Help Volume

Agilent Technologies 16760A Logic Analyzer

Agilent Technologies 16760A Logic Analyzer

The Agilent Technologies 16760A 1500 Mb/s State/800 MHz Timing logic analyzer offers 64M deep memory with up to 170 channels on a single time base (5 cards, 34 channels per card). Differential probing captures input signals as low as 200 mV p-p.

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

Glossary (see page 263)

Contents

Agilent Technologies 16760A Logic Analyzer

Contents

3 Using the Logic Analyzer in Timing or State Mode

Contents

4 Using the Logic Analyzer in Eye Scan Mode

Contents

5 Reference

Contents

Contents

6 Concepts

Contents

Contents

1

Getting Started

After you have connected the logic analyzer probes to your device under test (see "Step 1. Connect logic analyzer to the device under test" on page 15), all measurements will have the following initial steps:
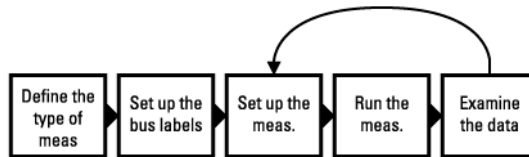
- "Step 2. Choose the sampling mode" on page 16

- "Step 3. Format labels for the probed signals" on page 19

In the timing (asynchronous) or state (synchronous) sampling modes, measurements will have these steps:

- "Step 4. Define the trigger condition" on page 22

- "Step 5. Run the measurement" on page 23

- "Step 6. Display the captured data" on page 24

In the eye scan sampling mode, measurements will have these steps:

- "Step 4. Select channels for the eye scan measurement" on page 26

- "Step 5. Set the eye scan range and resolution" on page 27

- "Step 6. Run the eye scan measurement" on page 28

- "Step 7. Display the captured eye scan data" on page 29

If you have previously saved a logic analyzer setup to a configuration file, or if configuration files are included with an analysis probe, you can load the configuration file to set up the logic analyzer and the measurement.

Once you have made a logic analyzer measurement, the measurement can be refined by repeating the measurement set up, run, and display steps.

Next: "Step 1. Connect logic analyzer to the device under test" on page 15

Probing and Sampling Mode Selection Steps

You will always take the following steps regardless of the sampling mode you plan to use.

- "Step 1. Connect logic analyzer to the device under test" on page 15

- "Step 2. Choose the sampling mode" on page 16

- "Step 3. Format labels for the probed signals" on page 19

Step 1. Connect logic analyzer to the device under test

Before you begin setting up the logic analyzer for a measurement, you need to physically connect the logic analyzer to your device under test.

There are four probing options available to connect your logic analyzer to the device under test:

- Single-ended probes with Samtec connectors (see page 35).

- Differential probes with Samtec connectors (see page 37).

- Single-ended probes with MICTOR connectors (see page 39).

- Use an *analysis probe* to connect to microprocessors and standard buses. When using an analysis probe, the Setup Assistant guides you through the connection and setup process for your particular logic analyzer and analysis probe.

Next: "Step 2. Choose the sampling mode" on page 16

Step 2. Choose the sampling mode
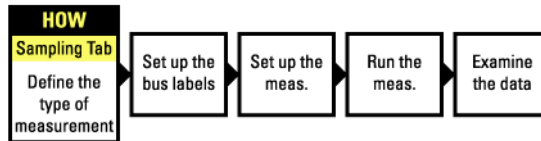
Choose timing, state, or eye mode
Set sampling mode configuration
Set up sample period or sampling clock

There are three logic analyzer sampling modes to choose from: *timing mode*, *state mode*, and *eye scan mode*.

In *timing mode*, the logic analyzer samples asynchronously, based on an internally-generated sampling clock.

In *state mode*, the logic analyzer samples synchronously, based on a sampling clock signal from the device under test. Typically, the signal used for sampling in state mode is a state machine or microprocessor clock signal.

In *eye scan mode*, the logic analyzer samples small windows of time and voltage on data channels around a clock signal from the device under test. The resulting eye diagrams let you validate and characterize the data valid windows of signals latched on the clock.

To choose the sampling mode
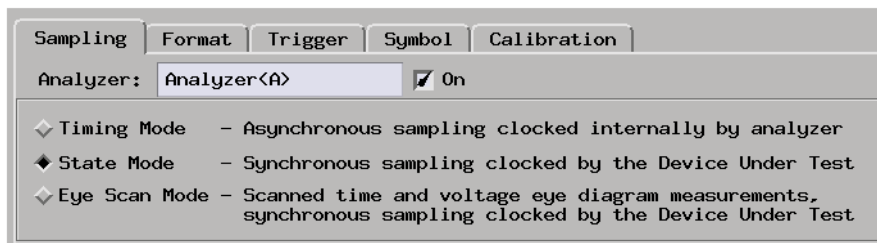
1. In the Sampling tab, choose *Timing Mode*, *State Mode*, or *Eye Scan Mode*.

If you chose Timing Mode

```
┌─Timing Mode Controls──────────────────────────────────┐
│  ┌──────────────────────────────────────────────┬───┐ │
│  │ 800 MHz / 64M Sample Conventional            │ ± │ │
│  └──────────────────────────────────────────────┴───┘ │
│                                                        │
│  Trigger Position┌─────────────┬───┐                   │
│                  │ Center      │ ± │                   │
│                  └─────────────┴───┘                   │
│                                                        │
│  Acquisition Depth┌─────┬───┐                          │
│                   │ 32M │ ± │                          │
│                   └─────┴───┘                          │
│                                                        │
│  Sample Period: ┌─────────┐                            │
│                 │ 1.25 ns │                            │
│                 └─────────┘                            │
└────────────────────────────────────────────────────────┘
```
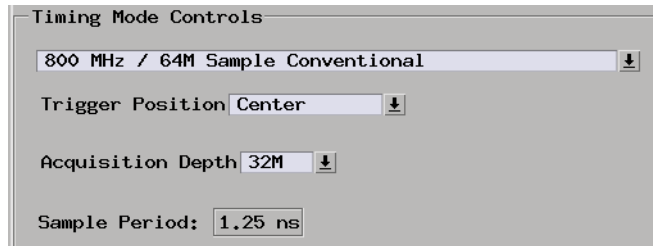
1. Select the timing analyzer conventional/transitional configuration.

   In the transitional timing configuration, the logic analyzer can capture a greater period of execution because only transitions are stored in memory.

2. If you chose the transitional timing configuration, set the sample period.

   To capture signal level changes reliably, the sample period should be less than half (many engineers prefer one-fourth) of the period of the fastest signal you want to measure.

If you chose State Mode

```
┌─State Mode Controls──────────────────────────────────────┐
│  ┌──────────────────────────┬───┐                        │
│  │ 800 Mb/s /  64M State    │ ± │  Trigger Position┌──────│
│  └──────────────────────────┴───┘                  │Center│
│  Acquisition Depth┌───────┬───┐                          │
│                   │ 64M   │ ± │                          │
│                   └───────┴───┘                          │
│  ┌─Clock Setup─────────────────────────────────────┐    │
│  │ Mode: ┌───────────────┬───┐    ┌──────┬────┐     │    │
│  │       │ Periodic clock│ ± │    │  Pod │ A1 │     │    │
│  │       └───────────────┴───┘    ├──────┼────┤     │    │
│  │ ┌──────────────────────┐       │ Clock│ J  │     │    │
│  │ │ Sampling Positions...│       ├──────┼────┤     │    │
│  │ └──────────────────────┘       │Activity│ ↕ │    │    │
│  │                                ├──────┼────┤     │    │
│  │                                │Master│ ⨏  │=> J↑│    │
│  │                                └──────┴────┘     │    │
│  └─────────────────────────────────────────────────┘    │
└──────────────────────────────────────────────────────────┘
```
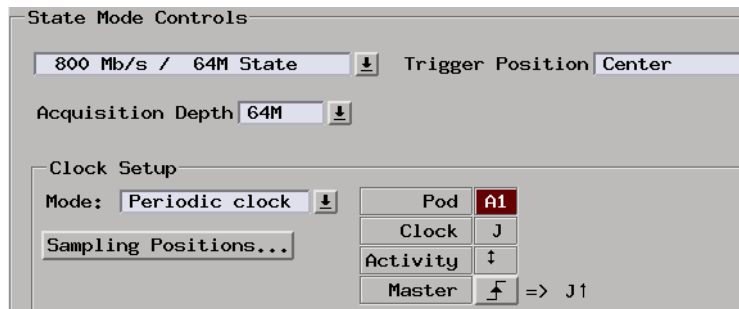
1. Select the state analyzer speed configuration.

   There are trade-offs between high-speed sampling, the number of available channels, triggering capabilities, and other logic analyzer characteristics. For example, in the 1250 Mb/s configuration, a periodic clock input signal is required.

2. In the Clock Setup, specify which clock signal edges from the device under test will be used as the sampling clock.

3. Specify the sampling position. Select the *Sampling Positions...* button, then select the *Run Eye Finder* button to locate the data valid window in relation to the sampling clock, and automatically set the sampling position of the logic analyzer.

   See Also "To automatically adjust sampling positions" on page 49

In either Timing Mode or State Mode

1. Specify the trigger position.

   The trigger is the event in the device under test that you want to capture data around.

   Specify whether you want to look at data after the trigger (Start), before and after the trigger (Center), before the trigger (End), or use a percentage of the logic analyzer's memory for data after the trigger (User Defined).

2. Set the acquisition memory depth.

   If you need less data and want measurements to run faster, you can limit the amount of trace memory that is filled with samples.

If you chose Eye Scan Mode
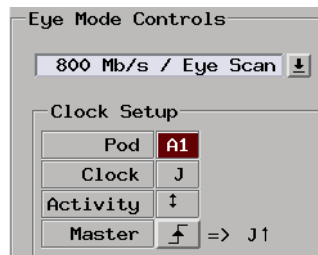
1. Select the eye scan mode speed configuration.

   There are trade-offs between high-speed sampling, the number of available channels, and other logic analyzer characteristics.

2. In the Clock Setup, specify which clock signal edges from the device under test will be used as the reference clock for the eye scan measurement.

Next: "Step 3. Format labels for the probed signals" on page 19

Step 3. Format labels for the probed signals
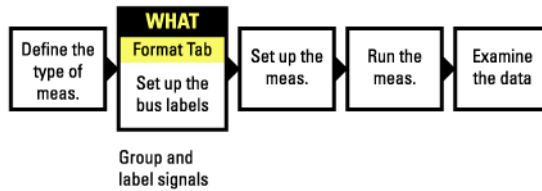
When a logic analyzer probes hundreds of signals in a device under test, you need to be able to give those channels more meaningful names than "pod 1, channel 1".

The Format tab is mainly used for assigning bus and signal names (from the device under test) to logic analyzer channels. These names are called *labels*. Labels are also used when setting up triggers and displaying captured data.

The Format tab also lets you do things like assign pods to the logic analyzer and specify the logic analyzer threshold voltage.

The Format tab has activity indicators that show whether the signal a channel is probing is above the threshold voltage (high), below the threshold voltage (low), or transitioning.
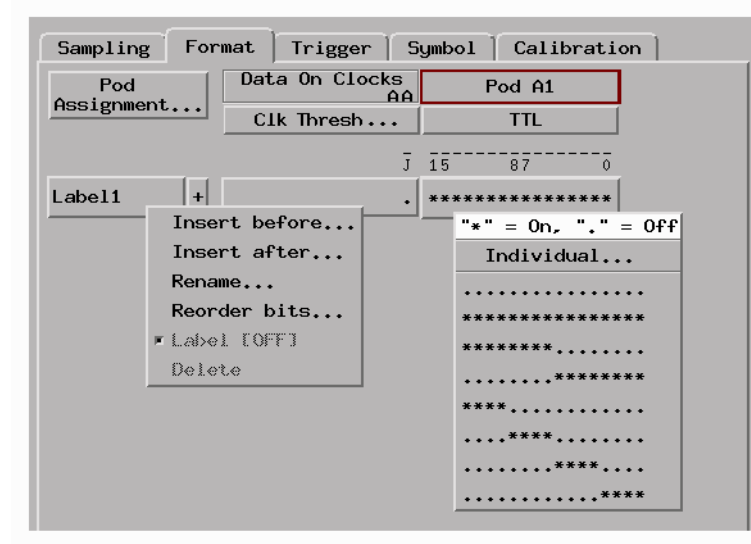
To assign pods to the logic analyzer

1. In the Format tab, select the *Pod Assignment* button.

2. In the Pod Assignment dialog, drag a pod to the appropriate logic analyzer.

3. Select the Close button.

To specify threshold voltages

The *threshold voltage* is the voltage level that a signal must cross before the logic analyzer recognizes a change in logic levels.

1. In the Format tab, select the button under the pod name.

2. In the Pod threshold dialog, select a Standard, External Ref, Differential, or a User Defined threshold voltage.

3. Select the Close button.

4. Select the *Clk Thresh* button.

5. In the Clock Thresholds dialog, select the button of the clock whose threshold you wish to set.

6. In the J, K, etc., threshold dialog, select a Standard, Differential, or a User

Defined threshold voltage.

7. Select the Close button.

8. Select the Close button.

To assign names to logic analyzer channels

1. Select a label button, and either:

   • Choose the *Rename* command, enter the label name, and select the OK button.

   • Or, choose the *Insert before* or *Insert after* command, enter the label name, and select the OK button.

2. In the label row, select the button of the pod that contains the channels you want to assign.

3. Either choose one of the standard channel assignments--dots (.) mean the channel is unassigned, asterisks (*) mean the channel is assigned--or choose *Individual*.

   If you chose *Individual*:

   a. In the "label - pod" dialog, select the channels you want to assign/unassign.

   b. Select the OK button.

Next:

• If you are using the logic analyzer in timing mode or in state mode, go to "Step 4. Define the trigger condition" on page 22.

• If you are using the logic analyzer in eye scan mode, go to "Step 4. Select channels for the eye scan measurement" on page 26.

Timing Mode or State Mode Steps

When you have selected the timing or state sampling modes, you need to perform the following steps.

- "Step 4. Define the trigger condition" on page 22

- "Step 5. Run the measurement" on page 23

- "Step 6. Display the captured data" on page 24

Step 4. Define the trigger condition

Set up trigger sequence
Define events
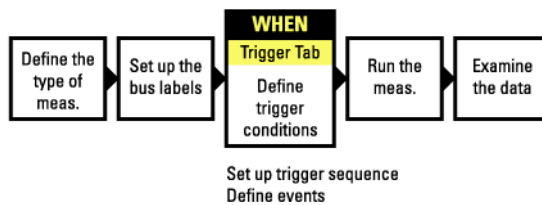
The trigger is the event in the device under test that you want to capture data around.

