
comments on display printouts, 86, 87  
connecting tool input and output ports, 61  
correlated Listing of Source Viewer, 39  
creating a file, 70  
creating ASCII symbol files, 70

## D

data associated with source line, 28  
data displayed in symbolic form, 65  
data source of Source Viewer, 39  
data, stepping through by source lines, 26  
decimal, 14  
define, 43, 44  
delete, 59  
demand driven data, 58  
directory search list (Source Viewer), 36  
disable emulator break on trigger (in Source Viewer), 33  
disable/enable run status window, 57  
display options (in Source Viewer), 38  
display reference, 54  
display window pops up when run completes, 85

displaying symbols to represent data, 65

## E

ELF symbol reader options, 76  
ELF/stabs file format, 69  
emulator break on trigger, enable/disable in Source Viewer, 33  
enable emulator break on trigger (in Source Viewer), 33  
error messages, trigger setup (in Source Viewer), 34  
everything, tracing, 30  
example, 70, 81, 82  
example measurements, 45

## F

fetches, multi-byte instruction, 38  
file formats for ASCII symbols, 70  
file versions, 66  
files, 66  
files, loading and unloading inverse-assembler files, 22  
files, loading user-defined symbol files, 84  
filter options, inverse assembler, 17  
find, 43  
finding labels, 14  
finding the symbol you want, 81  
font size, Source Viewer display option, 38  
fonts, size, 11  
function, tracing about, 32  
functions, 72

## G

G1, G2 markers, viewing source code at, 27  
go to state, 41  
goto, 42, 51, 52  
Goto in Listing alignment, 38

Goto In Listing tab (in Source Viewer), 27  
group run, 56

## H

halt (target system), tracing up to, 31  
hardcopy, 55  
help, 60  
help, symbols, 64  
help, topic index, 54  
hex, 14  
high-level source code, viewing, 24

## I

IEEE-695 file format, 69  
in ASCII format, 72, 73, 74  
in symbol browser, 81  
independent run, 56  
Info tab (in Source Viewer), 39  
instruction fetches, multi-byte, 38  
Intermodule window, for emulator break, 33  
Invasm, 17  
Invasm, using, 45  
Inverse Assemblers, in the listing display, 17  
inverse assembly, 45  
inverse assembly, alignment of display, 21  
inverse-assembler files, loading and unloading, 22

## L

label, name width, 10  
labels, 10, 11, 12, 13  
labels, listing display, 10  
license (for source correlation tool set), obtaining, 25  
line #, 14  
line number, 14  
line number, tracing about, 32

line numbers, 73  
line numbers, Source Viewer display option, 38  
Listing display, alignment when searching for source code, 38  
listing, print to file, 47  
Listing, Source Viewer correlated with, 39  
load symbols (in Source Viewer), 36  
loading, 66  
loading and unloading inverse-assembler files, 22  
loading files including symbols, 66  
loading listing configurations, 40  
loading object file symbols, 66  
loading user-defined symbol files, 84  
location, 42  
lock column, 13

## M

mark, 50, 51, 52  
markers, viewing source code at, 27  
masking off addresses of symbols, 82  
measurement examples, 45  
measurement, run options, 56  
messages, trigger setup (in Source Viewer), 34  
moving tools around the workspace, 63  
multi-byte instruction fetches, 38

## N

name, Source Viewer display option, 38  
numeric base, setting, 14

## O

object file symbol browser, 78

object file symbol files, 66  
object file symbols, 78, 81  
octal, 14  
odd-numbered addresses, 82  
odd-numbered addresses represented by symbols, 82  
offset, 82  
offset addresses, assigning, 68  
Offset By option of the symbol browser, 82  
OMF96 file format, 69  
OMFx86 file format, 69  
options, 43  
options, run, 56  
options, Source Viewer display, 38

## P

pattern, 42, 43, 44  
Pattern field, 81  
patterns, 42  
place, 50, 51, 52  
polarity of values in listing, 14  
prefetch, 82  
print to file, 47  
printer, 55  
printing windows, configurations, 55  
problems, trigger setup (in Source Viewer), 34  
program code, browsing in Source Viewer, 35  
program directory search list (Source Viewer), 36