1. In the Trigger tab, and in the Trigger Functions subtab, choose the type of trigger you want to specify, and select the *Replace* button.
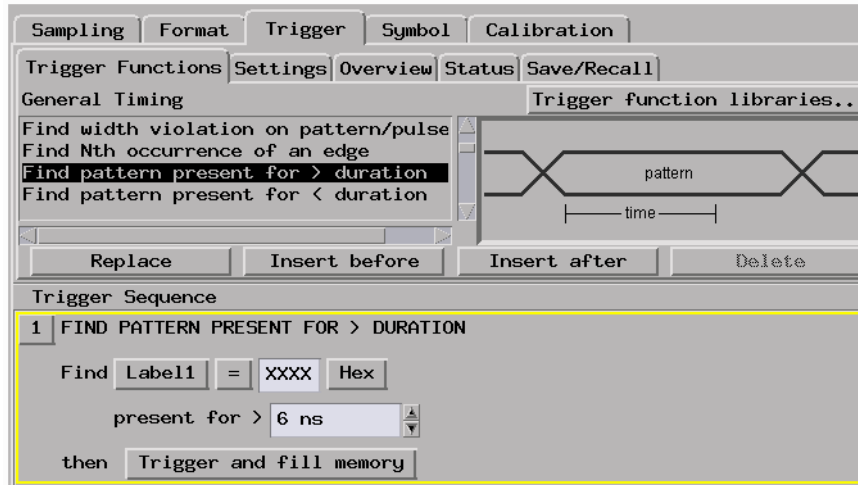
2. In the Trigger Sequence portion of the Trigger tab, select the buttons to define the label values and/or other conditions you want to trigger on.

Step 5. Run the measurement
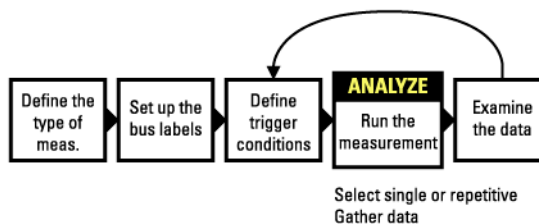
Once the trigger condition has been defined, you can run the measurement.

1. Select the Run Single button .

   When you run a measurement, the Stop button becomes available while the logic analyzer looks for the trigger condition.

Logic analyzers with deep acquisition memory take a noticeable amount of time to complete a run; however, messages like "Waiting in level 1" may indicate you need to stop the measurement and refine the trigger condition.

When the trigger condition is found, logic analyzer acquisition memory is filled, the captured data is processed to the display tools, and the Run Single button becomes available again.

Next: "Step 6. Display the captured data" on page 24

Step 6. Display the captured data
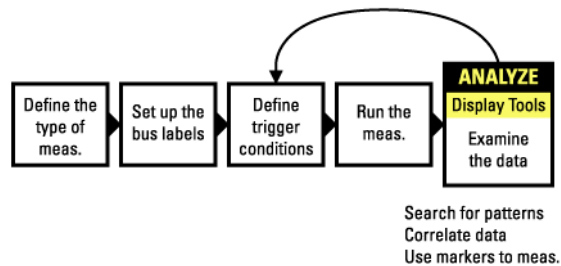
Once you have run a measurement and filled the logic analyzer's acquisition memory with captured data, you can display it with one of the display tools.

To open Waveform or Listing displays

Waveform displays are typically used when data is captured with the timing sampling mode, and Listing displays are used when data is captured with the state sampling mode.

1. From the Window menu, select your logic analyzer and choose the *Waveform* or *Listing* command.
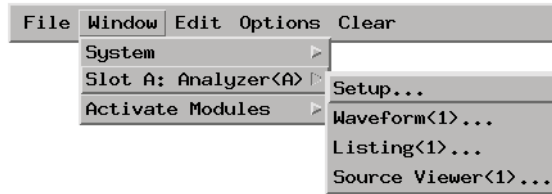
To add display tools via the Workspace window

1. Select the Workspace button (or from the Window menu, select System and Workspace).

2. In the Workspace window, scroll down to the Display portion of the tool icon list.
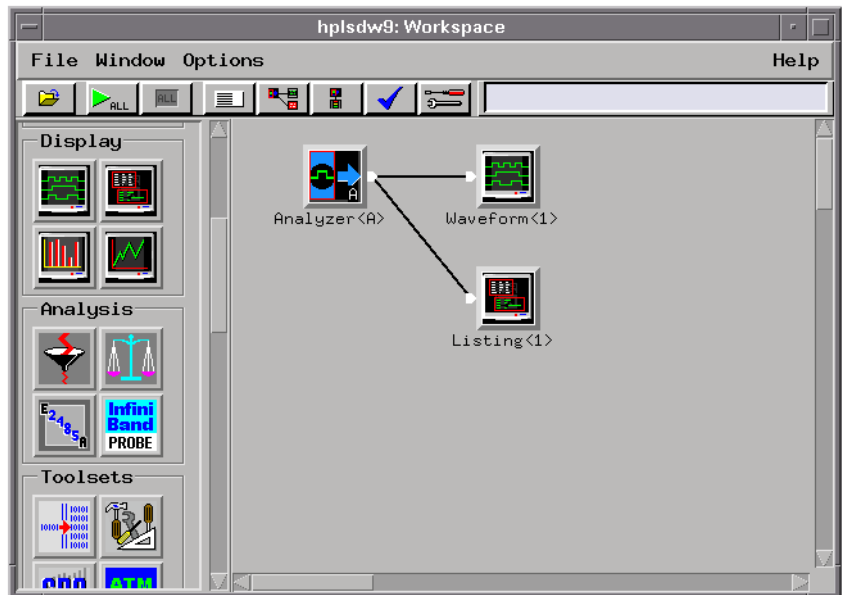
3. Drag the display tool icon and drop it on the analyzer icon.

4. To open the display tool, select its icon and choose the *Display* command.

Next: "For More Information..." on page 31

Eye Scan Mode Steps

When you have selected the eye scan sampling mode, you need to perform the following steps.

Step 4. Select channels for the eye scan measurement

Select labels to be
included in measurement
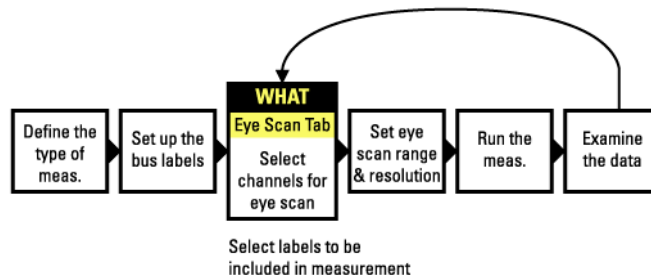
1. In the Eye Scan tab, Labels subtab, select the channels you want in the eye scan measurement.
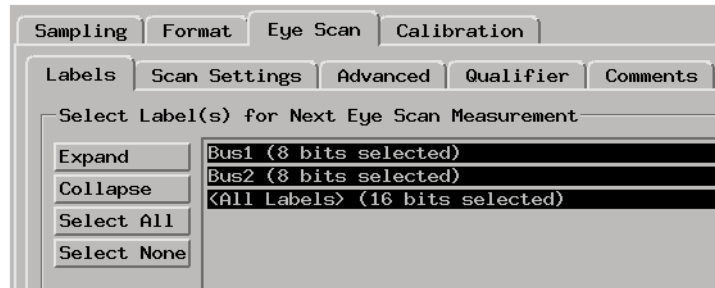
Step 5. Set the eye scan range and resolution
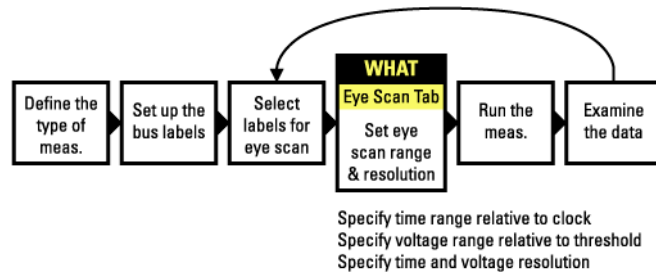
1. In the Eye Scan tab, Scan Settings subtab, select the settings for the eye
    scan measurement.
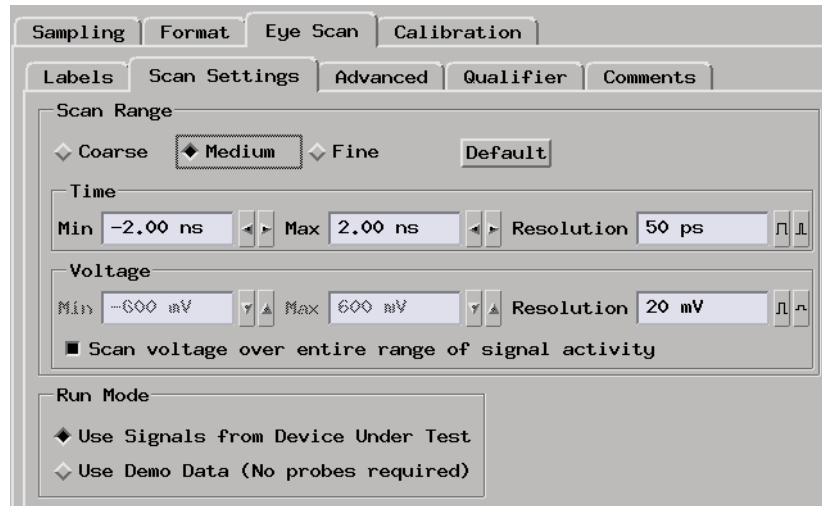
These settings define the number and size of the time and voltage windows used in the eye scan.

Measurements using the coarse settings run faster because there are fewer time and voltage windows to scan, but the resulting eye diagrams have less resolution.

Measurements using the fine settings result in eye diagrams with better resolution, but the measurements take longer to run because there are more time and voltage windows to scan.

Step 6. Run the eye scan measurement
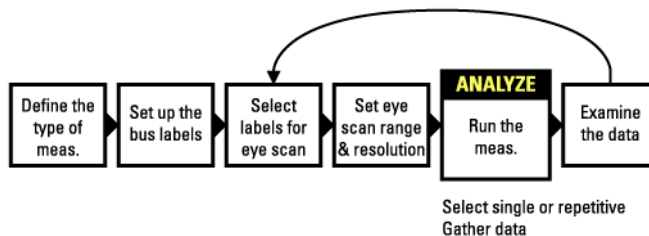
Once the eye scan settings have been selected, you can run the measurement.

1. Select the Run Single button 　.

    The Eye Scan display window opens, and the captured measurement data begins to appear.

    While the eye scan measurement runs, the Stop button becomes available.

    The estimated time of the measurement is shown in the status field.

Step 7. Display the captured eye scan data

Scale the display
Change display options
Use measurement tools
Display measurement info
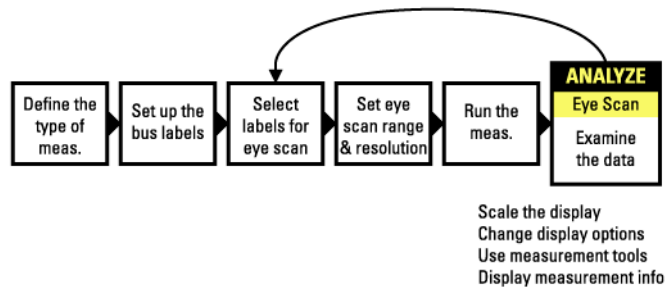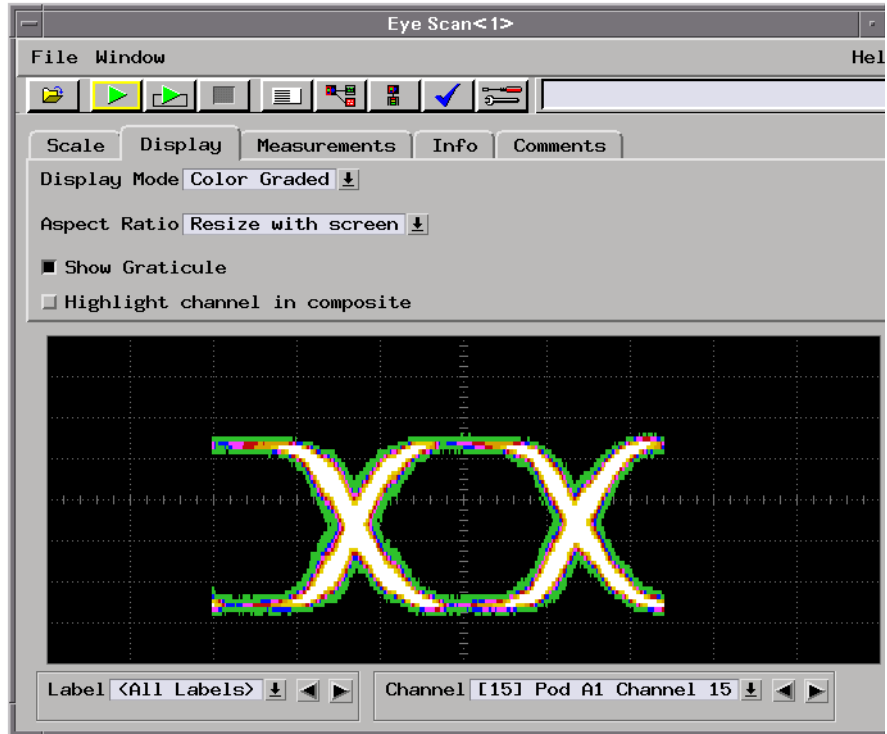
In the Eye Scan display window, use the following subtabs:

- *Scale* to zoom in or out on the captured data.

- *Display* to change the display options.

- *Measurements* to use tools for displaying useful information about the data.

- *Info* to display textual information about the captured measurement data.

- *Comments* to save your comments with the data.

Next: "For More Information..." on page 31

For More Information...

On connecting the logic analyzer:

- "Probing the Device Under Test" on page 35

- Setup Assistant (see the *Setup Assistant* help volume) (when using analysis probes).

- *Logic Analysis System and Measurement Modules Installation Guide* for probe pinout and circuit diagrams.

On choosing the sampling mode:

- "Choosing the Sampling Mode" on page 43

- "The Sampling Tab" on page 155

On formatting labels for probed signals:

- "Formatting Labels for Logic Analyzer Probes" on page 57

- "The Format Tab" on page 174

On setting up measurements:

- "Understanding Logic Analyzer Triggering" on page 240

- "Understanding Eye Scan Measurements" on page 259

- "Setting Up Triggers and Running Measurements" on page 69

- "Setting Up and Running Eye Scan Measurements" on page 119

- "The Trigger Tab" on page 176

- "The Eye Scan Tab" on page 204

On running measurements:

- "Running Measurements" on page 91

- "To run an eye scan measurement" on page 121

On displaying captured data:

- "Displaying Captured Data" on page 94

- "Displaying Captured Eye Scan Data" on page 133

- Using the Waveform Display Tool (see the *Waveform Display Tool* help volume)

- Using the Listing Display Tool (see the *Listing Display Tool* help volume)

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

- Using the Chart Display Tool (see the *Chart Display Tool* help volume)

- Using the Distribution Display Tool (see the *Distribution Display Tool* help volume)

- Using the Compare Analysis Tool (see the *Compare Tool* help volume)

2

Probing and Selecting the Sampling Mode

- "Probing the Device Under Test" on page 35

Probing the Device Under Test

When using the 16760A logic analyzer, there are four probing options available:

- "Using the E5378A Single-Ended Probe" on page 35
- "Using the E5379A Differential Probe" on page 37
- "Using the E5380A Mictor-Compatible Probe" on page 39
- "Using the E5382A Single-ended Flying Lead Probe Set" on page 40
- "Using an Analysis Probe" on page 41

Using the E5378A Single-Ended Probe

The E5378A single-ended probe has 34-channels and is capable of capturing data at rates up to 1500 Mb/s. The probe has the following inputs:

- 32 single-ended data inputs, in two groups (pods) of 16.
- Two differential clock inputs. Either or both clock inputs can be acquired as data inputs if not used as a clock.
- Two data threshold reference inputs, one for each pod (group of 16 data inputs).

This probe matches two 17-channel probe cables to a single-ended 100-pin Samtec connector.

Mechanical Considerations

Each E5379A probe requires a mating connector and support shroud built into the device under test.

You can order a probing connector kit, which contains five mating connectors and five support shrouds, using the Agilent part numbers:

- 16760-68702 for PC board thicknesses up to 0.062 in.
- 16760-68703 for PC board thicknesses up to 0.120 in.

You can order mating connectors separately using the Agilent part number:

- 1253-3620 (or Samtec #ASP-65067-01)

You can order support shrouds separately using the Agilent part numbers:

- 16760-02302 for PC board thicknesses up to 0.062 in.

- 16760-02303 for PC board thicknesses up to 0.120 in.

Electrical Considerations

Data threshold reference inputs

The E5378A single-ended probe has two inputs for threshold reference voltages for the data inputs. One input is for the even pod and the other is for the odd pod. The threshold inputs (pins 87 and 88) may be either:

Connected to a DC threshold reference voltage. In this case, the logic analyzer will use the threshold reference voltage to determine when the signal is high or low.

Or:

Grounded or left unconnected. In this case, you need to set the logic threshold voltage in the user interface.

The advantages of supplying a threshold voltage via the threshold input on the probe are:

- A threshold voltage supplied from the device under test will typically track changes in supply voltage, temperature, etc.

- A threshold voltage supplied from the device under test is typically the same threshold that the device's logic uses to evaluate the signals. Therefore, the data captured by the logic analyzer will be the same as the data interpreted by the device under test.

Clock Input

The clock input on the E5378A probe can be a differential signal or a single-ended signal.

If the clock input is a differential signal, select the "differential" option in the clock threshold user interface.

If the clock input is a single-ended signal, either ground the negative clock input and adjust the clock threshold voltage in the user interface, or connect the negative clock input to the DC clock threshold reference voltage.

In the user interface, the clock input threshold voltage adjustment is separate from the data input threshold voltage adjustments.

See Also "To set pod threshold voltages" on page 58

"To set clock threshold voltages" on page 59

For complete information, including mechanical considerations such as pin-out and footprint and electrical considerations such as circuit board design best practices, refer to the *Agilent Technologies E5378A, E5379A, and E5380A Probes for the 16760A Logic Analyzer* user's guide that comes with the probes.

Using the E5379A Differential Probe

The E5379A differential probe has 17 channels and is capable of capturing data at rates up to 1500 Mb/s. The probe has the following inputs:

- 16 differential data inputs.

- One differential clock input. The clock input can also be used as a data input.

This probe matches one 17-channel probe cable to a differential 100-pin Samtec connector. Two E5379A probes are required to support the inputs on one 16760A logic analyzer card.

Mechanical Considerations Each E5379A probe requires a mating connector and support shroud built into the device under test.

You can order a probing connector kit, which contains five mating connectors and five support shrouds, using the Agilent part numbers:

- 16760-68702 for PC board thicknesses up to 0.062 in.

- 16760-68703 for PC board thicknesses up to 0.120 in.

You can order mating connectors separately using the Agilent part number:

- 1253-3620 (or Samtec #ASP-65067-01)

You can order support shrouds separately using the Agilent part numbers:

- 16760-02302 for PC board thicknesses up to 0.062 in.

- 16760-02303 for PC board thicknesses up to 0.120 in.

Electrical Considerations

Data inputs

The E5379A differential probe can capture data on differential signals or single-ended signals.

When capturing data on differential signals, the logic analyzer will determine high and low states based on the crossover of the data and negative data inputs.

When capturing data on single-ended signals, either ground the negative data inputs and adjust the threshold voltage in the user interface, or connect the negative data inputs to the DC threshold reference voltage.

Clock input

The clock input on the E5379A differential probe can also be a differential signal or a single-ended signal, in the same way as described for the data inputs above. The clock input has a separate, independent threshold adjustment.

See Also

"To set pod threshold voltages" on page 58

"To set clock threshold voltages" on page 59

For complete information, including mechanical considerations such as pin-out and footprint and electrical considerations such as circuit board design best practices, refer to the *Agilent Technologies E5378A, E5379A, and E5380A Probes for the 16760A Logic Analyzer* user's guide that comes with the probes.

Using the E5380A Mictor-Compatible Probe

The E5380A MICTOR-compatible probe has a MICTOR connector end. If you have a device under test with connectors designed for the Agilent E5346A high-density probe adapter, you can also use the E5380A probe.

However, the lower bandwidth of the E5380A probe limits the maximum state speed of the logic analyzer to 600 Mbits/second.

The minimum input signal amplitude required by the E5380A probe is 300 mV.

The probe matches two 17-channel probe cables to a single-ended 38 pin MICTOR connector.

Mechanical Considerations

Each E5380A probe requires a MICTOR connector and support shroud built into the device under test.

You can order a probing connector kit, which contains five MICTOR connectors and five support shrouds, using the Agilent part numbers:

- E5346-68701 for PC board thicknesses up to 0.062 in.

- E5346-68702 for PC board thicknesses from 0.062 to 0.125 in.

You can order MICTOR connectors separately using the Agilent part number:

- 1252-7431 (or AMP part #2-767004-2)

You can order support shrouds separately using the Agilent part numbers:

- E5346-44701 for PC board thicknesses up to 0.062 in.

- E5346-44704 for PC board thicknesses from 0.062 to 0.125 in.

- E5346-44703 for PC board thicknesses from 0.125 to 0.70 in.

Electrical Considerations

All inputs on the E5380A MICTOR-compatible probe are single-ended. The E5380A probe does not have a threshold reference voltage input. When using the E5380A probe, the logic threshold voltage is adjusted in the user interface.

The clock input on the E5380A probe is single-ended. The clock threshold voltage may be adjusted independently of the data threshold voltages.

See Also

"To set pod threshold voltages" on page 58

"To set clock threshold voltages" on page 59

For complete information, including mechanical considerations such as pin-out and footprint and electrical considerations such as circuit board design best practices, refer to the *Agilent Technologies E5378A, E5379A, and E5380A Probes for the 16760A Logic Analyzer* user's guide that comes with the probes.

Using the E5382A Single-ended Flying Lead Probe Set

The E5382A is a 17-channel single-ended flying lead probe set. The E5382A lets you acquire signals from randomly located points in your target system. Two E5382As are required to support 34 channel logic analyzer cards. Four are required to support 68-channel logic analyzer cards.

Mechanical Considerations

The Agilent E5382A single-ended flying lead probe set comes with accessories that trade off flexibility, ease of use, and performance. Discussion and comparisons between four of the most common intended uses of the accessories are included in the *E5382A Single-ended Flying Lead Probe Set User's Guide* (supplied with the probe). A list of replacable parts and additional accessories is given below.

Electrical Considerations

Data inputs on the E5382A flying lead probe are single-ended. The E5382A probe does not have a threshold reference voltage input. When using the E5382A probe, the logic threshold voltage is adjusted in the user interface.

The maximum nondestructive input voltage is +/- 40 Vdc.

Clock Input

The clock input on the E5382A probe can be a differential signal or a

single-ended signal. If the clock input is a differential signal, select the "differential" option in the clock threshold user interface.

If the clock input is a single-ended signal, ground the negative clock input and adjust the clock threshold voltage in the user interface. The minimum voltage swing for single-ended clock operation is 250 mV p-p.

In the user interface, the clock input threshold voltage adjustment is separate from the data input threshold voltage adjustments.

Replacable Parts and Additional Accessories

A variety of accessories are supplied with the E5382A to allow you to access signals on various types of components on your PC board. The following table shows the part numbers for ordering replacement parts and additional accessories.

```
Description               Qty        Agilent Part Number
--------------            ---        -------------------

Probe Pin Kit              4         16517-82107

High Frequency Probing Kit 8         16517-82104
(4 resistive signal pins & 4 solder-down grounds)

Ground Extender           20         16517-82105

Grabber Clip Kit          20         16517-92109

Right-angle Ground Lead   20         16517-82106

Cable - Main               1         E5382-61601

Probe Tip to BNC Adapter   1         E9638A
```

Using an Analysis Probe

Analysis probes, formerly called preprocessors, are products that make it easier to probe a specific microprocessor or bus.

Generally, analysis probes consist of hardware that probes a microprocessor or bus and routes the probed signals to connectors for the logic analyzer probe cables.

Analysis probes include configuration files that properly set up the logic analyzer. Also included, typically, are inverse assemblers or other tools for decoding the captured data.

The Setup Assistant (see the *Setup Assistant* help volume) in the logic

analysis system helps you to properly configure the logic analyzer.

Choosing the Sampling Mode

There are three logic analyzer sampling modes to choose from: *timing mode*, *state mode*, or *eye scan mode*.

In *timing mode*, the logic analyzer samples asynchronously, based on an internally-generated sampling clock.

In *state mode*, the logic analyzer samples synchronously, based on a sampling clock signal (or signals) from the device under test. Typically, the signal used for sampling in state mode is a state machine or microprocessor clock signal.

In *eye scan mode*, the logic analyzer samples small windows of time and voltage on data channels around a clock signal from the device under test. The resulting eye diagrams let you validate and characterize the data valid windows of the signals on a bus.

- "Selecting the Timing Mode (Asynchronous Sampling)" on page 43
- "Selecting the State Mode (Synchronous Sampling)" on page 46
- "In Either Timing Mode or State Mode" on page 53
- "Selecting the Eye Scan Mode" on page 55

Selecting the Timing Mode (Asynchronous Sampling)

In *timing mode*, the logic analyzer samples asynchronously, based on an internally-generated sampling clock.

- "To select the timing mode" on page 44
- "To select the conventional/transitional configuration" on page 44
- "To specify the sample period" on page 45

To select the timing mode

1. Open the logic analyzer Setup window.

2. Select the Sampling tab.

3. Choose the Timing Mode option.

You can also select the timing sampling mode in the "Pod Assignment Dialog" on page 175.

To select the conventional/transitional configuration

1. In the Sampling tab with Timing Mode selected, select the timing analyzer configuration. You can choose between:

    -   800 MHz / 64M Sample Conventional

        In this configuration, the analyzer samples and stores measurement data at each sampling interval, as often as every 1.25 ns.

NOTE:

With the Sample Period at 1.25 ns, data is acquired at four times the trigger sequencer rate. This means that data must be present for at least four samples before the trigger sequencer can reliably detect it. The trigger sequencer could miss data present for less than four sample periods.

The trigger sequencer treats the data as a group of four samples for each sequencer clock. This means that the trigger point indication could be off by three samples.

Although the trigger sequencer cannot detect all data, the analyzer will correctly capture all data present for at least one sample period.

    -   400 MHz / 32M Sample Transitional or Store Qualified

        In this configuration, the analyzer samples data at regular intervals, as often as every 2.5 ns, but only stores the data if it's different than the previously stored data.

        You can further qualify storage of transitions by ignoring data changes on particular labels.

        A time tag is stored with each stored data sample so that all sampled values can be reconstructed and displayed later.

NOTE: If all pods are used, memory depth is reduced by half in order to store the required time tags.

NOTE: With the Sample Period at 2.5 ns, data is acquired at two times the trigger sequencer rate. This means that data must be present for at least two samples before the trigger sequencer can reliably detect it. The trigger sequencer could miss data present for less than two sample periods.

The trigger sequencer treats the data as a group of two samples for each sequencer clock. This means that the trigger point indication could be off by one sample.

Although the trigger sequencer cannot detect all data, the analyzer will correctly capture all data present for at least one sample period.

See Also   "To specify default storing" on page 76

"How Samples are Stored in Transitional Timing" on page 157

"Default Storing Subtab" on page 190

To specify the sample period

When the logic analyzer is in timing (asynchronous sampling) mode, the *Sample Period* setting specifies how often the logic analyzer samples the signals from the device under test.

1. In the Sampling tab, with Timing Mode selected, enter the desired time between logic analyzer samples.

    To capture signal level changes reliably, the sample period should be less than half (many engineers prefer one-fourth) of the period of the fastest signal you want to measure.

The sample rate is the inverse of the sample period.

NOTE: In the conventional timing configuration, the sample rate is fixed at 1.25 ns.

Selecting the State Mode (Synchronous Sampling)

In *state mode*, the logic analyzer samples synchronously, based on a sampling clock signal from the device under test. Typically, the signal used for sampling in state mode is a state machine or microprocessor clock signal.