## Q

qualify, 43

## R

readers.ini file, 75  
record headers, 70  
reference points (Trigger, G1, G2) viewing source code at, 27

- reference, display, 54
  - relative, time mode, 14
  - reload symbols (in Source Viewer), 36
  - relocating sections of code, 68
  - repetitive, 56
  - repetitive data display, 58
  - repositioning tools, 63
  - Resource in use by other machine (Source Viewer message), 34
  - results, 81
  - run, 56
  - run options, 57
  - run status, checking, 57
  - run status, disable window, 57
- S**
- saving listing configurations, 40
  - Scroll Files field, 78
  - search, 42, 43, 44
  - search list, source code directory, 36
  - Search Pattern field, 81
  - search string in source code, 35
  - search, pattern, 43
  - searching, 42
  - searching for state or pattern locations, 41
  - searching the symbol database, 81
  - sections, 72
  - simple example, 70
  - simple form, 70
  - single, 56
  - source code at markers or trigger, 27
  - source code directory search list (Source Viewer), 36
  - source code, alignment when searching in Listing, 38
  - source code, browsing in Source Viewer, 35
  - source code, searching for text in, 35
  - source file displayed in Source Viewer, 39
  - source line numbers, 73
  - source line, going to captured data of, 28
  - source line, trigger after, about, or before, 29
  - source lines, stepping through captured data by, 26
  - Source Viewer, 24
  - Source Viewer display options, 38
  - Source Viewer trace setup, modifying, 32
  - Source Viewer, opening, 25
  - Stabs symbol reader options, 76
  - start address, 72
  - state location, go to, 41
  - status, 57
  - Step Source tab (in Source Viewer), 26
  - stop, 56
  - stop, trace until, 31
  - subdirectories, adding or removing from search list, 36
  - symbol demangling, 75
  - symbol file formats, 69
  - symbol file versions, 66
  - symbol selector, 32
  - symbol selector dialog, 80, 81
  - symbol type, 78
  - symbol types, 78
  - Symbol width, in inverse assemblers, 19
  - symbols, 14, 81
  - Symbols tab (in Source Viewer), 36
  - symbols, displaying to represent data, 65
  - symbols, how they are used in the logic analyzer, 78
  - symbols, loading object file symbols, 66
  - symbols, loading user-defined symbol files, 84
  - symbols, outside defined sections, 75
  - symbols, setting up, 66
  - symbols, types and use, 64
  - symbols, user-defined details, 83
  - synchronize the inverse assembler, 21
- T**
- tab width, Source Viewer display option, 38
  - tab, symbols, 64
  - tabs, turning on or off in display window, 86
  - target system halt, tracing up to, 31
  - term, 44
  - terms, patterns, 43
  - Text Search tab (in Source Viewer), 35
  - TI COFF file format, 69
  - time modes, relative or absolute, 14
  - tools, adding and deleting, 59
  - tools, connecting, 61
  - tools, repositioning, 63
  - trace about a variable, function, or line number, 32
  - trace after, about, or before source line, 29
  - trace everything, 30
  - trace setup, modifying Source Viewer, 32
  - trace until stop, 31
  - trigger after, about, or before source line, 29
  - trigger setup problems (in Source Viewer), 34
  - trigger setup, modifying Source Viewer, 32

Trigger specification exceeds  
  resources available (Source  
  Viewer message), 34  
trigger term, 78, 82  
trigger, enable/disable emulator  
  break on (in Source Viewer),  
  33  
trigger, reference, 49  
trigger, viewing source code at, 27  
triggering beyond, 82  
triggering on a symbol, 78, 81, 82  
triggers, setting up based on source  
  code, 24  
twos', 14

## U

Unable to communicate with  
  machine (Source Viewer  
  message), 34  
until stop, tracing, 31  
User Symbols, 78  
user-defined symbol files, loading,  
  84  
user-defined symbols, details, 83

## V

variable, tracing about, 32  
variables, 74  
versions, 66  
versions of symbol files, 66

## W

wildcard characters, 81  
window, printing, 55  
workspace, 55  
workspace, clearing the, 62