The clock signal can be either *Periodic* (synchronous all the time), or *Aperiodic* (bursted or varying in frequency).

- "To select the state mode" on page 47

- "To select the state speed configuration" on page 47

- "To set up the sampling clock" on page 48

State Mode Sampling Position

In order for a *state mode* logic analyzer to accurately capture data from a device under test, the logic analyzer's setup/hold time (window) must fit within the device under test's data valid window.

Because the location of the data valid window relative to the bus clock is different for different types of buses, the logic analyzer lets you adjust the sampling position in order to accurately capture data on high-speed buses (see "Understanding State Mode Sampling Positions" on page 256).

When the device under test's data valid window is less than 2.5 ns (roughly, for clock speeds >= 200 MHz), it's easiest to use *eye finder* to locate the stable and transitioning regions of signals and to automatically adjust sampling positions.

- "To automatically adjust sampling positions" on page 49

When the device under test's data valid window is greater than 2.5 ns (roughly, for clock speeds < 200 MHz), it's easiest to adjust the sampling position manually, without using the logic analyzer to locate the stable and transitioning regions of signals.

- "To manually adjust sampling positions" on page 52

To select the state mode

1. Open the logic analyzer Setup window.

2. Select the Sampling tab.

3. Choose the State Mode option.

You can also select the state sampling mode in the "Pod Assignment Dialog" on page 175.

To select the state speed configuration

1. In the Sampling tab, with State Mode selected, select one of the state analyzer configurations.

    The selected configuration specifies the speed up to which the state mode sampling clock will match input clock edges from the device under test. For example, in the *800 Mb/s / 64M State* configuration, the state mode sampling clock will match input clock edges up to 800 MHz.

    The selected configuration also specifies the memory depth for samples and whether only half of the logic analyzer channels are available.

    You can choose from:

    *   1500 Mb/s / 128M Half Channel

        In this configuration: The input clock signal must be periodic, and both edges indicate valid data. The clock input can only be used as a clock and not as an extra data channel. The logic analyzer setup/hold window is 500 ps, and *eye finder* must be used to automatically adjust sampling positions. The limited set of *16760 Half Channel State* trigger functions are available. In half-channel mode the analyzer accesses only the even channels (0, 2, 4, etc.).

    *   1250 Mb/s / 128M Half Channel

        In this configuration: The input clock signal must be periodic, and both edges indicate valid data. The clock input can only be used as a clock and not as an extra data channel. The logic analyzer setup/hold window is 1 ns, and sampling positions may be adjusted automatically or manually. The limited set of *16760 Half Channel State* trigger functions are available. In half-channel mode the analyzer accesses only the even channels (0, 2, 4, etc.).

- 800 Mb/s / 64M State

  In this configuration: The input clock signal can be periodic or
  aperiodic, and either rising, falling, or both edges can indicate valid
  data. The logic analyzer setup/hold window is 1 ns, and sampling
  positions may be adjusted automatically or manually. The limited set of
  16760 State trigger functions are available.

- 400 Mb/s / 32M State

  In this configuration: Either rising, falling, or both edges can indicate
  valid data. The logic analyzer setup/hold window is 2.5 ns, and sampling
  positions may be adjusted automatically or manually. The limited set of
  Turbo State trigger functions are available.

- 200 Mb/s / 32M State

  In this configuration: Rising, falling, or both edges can indicate valid
  data. The logic analyzer setup/hold window is 3.0 ns, and sampling
  positions may be adjusted automatically or manually. The *General
  State* trigger functions are available.

In all configurations but the *200 Mb/s / 32M State* configuration, if time is
counted (that is time tags are being stored), one pod must be left
unassigned in order to store the time tags. In the *200 Mb/s / 32M State*
configuration, memory depth is reduced by half if all pods are used and
time or state counts are being stored.

See Also "To set up the sampling clock" on page 48

"To manually adjust sampling positions" on page 52

"To automatically adjust sampling positions" on page 49

"Trigger Functions Subtab" on page 177

To set up the sampling clock

For the clock input signal that will be used:

1. In the Clock Setup, select the desired *Mode*. Your choices are *Periodic* or
   Aperiodic. If the State Mode is set to 1250 or 1500 Mb/s configuration, the
   input clock must be *Periodic*.

2. Select the pod's Master button (under the activity indicator).

3. Specify when to sample. Your choices are *Rising Edge*, *Falling Edge*, or *Both Edges*. Only *Both Edges* is available in the 1250 and 1500 Mb/s configurations.

See Also   "To select the state speed configuration" on page 47

To automatically adjust sampling positions

When adjusting the state mode sampling position with *eye finder*, the logic analyzer looks at signals from the device under test, figures out the location of the data valid window in relation to the sampling clock, and automatically sets the sampling position.

Because *eye finder* automatically runs on individual channels, it can correct for the small delay effects caused by probe cables and circuit board traces. This makes the logic analyzer's setup/hold window smaller and lets you accurately capture data at higher clock speeds.

Eye finder requires:

• At least 500 transitions on each signal during its run. (You can use the advanced *eye finder* settings to cause longer or shorter runs.)

• All devices which can drive each signal should contribute to the stimulus.

• All device under test operating modes relevant to the eventual logic analysis measurement should contribute to the stimulus as well.

NOTE: *Eye finder* measurements and normal logic analyzer measurements cannot run simultaneously.

To run *eye finder*

1. Probe the device under test by connecting the logic analyzer channels.

2. Select the state (synchronous sampling) mode (see "To select the state mode" on page 47).

3. Format labels for those logic analyzer channels.

4. Make sure that the device under test and the logic analyzer have warmed up to their normal operating temperatures.

5. In the Format tab, select the *Setup/Hold* button.

6. In the Sampling Positions dialog, select the *Eye Finder* option.

7. In the Eye Finder Setup tab, select the *Use signals from Device Under Test* option.

   The *Use demo data (no probes required)* option is for demonstration purposes only.

8. Choose the labels that you wish to run *eye finder* on.

   You may want to run *eye finder* on channel subsets, for example, when certain bus signals transition in one operating mode (of the device under test) and other bus signals transition in a different operating mode.

9. Select the *Run Eye Finder* button.

   For more information on run messages, see "Eye Finder Run Messages" on page 162.

   When *eye finder* finds more than one stable region on a channel, it uses the current sampling position as a hint about which stable region it should suggest a position for.

   If *eye finder* picks the wrong stable region, you can expand the label and drag the blue Sampling Position line into the correct stable region. The suggested sampling position for that region will be shown (see "How Selected/Suggested Positions Behave" on page 161).

10. If you have moved the sampling position and wish to return to the suggested positions, go to the Eye Finder Results tab, select a label button or the Results menu, and choose the "set to suggested" command.

    For more information on informational messages in the Eye Finder Results tab, see "Eye Finder Info Messages" on page 165.

Eye finder finds optimal sampling positions for the actual specific conditions -- amplitude, offset, slew rates, and ambient temperature. Therefore, you will get the best results by running *eye finder* under the same conditions that will be present when logic analysis measurements are made.

To run *eye finder* repetitively

1. Select the *Repetitive Run* option in the Eye Finder Setup tab.

2. Select the *Run Eye Finder (r)* button.

In the Eye Finder Results tab, you can see how the stable and transitioning areas vary over time.

3. Select the *Stop Eye Finder* button.

To view *eye finder* data as a bus composite

When you want a compressed, high-level view of the *eye finder* data:

1. In the Eye Finder Results tab, select the label button and choose the *View as Bus Composite* command.
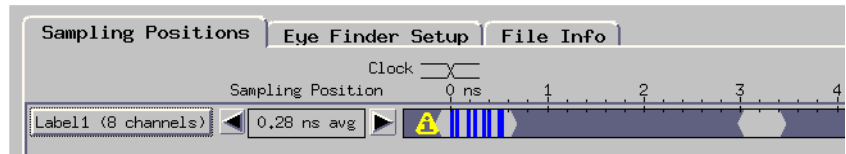
Average sampling positions as well as stable and transitioning areas are displayed for the whole label. This is the default. Stable areas show positions where every channel in the label is stable.

To view *eye finder* data as a stack of channels

When you want more resolution in your view of the *eye finder* data:

1. In the Eye Finder Results tab, select the label button and choose the *View as Stack of Channels* command.
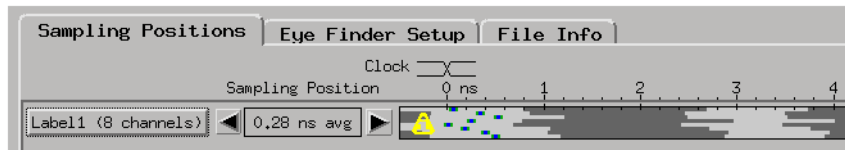
Individual sampling positions and stable and transitioning areas for all the channels in a label are shown.

To save/load *eye finder* data

While the *eye finder* sampling positions are saved with the logic analyzer configuration, *eye finder* measurement data is not; therefore,

eye finder data must be saved and loaded separately.

1. In the File Info tab, select the *Save As...* or *Load...* buttons.

    You can also choose the *Save Eye Finder* or *Load Eye Finder* command from the File menu.

2. In the file browser dialog, name the file to be saved or select the file to be loaded.

    For more information on save/load messages, see "Eye Finder Load/Save Messages" on page 167.

See Also   "Understanding State Mode Sampling Positions" on page 256

"Eye Finder Advanced Settings Dialog" on page 170

"To manually adjust sampling positions" on page 52

To manually adjust sampling positions

Although the *Eye Finder* option was intended for automatically adjusting state mode sampling positions, you can also use it to manually adjust sampling positions. You don't have to *Run Eye Finder* to locate stable and transitioning regions on signals, just go directly to the Eye Finder Results tab, and drag the sampling positions to the proper locations.

1. Select the state (synchronous sampling) mode (see "To select the state mode" on page 47).

2. In the Format tab, select the *Setup/Hold* button.

3. In the Sampling Positions dialog, select the *Eye Finder* option.

4. In the Eye Finder Results tab, drag the sampling positions to the proper locations.

    You can select bus labels to expand or collapse the channels in the label.

When using the Eye Finder option to manually adjust state mode sampling positions, the sampling positions are saved with the logic analyzer configuration (see "Saving and Loading Logic Analyzer Configurations" on page 116).

In Either Timing Mode or State Mode

- "To specify the trigger position" on page 53
- "To set acquisition memory depth" on page 53
- "To name an analyzer" on page 54
- "To turn an analyzer off or on" on page 54

To specify the trigger position

1. In the Sampling tab (or in the Settings subtab of the Trigger tab), select the trigger position.

   Specify whether you want to look at data after the trigger (Start), before and after the trigger (Center), before the trigger (End), or use a percentage of the logic analyzer's memory for data after the trigger (User Defined).

In the timing sampling mode's *800 MHz / 64M Sample Conventional* configuration, when a Run is started, the analyzer will not look for a trigger until the specified percentage of pretrigger data has been stored. After a trigger has been detected, the specified percentage of posttrigger data is stored before the analyzer halts. This ensures that both pretrigger and posttrigger storage are complete.

In the state sampling mode, or in the timing sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration, when a Run is started, the analyzer immediately looks for the trigger condition. In other words, the trigger position setting specifies the *maximum* amount of data that should be stored before the trigger.

To set acquisition memory depth

If you need less data and want measurements to run faster, you can limit the number of samples that are stored in logic analyzer acquisition memory.

1. In the Sampling tab (or in the Settings subtab of the Trigger tab), select the acquisition depth.

   The number of samples that can be chosen for the Acquisition Depth are approximations. The combination of *count* tags, pod assignments, and configuration modes affect what choices are available.

To name an analyzer

You can give more descriptive names to a logic analyzer.

1. In the Sampling tab, select the *Analyzer Name* field.

2. Enter the new name.

   The name now appears below the instrument tool icon in the workspace.

You can also name analyzers in the "Pod Assignment Dialog" on page 175.

To turn an analyzer off or on

You may want to turn an analyzer off if you don't want it to be included in further measurements.

To turn an analyzer off

1. In the Sampling tab, select the *On* box that is checked.

2. In the Analyzer Shutdown Options dialog, choose either:

   • Soft -- This will leave the logic analyzer window but turn off most options.

   • Hard -- This will remove the logic analyzer and its display tools from the Workspace.

You can also turn an analyzer off in the "Pod Assignment Dialog" on page 175.

To turn an analyzer back on

1. If you used the *Soft* option when turning the logic analyzer off, you can turn it on again by selecting the *Off* check box.

2. If you used the *Hard* option when turning the logic analyzer off, you can

turn it on again by selecting the *Setup* button in the System window or by dragging the analyzer's instrument tool icon to the workspace in the Workspace window.

Selecting the Eye Scan Mode

In *eye scan mode*, the logic analyzer becomes a tool for validating and characterizing the data valid windows of signals latched by a clock. The *Eye Scan* display shows eye diagrams for each channel, like an oscilloscope, but with less accuracy and resolution. The *Eye Scan* display can help you quickly identify setup/hold or other signal integrity anomalies which can then be examined in greater detail with a high-speed or high-bandwidth oscilloscope.

- "To select the eye scan mode" on page 55
- "To select the eye scan mode speed configuration" on page 55
- "To set up the eye scan mode reference clock" on page 56

See Also   "Setting Up and Running Eye Scan Measurements" on page 119

"Displaying Captured Eye Scan Data" on page 133

"Saving and Loading Captured Eye Scan Data" on page 152

"Understanding Eye Scan Measurements" on page 259

To select the eye scan mode

1. Open the logic analyzer Setup window.

2. Select the Sampling tab.

3. Choose the Eye Scan Mode option.

To select the eye scan mode speed configuration

1. In the Sampling tab, with Eye Scan Mode selected, select one of the eye scan mode speed configurations.

   The selected configuration specifies the speed that the input reference clock may be as fast as. For example, in the *800 Mb/s / Eye Scan*

configuration, the input reference clock edges can occur at rates up to 800 MHz.

You can choose from:

- 800 Mb/s / Eye Scan

  In this configuration: Either rising, falling, or both edges of the input reference clock can indicate valid data. All of the logic analyzer channels are available. The maximum number of clocks that can be processed at each scan point is 60,000,000.

- 1500 Mb/s / Eye Scan

  In this configuration: Both edges of the input reference clock indicate valid data. Only half of the logic analyzer channels are available. The maximum number of clocks that can be processed at each scan point is 120,000,000.

To set up the eye scan mode reference clock

For the clock input signal that will be used:

1. Select the pod's Master button (under the activity indicator).

2. Select whether data is valid on the *Rising Edge*, *Falling Edge*, or *Both Edges*.

   In the *1500 Mb/s / Eye Scan* speed configuration, data must be valid on *Both Edges*.

See Also   "To select the eye scan mode speed configuration" on page 55

Formatting Labels for Logic Analyzer Probes

The Format tab is mainly for assigning bus and signal names (from the device under test), to logic analyzer channels. These names are called labels. Labels are used when setting up triggers and displaying captured data.

The Format tab also lets you do things like assign pods to the logic analyzer, specify the logic analyzer threshold voltage, change the label polarity, reorder bits in a label, and turn labels off or on.

The Format tab has activity indicators that show signal levels.

- "To assign pods to the analyzer" on page 57

- "To set pod threshold voltages" on page 58

- "To set clock threshold voltages" on page 59

- "To assign probe channels to labels" on page 60

- "To import label names and assignments from a netlist" on page 62

- "To import label definitions from an ASCII file" on page 63

- "To export label definitions to an ASCII file" on page 64

- "To change the label polarity" on page 64

- "To reorder bits in a label" on page 65

- "To turn labels off or on" on page 66

To assign pods to the analyzer

The logic analyzer pods can be assigned to the logic analyzer, or, they can be left unassigned.

1. In the *Format* tab, select the *Pod Assignment* button.

2. In the Pod Assignment dialog, drag a pod to the logic analyzer.

3. Select the Close button.

Capturing Data on 17 Channels in State Mode

On a single-card 16760A logic analyzer in the state (synchronous) sampling mode, you can assign pod 2 to the logic analyzer and unassign pod 1. (One pod must be unassigned in order to store time tags.) Even though pod 1 is unassigned, its J clock input is still used as the sampling clock input. This allows the 16 channels and the K clock input on pod 2 to be used as data sampling channels. This lets you capture data on high-speed buses that have 17 data bits, and a clock.

When you assign pods this way, the logic analyzer loses its ability to detect, to prevent triggering or sequencing on, and to remove an initial spurious sample in the acquisition (which can occur if the logic analyzer measurement is started before the state clock input signal starts toggling). However, because most high-speed devices have continuously running periodic clocks, the appearance of an initial spurious sample is unlikely.

When the K clock input is used as a data channel like this, it cannot be used as a qualifier signal for eye scan measurements (unless the data signal also happens to be the necessary clock qualification input for eye scan).

See Also

"To set up qualified eye scan measurements" on page 122

"Selecting the State Mode (Synchronous Sampling)" on page 46

To set pod threshold voltages

The *threshold voltage* is the voltage level that a signal must cross before the logic analyzer recognizes a change in logic levels.

1. In the *Format* tab, select the threshold button located just below the pod name.

2. In the Pod threshold dialog, either:

   • Select the *Standard* option; then, select one of the predefined threshold voltages from the drop-down list.

   • Select the *External Ref* option. This option appears when the E5378A single-ended probe is used. It should be selected when the probe's threshold voltage reference inputs are used and are connected to the

appropriate threshold voltage reference level.

- Select the *Differential* option. This option appears when the E5379A differential probe is used. It should be selected when differential signals are probed. The difference voltage (Vin+ - Vin-) must be greater than or equal to 200 mV p-p.

- Select the *User Defined* option and enter the desired threshold voltage value. The threshold level is selectable from -6.0 volts to +6.0 volts.

3. If you don't want the change to apply to all pods and clock input thresholds, deselect the checked box next to *Apply threshold setting to all pods*.

4. Select the *Close* button.

NOTE: The logic analyzer requires a minimum voltage swing of 250 mV for the E5378A single-ended probe or 300 mV for the E5380A MICTOR-compatible probe to recognize changes in logic levels. If you are using the E5379A differential probe, 200 mV differential is required.

NOTE: The specified pod threshold voltage is also applied to the pod's clock threshold if *Apply settings to all pods* is selected. However, the pod's clock threshold can also be changed independently.

See Also

To set clock threshold voltages

The *threshold voltage* is the voltage level that a signal must cross before the logic analyzer recognizes a change in logic levels.

1. In the *Format* tab, select the *Clk Thresh...* button located just below *Data On Clocks*.

2. In the Clock Thresholds dialog, select the button of the clock whose threshold voltage you wish to set.

3. In the J, K, etc., threshold dialog, either:

   • Select the *Standard* option; then, select one of the predefined threshold voltages from the drop-down list.

   • Select the *Differential* option. This option appears when the E5378A single-ended probe or the E5379A differential probe is used. It should be selected when the clock input is a differential signal. The difference voltage (Vin+ - Vin-) must be greater than or equal to 200 mV p-p.

   • Select the *User Defined* option and enter the desired threshold voltage value. The threshold level is selectable from -3.0 volts to +5.0 volts.

4. Select *Close* to close the J, K, etc., threshold dialog.

5. Select *Close* to close the Clock Thresholds dialog.

See Also   "To set pod threshold voltages" on page 58

"Using the E5378A Single-Ended Probe" on page 35

"Using the E5379A Differential Probe" on page 37

"Using the E5380A Mictor-Compatible Probe" on page 39

"Using the E5382A Single-ended Flying Lead Probe Set" on page 40

To assign probe channels to labels

The logic analyzer lets you assign names (labels) to logic analyzer channels so that it's easier to set up triggers and interpret the captured data when displayed.

Typically, you give labels the names of the buses and signals in the device under test that are are being probed.

1. In the Format tab, select a label button, and either:

   • Choose the *Rename* command, enter the label name, and select the OK button.

- Or, choose the *Insert before* or *Insert after* command, enter the label name, and select the OK button.

2. In the label row, select the button of the pod that contains the channels you want to assign.

3. Either choose one of the standard label assignments or choose *Individual*.

   (*) (asterisk) indicates an assigned bit.

   (.) (period) indicates an unassigned bit.

   (R) indicates an assigned bit in a reordered label.

   If you chose *Individual*:

   a. In the "label - pod" dialog, select the channels you want to assign/unassign.

   b. Select the OK button.

A maximum of 32 channels can be assigned to a label.

In the Format tab, least significant pod channels (bit 0) are on the right and most significant pod channels (bit 15) are on the left. (The bit numbers are shown just below the activity indicators.)

Labels can contain bits that are not consecutive; however, bits are always numbered consecutively within a label.

To delete labels

1. Select the label name that you want to delete.

2. Choose *Delete*.

   If only one label is defined, it cannot be deleted.

When you delete labels, their bit assignments are not saved. However, you can make a label inactive and save its bit assignments by turning the label off.

See Also   "To reorder bits in a label" on page 65

"To turn labels off or on" on page 66

To import label names and assignments from a netlist

You can create label names and assign logic analyzer probe channels by importing netlists. These netlists come from the Electronic Design Automation (EDA) tools used to design the device under test, and they contain information about the signals on the connectors built into the device under test for the E5378A, E5379A, or E5380A probes for the 16760A logic analyzer.

1. In the *Format* tab, select *File* from the menu; then, select the *Import Netlist...* menu item.

2. Read the information and follow the instructions in the Import Netlist wizard's dialogs.

   Select *Next* --> to go to the next dialog, <-- *Prev* to go to the previous dialog, *Cancel* to exit the wizard, or *Done* to complete the netlist import.

The E5378A and E5380A probes require two logic analyzer pods, and the E5379A probe requires one logic analyzer pod. When mapping connector names to logic analyzer pods in the Import Netlist wizard, two pods must be assigned to the logic analyzer in order for the E5378A and E5380A probe types to be selectable.

The E5379A probe is for differential signals where there are two connector pins (negative and positive) for each signal. The Import Netlist wizard will either merge the two net names or append "-/+" when creating label names to indicate that the labels are for differential signals.

Multiple labels are automatically created for buses wider than 32-bits because 32 is the maximum number of channels that can be assigned to a label.

See Also

"Probing the Device Under Test" on page 35

"To assign pods to the analyzer" on page 57

To import label definitions from an ASCII file

You can create label names and assign logic analyzer probe channels by importing label definitions from an ASCII file.

1. In the *Format* tab, select *File* from the menu; then, select the *Import Labels...* menu item.

2. In the *File Selection* dialog, select the name of the file that contains the label definitions.

3. Select *OK*.

Up to 126 labels can be defined. Label names can be up to 20 characters long (additional characters are truncated). A maximum of 32 channels can be assigned to a label.

When updating labels by importing label definitions, make sure that the labels are turned ON. Labels that are not active will not be updated.

Label Definition File Format

Label definition files have one definition per line, where each line has the format:

`label_name;pod_name[channel_list];pod_name[channel_list] ...`

In a channel list, individual channels are separated by commas (","), and a range of channels is separated by a colon (":").

Examples

To assign label name "Blue" to channel 5 on pod A2:

`Blue;A2[5]`

To assign label name "Green" to channels 5 through 2 and channel 0 on pod A2:

`Green;A2[5:2,0]`

To assign label name "Yellow" to channel 1 on pod A2 and channel 0 on pod A1:

`Yellow;A2[1];A1[0]`

To assign label name "Orange" to channels 15 through 5 on pod A3, channel 5 on pod A2, and channel 6 on pod A1:

`Orange;A3[15:5];A2[5];A1[6]`

To assign label name "Red" to the K clock of the logic analyzer in slot A:

`Red;CK[AK]`

See Also   "To assign probe channels to labels" on page 60

"To import label names and assignments from a netlist" on page 62

"To export label definitions to an ASCII file" on page 64

"To turn labels off or on" on page 66

To export label definitions to an ASCII file

1. In the *Format* tab, select *File* from the menu; then, select the *Export Labels...* menu item.

2. In the *File Selection* dialog, select the name of the file that will contain the label definitions.

3. Select *OK*.

See Also   "To assign probe channels to labels" on page 60

"To import label names and assignments from a netlist" on page 62

"To import label definitions from an ASCII file" on page 63

To change the label polarity

While negative logic is rare in circuits (the main exception at this time is RAMBUS), you can change the label polarity if the device under test uses negative logic.

1. In the Format tab, select the polarity button (next to the label button) to toggle between positive (+) and negative (-) polarity.

Positive polarity means that a high voltage is a logic "1".

Negative polarity means that a high voltage is a logic "0".

Changing the label polarity will have the following effects:

- "1" and "0" values flip in the trigger condition.

- Waveforms and bus values (where shown) invert in the Waveform display tool.

- "1" and "0" values flip in the Listing display tool.

Changing the label polarity does not affect:

- Edge definitions for clock setup and *edge terms*.

- Symbol definitions for the logic analyzer.

- Activity indicators.

To reorder bits in a label

In cases where buses in the device under test haven't been probed with consecutive logic analyzer channels, you can reorder the bits in a label.

1. In the Format tab, select the label button whose bits you want to reorder.

2. Choose *Reorder bits*.

3. In the Change Bit Order dialog:

    - To reorder the bits individually, enter the bit that the probe channel should be mapped to.

    - To swap the high and low order bytes or words, select the button *Big Endian to Little Endian* at the bottom of the dialog.

    - To return to sequentially ordered bits, select the button *Default Order* at the bottom of the dialog.

4. Select the *OK* button.

    The label now shows an "R" to indicate that the assigned bit has been reordered.

NOTE:
Labels with reordered bits cannot be used as *range terms* or <, <=,>, >= in triggers.

To turn labels off or on

When you temporarily want to remove a label and its data, you can turn off the label. The label name and its bit assignments are preserved.

NOTE:
In the timing mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration, the logic analyzer only looks for transitions on labels that are turned on. Data is stored for the labels that are turned off, but only when there is a transition on labels that are turned on. If you turn a label on after a run, or re-assign a channel from a label that is turned off to a label that is turned on, be aware that transitions on that label or channel are only coincidental to labels that were turned on at the time of the run.

To turn a label off

1. In the Format tab, select the label button that you want to turn off.
2. Choose *Label [ON]* to toggle it off.

   At least one label must remain on.

To turn a label on

1. In the Format tab, select the label button that you want to turn on.
2. Choose *Label [OFF]* to toggle it on.

To display a label that was off

1. Turn on the label.
2. At the bottom of the window, select the *Apply* button.

   The label's data appears in the display windows.

3

Using the Logic Analyzer in Timing or State Mode

- "Setting Up Triggers and Running Measurements" on page 69

See Also   Measurement Examples (see the *Measurement Examples* help volume)

Setting Up Triggers and Running Measurements

This section describes setting up triggers for the timing and state sampling modes and for all configurations within these modes. Some triggering functionality is only available in certain modes and configurations.

- "Using Trigger Functions" on page 70

- "Using Other Trigger Features" on page 75

- "Editing the Trigger Sequence (Timing or 200, 400 Mb/s State Only)" on page 78

- "Editing Advanced Trigger Functions (Timing or 200 Mb/s State Only)" on page 83

- "Saving/Recalling Trigger Setups" on page 90

- "Running Measurements" on page 91

In General...   Use trigger functions for basic measurements.

For more complicated measurements, where no trigger function exists, start with a trigger function that's similar to the measurement you want to make. Then, if possible, break down the trigger function and edit the advanced trigger sequence levels.

Timing Analyzer Triggers   Everything that looks like a button in the trigger definition gives you a way to modify the trigger setup.

For example, to look for a edge instead of a pattern, select the button that equates a label with a pattern and choose an edge comparison instead.

State Analyzer Triggers   For every analysis sample, the logic analyzer needs to know two things:

1. Should some action (like a trigger) be taken as a result of this sample?

2. What should be done with this sample? That is, should it be stored in logic analyzer memory or should it be discarded? (This question doesn't need to be asked when using the conventional timing analyzer configuration because all samples are stored.)

State and transitional timing analysis trigger definitions are made simpler with a *default storage* qualifier. This makes it possible to ignore, at all trigger sequence levels, the question about what to do with the captured data samples.

Of course, sometimes it's useful to specify storage qualifiers at certain levels in the trigger sequence. For this, you can insert storage *actions* in the trigger sequence before trigger or goto actions. Storage actions in the trigger sequence override the default storage qualifier for the samples that cause the trigger or goto actions to occur. Storage actions can also be used to turn on or off the default storing.

Using Trigger Functions

Many common measurement setups are provided with the logic analyzer. These setups are called *trigger functions*, and you can use them for quick measurement setup.

For more complicated timing mode measurements, where no trigger function exists, start with a trigger function that's similar to the measurement you want to make. Then, *break down* the trigger function and edit the advanced trigger specification.

NOTE: In the 16760A logic analyzer, you cannot break down trigger functions in the 400, 800, 1250 and 1500 Mb/s state mode configurations.

- "To select a trigger function" on page 70

- "To specify a label pattern event (Timing only)" on page 71

- "To specify a label edge event" on page 72

- "To break down a trigger function (timing or 200 Mb/s state only)" on page 72

- "To create a trigger function library (timing or 200 Mb/s state only)" on page 73

To select a trigger function

1. In the Trigger tab's Trigger Functions subtab, select the appropriate

trigger function.

A picture describing the trigger function is shown.

2. Select the *Replace* button (or *Insert before* or *Insert after* button) to move it to the Trigger Sequence below.

3. In the Trigger Sequence, select and/or enter the appropriate labels, values, and options.

To specify a label pattern event (Timing only)

Label pattern events let you specify patterns or ranges on a bus.

1. Select the label name button and choose the label that you want to look for a pattern on.

   You can also insert other label events if you want to look for multiple patterns on multiple labels. Once another label event is inserted, you can choose *And* if both label events must occur in the same sample or *Or* if only one of the label events must occur.

2. Select the operator button and choose the appropriate operator.

   The *In range* and *Not in range* operators consider the values you enter to be inside the range. Ranges and inequalities cannot be set on labels whose bits have been reordered.

3. Select the number base button, and choose the number base that you want.

   If the number base is changed in one window, the number base in other windows may not change accordingly. For example, the number base assigned to symbols is unique, as is the number base assigned in the Listing window.

4. Enter the label value.

   *X*s mean you don't care about the value on the specified bits. *X*s are not allowed in ranges or inequalities.

   If you chose the *Symbols* or *Line #s* number base, select the *Absolute XXXX* button, and use the Symbol Selector dialog to choose the symbol or line number value.

See Also

"To enter symbolic label values" on page 105

"Symbols Selector Dialog" on page 195

To specify a label edge event

Label edge events let you specify edges and glitches on a bus. Label edge events are only available in certain timing mode trigger functions.

1. Select the label name button and choose the label that you want to look for a pattern on.

   You can also insert other label events if you want to look for multiple patterns on multiple labels. Once another label event is inserted, you can choose *And* if both label events must occur in the same sample or *Or* if only one of the label events must occur.

2. Select the edge assignment button.

3. In the Specify Edge/Glitch dialog, select the edges or glitches that you're looking for on particular logic analyzer channels.

   When you select multiple edges or glitches, they are ORed together, and any one of the edges or glitches in a sample will satisfy the label edge event. If you want to AND edges or glitches on a label, insert multiple label edge events and AND them together.

4. Select the OK button.

See Also "To specify a label pattern event (Timing only)" on page 71

To break down a trigger function (timing or 200 Mb/s state only)

When a trigger function doesn't quite let you set up the trigger you want, you can break it down and edit the resulting advanced trigger function.

1. In the Trigger tab, select the number button of the trigger sequence level whose trigger function you want to break down.

2. Choose *Break down function*.

   Breaking down the trigger function will be permanent (although you can choose the Undo command from the Edit menu if no other editing has

taken place).

If you only want to look the advanced trigger function, without editing it, you can *expand* the trigger function.

3. Select *OK* in the confirmation dialog.

To expand a trigger function

1. In the Trigger tab, select the number button of the trigger sequence level whose trigger function you want to expand.

2. Choose *Expand function*.

To compress a trigger function

Expanded trigger functions can be compressed back into their original form.

1. In the Trigger tab, select the number button of the trigger sequence level whose trigger function you want to compress.

2. Choose *Compress function*.

See Also   "Editing Advanced Trigger Functions (Timing or 200 Mb/s State Only)" on page 83 for information on editing trigger functions that are broken down.

To create a trigger function library (timing or 200 Mb/s state only)

You can create your own libraries of trigger functions that are separate from logic analyzer configuration files (unlike saved/recalled trigger setups).

1. In the Trigger tab's Trigger Functions subtab, select the *Trigger function libraries* button.

2. In the Trigger function libraries dialog, select the *Create* button.

3. In the Create User Library dialog, enter the library name and description, and select OK.

4. In the Edit Trigger Function Library dialog, choose the *Add function* button.

5. In the Create User Function dialog, enter the function name and

description, and select the levels from the current trigger sequence that be the trigger function; then, select OK.

Once you have created a trigger function library with trigger functions, you can:

- Load or unload the trigger function library.

- Insert and break down trigger functions from the loaded library just like normal trigger functions.

- Copy trigger function libraries to other logic analysis systems and load them into other logic analyzers that have trigger function library capability.

- Edit the trigger function library, adding or deleting functions, or delete the library.

NOTE: If a *trigger sequence* or configuration file uses a trigger function library that has been deleted, or a trigger function that has been deleted from a library, the logic analyzer replaces the missing function with the default trigger function.

To load/unload trigger function libraries

1. In the Trigger tab's Trigger Functions subtab, select the *Trigger function libraries* button.

2. Select the library from the list.

   Only libraries created in the same sampling mode are available.

3. Select the *Load* (or *Unload*) button.

   All of the library's trigger functions are added to (or removed from) the list of trigger functions.

To copy trigger function libraries between systems

1. Connect your logic analysis system to the network. (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

2. Using a computer on your network, copy the appropriate files from the /logic/trigger_functions/ directory to a central location, or directly to other logic analyzers on the network.

See Also   "To break down a trigger function (timing or 200 Mb/s state only)" on

page 72

Using Other Trigger Features

Other subtabs in the Trigger tab let you do things like: specifying whether a state or time count is stored with samples, or setting up the default storing options.

-

-

-

To count time, states, or turn counting off

You can specify whether a time or state count is stored with samples.

1. In the Trigger tab's Settings subtab, select the Count option button and choose either *Off*, *Time*, or *States*.

    The *States* option is only available in the state sampling mode's *200 Mb/s / 32M State* configuration.

2. If you chose *States*:

    a. Select the *Define* button.

    b. In the State count qualify dialog, select the *Count if* or *Count if NOT* option.

    c. Specify events that identify the states to be counted or not counted.

    d. If you would like to specify the evaluation order of the event list, select *Group events*. Then, in the Group Events dialog, either select the *Add parens* button to group events or select the *Remove parens* button to ungroup events. When you're done grouping events, select the OK button.

In the state mode's *200 Mb/s / 32M State* configuration, memory depth is reduced by half if all pods are used and time or state counts are being stored. In all other state mode configurations, time counts require one

pod to be left unassigned.

See Also   "To select the state speed configuration" on page 47

"To assign pods to the analyzer" on page 57

To specify default storing

You can set up default storing so that only the data samples you're interested in are saved in logic analyzer acquisition memory.

NOTE: Default storing in the timing sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration requires time tags to be stored with each stored data sample in order for all sampled values to be reconstructed and displayed later. If all data pods are used, memory depth is reduced by half in order to store the required time tags.

1. In the Trigger tab's Default Storing subtab, select the *Store by default* option button and choose:

   - *Anything* to store all samples.

   - *Nothing* to store no samples.

   - *Custom* to specify which samples are stored.

   - *Transitions* (only available in the timing sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration) to store a sample only if it's different than the previously stored sample.

2. If you chose *Custom*:

   a. Select the *Store if* or *Store if NOT* option.

   b. Specify events that identify the states to be stored or not stored.

   c. If you would like to specify the evaluation order of the event list, select *Group events*. Then, in the Group Events dialog, either select the *Add parens* button to group events or select the *Remove parens* button to ungroup events. When you're done grouping events, select the OK button.

3. If you chose *Transitions* and you would like to further qualify storage of transitions by ignoring data changes on particular labels (for example, if there's a high occurrence of transitions on labels that are meaningless in

the context of the measurement):

a. Select the *Select Labels* button.

b. In the *Transitional Label Select* dialog, highlight the desired label from the *Available Labels* list; then, select the right-arrow button to move that label to the *Ignore Edges On* list.

c. Repeat as needed for additional labels.

d. Select *OK* to save the selection and close the dialog.

NOTE: If you have a channel that is shared across multiple labels, all labels containing that channel must be on the *Ignore* list before transitions on that channel will no longer cause a sample to be stored.

Ignoring data changes on particular labels let you store more of the transitions you're interested in over a greater period of time.

In the trigger sequence, unless *Transitions* is selected, you can override default storing for the samples that cause actions to occur, or you can turn default storing on or off, by inserting store actions.

The Agilent Technologies 16760A logic analyzer does not implement the "Branches taken" feature of past logic analyzers. The best way to store only the states that cause sequence level branches is by setting up default storing to *Nothing*, and inserting a *Store sample* action in each sequence level.

NOTE: When store qualification is performed in the state mode's 400, 800, 1250, and 1500 Mb/s configurations, there may be the case where data stored in memory is further disqualified. As a result, you may see a non-contiguous listing of states as well as a reduction of usable memory. In the timing mode's transitional configuration, these extra samples are not removed, so sometimes data not meeting the store qualifier is displayed.

To clear default storing changes

1. When the Trigger tab is displayed, select *Clear Default Store* from the Clear menu.

See Also "Storage Qualification" on page 249 in "Understanding Logic Analyzer Triggering" on page 240

To specify whether default storing is initially on or off

In the *state* sampling mode, or in the *timing* sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration when *Transitions* are not stored by default, you can specify whether the default storing is initially on or off.

1. In the Trigger tab's Default Storing subtab, select the *At start of acquisition* option button and choose either *On* or *Off*.

See Also

Editing the Trigger Sequence (Timing or 200, 400 Mb/s State Only)

When you want to trigger on several events in the device under test that follow one another, you need to use multiple levels in the trigger sequence.

For example, multiple levels in the trigger sequence let you trigger on a particular function calling sequence or capture only the execution within a particular program loop.

• "To clear the trigger sequence" on page 83

To insert/replace/delete sequence levels

To insert sequence levels

1. In the Trigger tab's Trigger Sequence area, select the level that you want to insert before or after.

   A yellow box appears around the level.

2. In the Trigger Functions subtab, select the trigger function you want to insert.

   A picture describing the trigger function is shown.

3. Select the *Insert before* or *Insert after* button, or select the level button and choose *Insert LEVEL before* or *Insert LEVEL after*.

To replace sequence levels

1. In the Trigger tab's Trigger Sequence area, select the level that you want to replace.

   A yellow box appears around the level.

2. In the Trigger Functions subtab, select the trigger function you want to insert.

   A picture describing the trigger function is shown.

3. Select the *Replace* button, or select the level button and choose *Replace LEVEL*.

To delete sequence levels

1. In the Trigger tab's Trigger Sequence area, select the level that you want to delete.

   A yellow box appears around the level.

2. Select the *Delete* button, or select the level button and choose *Delete LEVEL*.

See Also   "To cut/copy-and-paste sequence levels" on page 80

To cut/copy-and-paste sequence levels

You can change the order of levels in the trigger sequence by cutting-
and-pasting or you can copy levels by copying-and-pasting.

1. In the Trigger tab's Trigger Sequence area, select the level that you want to
   cut or copy.

   A yellow box appears around the level.

2. Select *Cut level* or *Copy level* from the Edit menu, or select the level
   button and choose *Cut LEVEL* or *Copy LEVEL*.

3. Select the level that you want to paste before or after.

4. Select *Paste level before* or *Paste level after* from the Edit menu, or select
   the level button and choose *Paste LEVEL before* or *Paste LEVEL after*.

See Also   "To insert/replace/delete sequence levels" on page 79

To specify a level's goto or trigger action (timing or 200 Mb/s state only)

When using multiple levels in the trigger sequence, you specify the
event search order by setting the goto or trigger action in each
sequence level.

1. In the Trigger tab's Trigger Sequence area, select the level whose goto or
   trigger action you want to specify.

   A yellow box appears around the level.

2. Select the Trigger or Goto button and choose the appropriate Goto or
   Trigger action.

3. If you chose the *Goto* or *Trigger and goto* action, select the level button
   and choose the appropriate level.

Searching for events that trigger the analyzer always starts at the first
level. Searching stops after one of the *Trigger and fill memory*
actions.

One level can branch to one of several other levels depending on the

evaluation of the sample. You can set up multi-way branches using advanced trigger functions or by selecting an *If* button and choosing *Insert BRANCH*.

NOTE: When you want to test a single sample for multiple conditions and take different actions based on which is true, use branches within a trigger sequence level. When you want to test different samples, use different sequence levels.

NOTE: In the timing mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration, only 2 branches are available per sequence level.

See Also "Understanding Logic Analyzer Triggering" on page 240

"To view a picture of the trigger sequence" on page 83

"Advanced Trigger Functions" on page 183

To send e-mail when the trigger occurs (timing or 200 Mb/s state only)

You can set up the logic analyzer to send e-mail when the trigger occurs. This is useful when triggering on an event that rarely occurs, when you may not be around the logic analyzer to see that it triggered.

1. In the Trigger tab's Trigger Sequence area, select the level whose trigger you want to send e-mail on.

    A yellow box appears around the level.

2. Select the Trigger or Goto button and choose the *Trigger, send e-mail, and fill memory* action.

3. Select the *E-mail Setup* button.

4. In the E-mail Setup dialog, enter the name of the SMTP (see page 82) mail server (if you don't know this, contact your *System Administrator*), the recipient's e-mail address (use spaces to separate multiple addresses), and the text of the message.

    If you want e-mail to be sent on each trigger of a repetitive run, select the *Send e-mail on repetitive run* check box.

5. Select the OK button.

Note that the e-mail is sent when the trigger occurs and not after the logic analyzer's acquisition memory is full.

You only need to specify one *send e-mail* action per trigger sequence. As long as one trigger action sends e-mail, any trigger in the sequence will result in e-mail being sent. (You cannot specify different *send e-mail* setups in a trigger sequence.)

If the SMTP server has a problem with the default sender address

- You may need to specify a sender address that is recognizable by the server. A possible address might be the one specified in the *To:* field.

Message Format   The automatically generated text is shown as follows:

Example: *system14 : Slot C : Analyzer C has triggered*

Where *system14* is the analysis system IP address or alias you have assigned to it; *Slot C* is the frame slot the module is in; *Analyzer C* identifies the specific analyzer module from others when configured in a multi-module frame configuration.

Any text you add in the text entry area of the e-mail setup dialog will appear after the automatically generated text.

What is SMTP? SMTP (Simple Mail Transfer Protocol) is a TCP/IP protocol used in sending and receiving e-mail. A protocol is the special set of rules for communicating the end points in a telecommunication connection as they send signals back and forth.

Protocols exist at several levels in a telecommunication connection. There are hardware telephone protocols. There are protocols between the end points in communicating programs within the same computer or at different locations. Both end points must recognize and observe the protocol.

On the Internet, there are the following TCP/IP protocols:

- TCP (Transmission Control Protocol), which uses a set of rules to exchange messages with other Internet points at the information packet level.

- IP (Internet Protocol), which uses a set of rules to send and receive messages at the Internet address level.

- HTTP, FTP, SMTP and other protocols, each with defined sets of rules to use with other Internet points relative to a defined set of capabilities.

To view a picture of the trigger sequence

1. In the Trigger tab, select the Overview subtab.

   A picture of the trigger sequence is shown.

See Also   "Editing the Trigger Sequence (Timing or 200, 400 Mb/s State Only)" on page 78

To clear the trigger sequence

1. When the Trigger tab is displayed, select *Trigger Sequence* or *All* from the Clear menu.

   Selecting *Trigger Sequence* restores the default trigger sequence for the selected sampling mode.

   Selecting *All* restores the default trigger sequence, trigger settings, and default storing if in the state sampling mode.

To restore default trigger settings

1. When the Trigger tab is displayed, select *Settings* from the Clear menu.

   Settings (acquisition depth and trigger position) are returned to their defaults. In the state sampling mode, time tags are turned back on. In the timing sampling mode, the sample period returns to its fastest setting.

Editing Advanced Trigger Functions (Timing or 200 Mb/s State Only)

After you break down a trigger function (if it didn't quite provide the trigger capability you need), or after you select one of the advanced trigger functions, you're ready to edit the advanced trigger function.

All trigger functions look for *events* and, if those events are found, take *actions*.

Most often, the event is something that occurs on the probed signals

(label events), and the action is to trigger the logic analyzer. However, events can also test timer, counter, and/or flag values that are set up in the logic analyzer, and actions can include setting up timers, counters, and flags as well as specifying special store actions.

- "To specify a duration or occurrence count for events" on page 84

- "To insert a store action (state mode)" on page 85

- "To insert timer actions/events" on page 85

- "To insert counter actions/events" on page 86

- "To insert flag actions/events" on page 87

- "To insert a "reset occurrence counter" action" on page 89

- "To group events" on page 89

- "To use named events" on page 89

To specify a duration or occurrence count for events

When working with advanced trigger functions or when you break down other trigger functions, you can specify that an event be present for a certain amount of time, or occur in a certain number of samples, before the associated action is taken.

To specify a time duration for events (timing only)

1. In the Trigger tab's Trigger Sequence area, if the *present for* > button is not present, select the *occurs* button and choose *present for* >.

2. Enter a time duration value.

The event must be present for the specified period of time before the action is taken.

To specify a < duration, break down the *Find pattern present for* < *duration* trigger function. (It actually uses occurrence counts and four sequence levels.)

To specify an occurrence count for events (timing, 200, 400 Mb/s only)

1. In the Trigger tab's Trigger Sequence area, if the *occurs* button is not present, select the *present for* > button and choose *occurs*.

2. Enter an occurrence count value.

3. If the occurrence count is greater than 1, select whether the event should occur *consecutively* or *eventually*.

The event must occur the specified number of times before the action is taken.

To insert a store action (state mode)

You can insert store actions to override the default storage qualifier for the samples that cause actions to occur, and you can insert store actions to turn default storing on or off.

1. In the Trigger tab's Trigger Sequence area, select one of the action buttons (for example, Trigger or Goto), choose *Insert ACTION*, choose *Store*, and choose either *Store sample*, *Don't store sample*, *Turn on default storing*, or *Turn off default storing*.

You can use store actions to set up sequence level storage qualification.

See Also   "Storage Qualification" on page 249 in "Understanding Logic Analyzer Triggering" on page 240

"To specify default storing" on page 76

"To specify whether default storing is initially on or off" on page 78

To insert timer actions/events

Timers are like stopwatches. You can insert actions to start (from zero), stop (and reset), pause, or resume a timer. You can insert timer events in a different sequence level to test the value of a timer.

NOTE: No timer is available for the first *pod pair* assigned to a logic analyzer. For each additional pod pair assigned to the analyzer, an additional timer is available.

To insert a timer action

1. In the Trigger tab's Trigger Sequence area, select one of the action buttons (for example, Trigger or Goto), choose *Insert ACTION*, choose *Timer*, and choose either *Start from reset*, *Stop and reset*, *Pause*, or *Resume*.

To insert a timer event

Timer events are like other *events* in that they evaluate to either true or false.

1. In the Trigger tab's Trigger Sequence area, select one of the existing event buttons (for example, a label name, Anything, Timer, Counter, or Flag) and choose to insert or replace a *Timer*.

2. Select the timer number button and choose the number of the timer you want to test.

3. Select the operator button and choose either >= or <.

4. Enter the time value.

   The minimum value you can test a timer for depends on the timing/state analyzer configuration.

See Also   "To assign pods to the analyzer" on page 57

To insert counter actions/events

Global counters are available in the trigger sequence. You can insert actions to reset or increment a counter. You can insert counter events in a different sequence level to test the value of a counter.

To insert a counter action

1. In the Trigger tab's Trigger Sequence area, select one of the action buttons (for example, Trigger or Goto), choose *Insert ACTION*, choose *Counter*, and choose either *Reset* or *Increment*.

To insert a counter event

Counter events are like other *events* in that they evaluate to either true or false.

1. In the Trigger tab's Trigger Sequence area, select one of the existing event buttons (for example, a label name, Anything, Timer, Counter, or Flag) and choose to insert or replace a *Counter*.

2. Select the counter number button and choose the number of the counter you want to test.

3. Select the operator button and choose either >= or <.

4. Enter the counter value.

To insert flag actions/events

Flags can be used to signal between modules in the logic analysis system mainframe, an expansion frame, or in multiple frames connected with the multiframe module.

There are 4 flags that are shared across all connected logic analysis system frames. A flag may be driven or received by multiple modules.

Using flags, logic analyzer modules can communicate back and forth with each other multiple times during a data acquisition, both before and after their trigger events occur. (By comparison, the Intermodule window lets one module arm another module one time when its trigger occurs.)

By default, flags are cleared. You can insert *actions* to set, clear, pulse set, or pulse clear a flag. You can insert flag *events* in different logic analyzer modules to test whether a flag is set or clear.

NOTE: In all but the slowest state speed, the logic analyzer can check flags by inserting an *event*, but cannot change flag status with an *action*. Flag *actions* are not available when not using the slowest state speed.

A flag that is set by a module remains set until that module clears it. If multiple modules set the same flag, all of those modules must clear the flag before it becomes clear.

Flags can also be used to drive the logic analysis system's Port Out signal.

To insert a flag action

You can use the *Set/clear/pulse flag* trigger function to insert a flag action. When editing advanced trigger functions, follow these steps to insert a flag action:

1. In the Trigger tab's Trigger Sequence area, select one of the action buttons (for example, Trigger or Goto), choose *Insert ACTION*, choose *Flag*, and choose either *Set*, *Clear*, *Pulse set*, or *Pulse clear*.

   Flags in Pulse mode sit in the opposite state when not being pulsed. If you

insert a *Pulse set* action for a flag in one analyzer, you cannot insert a *Pulse clear* action for the same flag in a different analyzer.

NOTE: Within an analyzer, the same flag cannot be used in both Pulse and Level (Set/Clear) modes. If a flag action is inserted or modified with a different mode than other actions for the same flag, all actions for that flag will change to match the new mode.

2. If you chose *Pulse set* or *Pulse clear*, enter the width of the pulse.

   Pulse width is adjustable from 50 ns to 1.275 us in 5 ns steps.

NOTE: Within an analyzer, a flag's pulse width must be the same in every action for that flag. Whenever the pulse width is changed in a flag action, it changes in all other actions for that flag.

3. Select the flag number button and choose the number of the flag you want the action to occur on.

To insert a flag event

Flag events are like other *events* in that they evaluate to either true or false.

You can use the *Wait for flag* trigger function to insert a flag event. When editing advanced trigger functions, follow these steps to insert a flag event:

1. In the Trigger tab's Trigger Sequence area, select one of the existing event buttons (for example, a label name, Anything, Timer, Counter, or Flag) and choose to insert or replace a *Flag*.

2. Select the flag number button and choose the number of the flag you want to test.

3. Select whether you're testing if the flag is *Set* or *Clear*.

There is approximately 100 ns of delay before a flag action can be seen by a flag event.

To drive the Port Out signal with a flag

1. In the main logic analysis system window, select the *Port Out* button.

2. In the Port Out dialog, select the *Type*, *Polarity*, and *Output* options.

When driving the Port Out signal with a flag, you can select the
Feedthrough type to pass the current state of the flag (set or clear)
directly to Port Out.

3. For the *Armed by* option, select the flag that will drive the Port Out signal.

4. *Close* the Port Out dialog.

5. Insert a flag action in one of the logic analyzer modules to drive the flag.

There is approximately 100 ns of delay between a flag action and the
signal on Port Out.

To insert a "reset occurrence counter" action

You can reset an occurrence counter if some event occurs by inserting
a "reset occurrence counter" action.

1. In the Trigger tab's Trigger Sequence area, select one of the action buttons
   (for example, Trigger or Goto), choose *Insert ACTION*, and choose *Reset
   occurrence counter*.

See Also   "To specify a duration or occurrence count for events" on page 84

To group events

When you are working with advanced trigger functions (or when you
break down other trigger functions) and there are multiple events in an
event list, you can specify their evaluation order by grouping the
events.

1. In the Trigger tab's Trigger Sequence area, select the *If*, *If not*, *Else if*, or
   Else if not button, and choose *Group events*.

2. In the Group Events dialog, either select the *Add parens* button to group
   events or select the *Remove parens* button to ungroup events.

3. Select the OK button.

To use named events

When you are working with advanced trigger functions (or when you
break down other trigger functions), you can name an event list and
use it later when inserting or replacing events.

To give an event list a name

1. In the Trigger tab's Trigger Sequence area, select the *If*, *If not*, *Else if*, or *Else if not* button, and choose *Name event list*.

2. In the Name Event List dialog, enter the name and select the OK button.

To insert a named event

1. In the Trigger tab's Trigger Sequence area, select a label name button and choose to insert or replace a *Named event*.

2. In the Named Event selection dialog, select the named event, and select the OK button.

To edit a named event

1. In the Trigger tab's Trigger Sequence area, select the named event button and choose *Edit locally* or *Edit globally*.

   Locally means to edit (and rename) this instance of the named event. Globally means to edit all instances of the named event.

2. In the Edit dialog, edit the event list as you would edit it in the Trigger tab's Trigger Sequence area.

Saving/Recalling Trigger Setups

You can save a trigger setup within a session by using trigger save/recall.

- "To save a trigger setup" on page 90
- "To recall a trigger setup" on page 91
- "To clear the trigger save/recall list" on page 91

See Also   "Save/Recall Subtab" on page 191

To save a trigger setup

1. Set up the trigger.

2. In the Trigger tab's Save/Recall subtab, select the Save button.

3. Select a memory location to store the trigger setup in.

4. In the Buffer Name dialog, enter a descriptive name for the trigger setup.

To recall a trigger setup

1. In the Trigger tab's Save/Recall subtab, select the Recall button.

2. Choose the trigger setup from one of the previous measurements or one of the save/recall memories.

Recalling a trigger setup changes the trigger arming, memory depth, and trigger position as well as the trigger sequence. Recalling a trigger setup will not change the sampling mode configuration.

If one of the settings in the recalled trigger setup conflicts with the sampling mode configuration, it will be set to the closest setting.

Also, if the trigger setup uses a trigger function library that does not exist on this mainframe, it will not load correctly.

To clear the trigger save/recall list

1. When the Trigger tab is displayed, select *Save/Recall Memories* from the Clear menu.

Running Measurements

After you set up a trigger, you're ready to run the logic analyzer measurement.

- "To start/stop measurements" on page 91
- "If nothing happens when you start a measurement" on page 92
- "To view the trigger status" on page 93

To start/stop measurements

To start measurements

1. Select the Run Single ▶, Run Repetitive ▶, Group Run Single ▶GROUP,

Group Run Repetitive, or Run All button.

Run starts only the instrument you are using. Single runs gather data until the logic analyzer memory is full, and then stop.

Repetitive runs keep repeating the same measurement and are useful for gathering statistics.

Group Run (or repetitive group run) starts all instruments attached to group run in the Intermodule window.

Run All starts all instruments currently placed in the workspace.

To stop a measurement

1. Select the Stop or Stop All button.

If nothing happens when you start a measurement

- Analyzers with deep memory take a noticeable amount of time to complete a run. Because data is not displayed until acquisition completes, it may look like nothing is happening. Check the Run Status window to see if the logic analyzer is still running.

- Messages such as "Waiting in level 1" may indicate you need to refine your trigger.

- If the status shows as "Stopped", the analyzer either finished the acquisition, or was unable to run. The cause of the problem is listed in the bottom half of the Run Status window.

- Look for an error message in the *message bar* at the top of the window. Common messages are "slow or missing clock" and "Waiting for trigger".

- If *Run* briefly changed to *Stop* or *Cancel*, select the *Window* menu, choose the logic analyzer's slot, then choose the Waveform or Listing display.

See Also   "Slow or Missing Clock" on page 221

"Waiting for Trigger" on page 225

"Error Messages" on page 211

To view the trigger status

While a logic analyzer measurement is running, you can view the trigger status to see the sequence level that is evaluating captured data, occurrence and global counter values, and flag values.

1. In the Trigger tab, select the Status subtab.

See Also   Run status (in the system help volume).

Displaying Captured Data

Once you have run a measurement and filled the logic analyzer's acquisition memory with captured data, you can display the captured data with one of the display tools.

You can use analysis tools to filter data and compare data sets.

You can also analyze captured data with toolsets like the Serial Analysis Toolset and the System Performance Analysis Toolset.

- "To open Waveform or Listing displays" on page 94

- "To use other display tools" on page 95

- "If the captured data doesn't look correct" on page 97

- "If there are filtered data holes in display memory" on page 98

- "To display symbols for data values" on page 99

- "To cancel the display processing of captured data" on page 100

To open Waveform or Listing displays

Waveform displays are typically used when data is captured with the timing sampling mode, and Listing displays are used when data is captured with the state sampling mode.

1. From the Window menu, select your logic analyzer and choose the *Waveform* or *Listing* command.
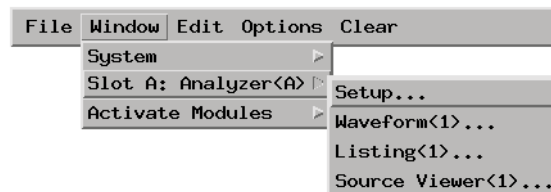
Waveform and Listing (and other) display tools provide global markers that can be used to correlate data that is captured by different instrument modules or displayed differently in other display tool windows.

The Waveform and Listing display tools also give you the ability to search for particular data values captured on labels.

Listing displays let you load inverse assemblers that will decode captured data into assembly language mnemonics. From the Listing display, you can also open Source Correlation Toolset (Source Viewer) windows that can display the high-level language source code that is associated with captured data.

See Also   Using the Digital Waveform Display Tool (see the *Waveform Display Tool* help volume)

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

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

To use other display tools

You can add display tools to your logic analyzer via the Workspace window.

1. Select the Workspace button (or from the Window menu, select System and Workspace).

2. In the Workspace window, scroll down to the Display portion of the tool icon list.
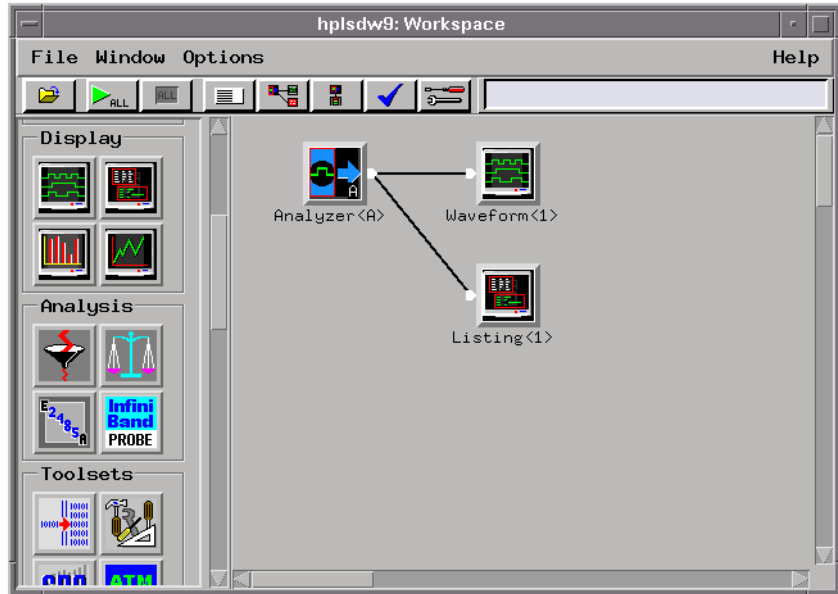
3. Drag the display tool icon and drop it on the analyzer icon.

4. To open the display tool, select its icon and choose the *Display* command.

You can use the Chart display tool to chart the data on a label over time. For example, if you use storage qualification (in the state sampling mode) or the Pattern Filter analysis tool, you can chart variable values.

You can use the Distribution display tool to show how often different values (among the possible values) are captured on a label.

You can use the Compare analysis tool to show the differences between two measurement data sets. For example, you can run a measurement on one circuit board, then run the same measurement on a different circuit board (or on the same circuit board in different environmental conditions), and compare the results.

You can use the Pattern Filter analysis tool to remove samples from a measurement data set before displaying or exporting the data. This lets you look at selected samples without having to re-capture data.

You can use the Serial Analysis toolset to convert streams of serial data into parallel words which are easier to view and analyze.

You can use the System Performance Analysis toolset to do things like: isolate the root cause of performance bottlenecks, measure function execution times, view the occurrence rate of an event, analyze bus occupation and bandwidth, analyze bus stability, analyze jitter or time dispersion, etc.

See Also   Using the Chart Display Tool (see the *Chart Display Tool* help volume)

Using the Distribution Display Tool (see the *Distribution Display Tool* help volume)

Using the Compare Analysis Tool (see the *Compare Tool* help volume)

Using the Pattern Filter Analysis Tool (see the *Pattern Filter Tool* help volume)

Using the Serial Analysis Tool (see the *Serial Analysis Tool* help volume)

Using the System Performance Analyzer (see the *System Performance Analyzer* help volume)

Measurement Examples (see the *Measurement Examples* help volume)

If the captured data doesn't look correct

Intermittent Data Errors   Check for poor connections, incorrect signal levels on the hardware, incorrect logic levels under the logic analyzer's Config tab, or marginal timing for signals.

Unwanted Triggers   If you are using an inverse assembler or a pipeline, triggers can be caused by instructions that were fetched but not executed. To fix, add the prefetch queue or pipeline depth to the trigger address.

The depth of the prefetch queue depends on the processor that you are analyzing, and can be quite deep.

Another solution which is sometimes preferred with very deep prefetch queues is to add writes to dummy variables to your software. Put the instruction just before the area you want to trigger on, then trigger on

the actual write to this variable. Although the instruction is prefetched, the analyzer can be set to only trigger when the write is executed.

Capacitive Loading on the Device Under Test

Excessive capacitive loading can degrade signals, resulting in suspicious data or even system lockup. All analysis probes add capacitive loading, as can custom probes you design for your device under test. To reduce loading, remove as many pin protectors, extenders, and adapters as possible.

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

If there are filtered data holes in display memory

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:

1. In the workspace, attach a Filter tool to the output of the analyzer.

2. In the Filter, select "Pass Matching Data"

3. In the filter terms, assure the default pattern of all "Don't Cares" (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.

To display symbols for data values

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. Select *Properties...*.

3. In the Properties dialog:

   - Set ShowValue to *On*.

   - Set Base to *Symbols* or *Line#*.

   - 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#*.

   The symbolic names for the values now appear instead of numeric data.

See Also   "Using Symbols" on page 101

To cancel the display processing of captured data

You can cancel the processing of captured data if it is taking too long.

1. Select the Cancel button.

Using Symbols

You can use symbol names in place of data values when:

- Setting up triggers

- Displaying captured data

- Searching for patterns in Listing displays

- Setting up pattern filters

- Setting up ranges in the System Performance Analyzer

Symbol names can be: variable names, procedure names, function names, source file line numbers, etc.

You can load symbol name definitions into the logic analyzer from a program's object file or from a general-purpose ASCII format symbol file, or you can define symbol names in the logic analyzer.

- "To load object file symbols" on page 102

- "To adjust symbol values for relocated code" on page 103

- "To create user-defined symbols" on page 104

- "To enter symbolic label values" on page 105

- "To create an ASCII symbol file" on page 106

- "To create a readers.ini file" on page 107

See Also   To go to a pattern in the Listing (see the *Listing Display Tool* help volume)

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

To define System Performance Analyzer state interval ranges (see the *System Performance Analyzer* help volume)

To load object file symbols

Object files are created by your compiler/linker or other software
development tools.

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 106).

3. Select the *Symbol* tab and then the *Object File* tab.

4. 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.

5. Specify the directory to contain the symbol database file (*.ns*) in the field
   under, *Create Symbol File (.ns) in This Directory*. Select *Browse...* if
   you wish to find an existing directory name.

6. In the *Load This Object/Symbol File For Label* field, enter the object file
   name containing the symbols. Select *Browse...* to find the object file and
   select *Load* in the Browser dialog.

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

7. If your program relocates code, see "To adjust symbol values for relocated
   code" on page 103.

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.

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.

The values of the object file symbols being used in the trigger sequence
or in SPA state-interval ranges will be updated automatically each time

the object file symbols are reloaded.

To delete object file symbol files

1. Select the *Symbol* tab, and then the *Object File* tab.

2. Select the file name you want to delete in the text box labeled, *Object Files with Symbols Loaded For Label*.

3. Select *Unload*.

See Also   "Symbol File Formats" on page 197

To adjust symbol values for relocated 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 adjust symbol values for a single section of code

1. Select the *Symbol* tab and then the *Object File* tab.

2. In the *Object Files with Symbols Loaded For Label* list, select the file whose symbols you wish to relocate.

3. Select the *Relocate Sections...* button.

4. In the *Section Relocation* dialog, select the field you wish to edit in the section list.

5. Enter the new value for that field and press Enter on your keyboard.

6. Repeat steps 4 through 6 above for any other sections to be relocated.

7. Select *Close*.

To adjust all symbol values

1. Select the *Symbol* tab and then the *Object File* tab.

2. In the *Object Files with Symbols Loaded For Label* list, select the file

whose symbols you wish to relocate.

3. Select the *Relocate Sections...* button.

4. Enter the desired offset in the *Offset all sections by* field. The offset is applied from the linked address or segment.

5. Select *Apply Offset*.

6. Select *Close*.

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 *Add*.

8. Repeat steps 3 through 7 for additional symbols.

9. You can edit your list of symbols by replacing or deleting them, if desired.

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.

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 *Load*.

   User-defined symbols that were resident in the logic analyzer when the configuration was saved are now loaded and ready to use.

To enter symbolic label values

When entering label values in the trigger sequence:

1. Choose the *Symbols* or *Line #s* number base.

2. Select the *Absolute XXXX* button.

3. 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 196) 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.

4. Select the symbol you want to use from the list of *Matching Symbols*.

5. If you are using object file symbols, you may need to:

   - Set *Offset By* (see page 196) to compensate for microprocessor prefetches.

   - Set Align to x Byte (see page 197) to trigger on odd-byte boundaries.

6. Select the Beginning, End, or Range of the symbol.

7. Select the *OK* button.

   The name of your symbol now appears as the value of the label.

8. Select the *Cancel* button to exit the *Symbol Selector* dialog without selecting a symbol.

NOTE: When you use symbolic label values in trigger sequences, information about the symbols is saved with the logic analyzer configuration (and in the trigger Save/Recall buffer). If you re-compile your program and reload the logic analyzer configuration (or reload symbols and recall a trigger), the symbol values are automatically re-calculated.

Triggers set up by the Source Viewer now use symbolic values and are re-calculated in the same way.

See Also   "To specify a label pattern event (Timing only)" on page 71

"Symbols Selector Dialog" on page 195

To create an ASCII symbol file

General-purpose ASCII symbol files are created with text editing/ processing tools.

See Also   "General-Purpose ASCII (GPA) Symbol File Format" on page 198

To create a readers.ini file

You can change how an ELF/Stabs, Ticoff 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
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 will 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
```

Printing/Exporting Captured Data

To print captured data

You can print captured data from display tool windows.

1. In the display tool window, select *Print this window* from the File menu.

To export captured data

You can use the File Out tool to save measurement data to an ASCII format file which can then be imported into a spreadsheet application, a debugger, or some other post-processing tool.

1. Select the Workspace button (or from the Window menu, select System and Workspace).

2. In the Workspace window, scroll down to the Utilities portion of the tool icon list.

3. Drag the File Out tool icon and drop it on the analyzer icon.

4. To open the File Out tool, select its icon and choose the *Display* command.

5. Select the file name, automatic file sequencing, and output file format options.

6. Select the *Save Data* button.

To re-import captured data

You can use the File In tool to re-import measurement data into the logic analysis system for further analysis.

1. Select the Workspace button (or from the Window menu, select System and Workspace).

2. In the Workspace window, scroll down to the Utilities portion of the tool icon list.

3. Drag the File In tool icon and drop it in the workspace.

4. To open the File In tool, select its icon and choose the *Display* command.

5. Select the file name and automatic file sequencing options.

6. Select the *Read File* button.

7. Drag display, analysis, or toolset icons and drop them on the File In tool icon to view the imported data.

See Also

Printing Windows - Configurations (see the *Agilent Technologies 16700A/ B-Series Logic Analysis System* help volume)

Using the File Out Tool (see the *File Out Tool* help volume)

Using the File In Tool (see the *File In Tool* help volume)

Cross-Triggering

An instrument must be armed before it can look for a *trigger*. By default, instruments are set to be armed immediately when you *Run* the measurement.

However, you can set an analyzer instrument to be armed by another instrument (in a different slot or frame).

- "To cross-trigger with another instrument" on page 112

To cross-trigger with another instrument

1. Select the Intermodule button (or from the Window menu pick, select System and Intermodule).

2. In the Intermodule window, select the desired instrument icon and arm it from the *Group Run*.

3. Select the next instrument icon and arm it from the first instrument.

When the logic analyzer waits for the arm signal (timing or 200 Mb/s state only)

1. In the Trigger tab's Trigger Sequence area, select the sequence level that should wait for the other analyzer's trigger.

2. In the *Trigger Functions* subtab, select the *Wait for arm in* trigger function; then, select either the *Replace* or *Insert before* button.

   (You can also use an advanced trigger function, edit it, and insert an *Arm in from IMB* event.)

NOTE: If the trigger sequence does not pass through the level containing the *Wait for Arm In* event, the logic analyzer will not wait for the arming signal.

When the logic analyzer waits for the arm signal (400, 800, 1250, or 1500 Mb/s state only)

1. In the existing trigger function, select the *Label* field; then, select *Insert Event* (before or after).

2. Choose the *Arm in from IMB* menu item.

When the logic analyzer drives the arm signal

1. Set up the logic analyzer trigger as you would normally.

   In the Trigger tab's Trigger Sequence area, the trigger actions will show *arm out* to indicate that the logic analyzer's trigger will drive the arm signal when it's trigger condition is satisfied.

2. Run the measurement.

See Also   Intermodule Window (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Group Run Arming Tree (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Solving Logic Analysis Problems

- "To test the logic analyzer hardware" on page 114

To test the logic analyzer hardware

In order to verify that the logic analyzer hardware is operational, run the Self Test utility. The Self Test function of the logic analysis system performs functional tests on both the system and any installed modules.

1. Disconnect all probes of the logic analyzer *module*.

2. If you have any work in progress, save it to a configuration file. (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

3. Disconnect all loads, adapters, or analysis probes from the probe cable ends.

4. From the system window, select the *System Admin* icon.

5. Select the *Admin* tab, then *Self Test...*.

   The system closes all windows before starting up Self Test.

6. Select *Master Frame*.

   If the module is in an expansion frame, select *Expansion Frame*.

7. Select the logic analyzer that you want to test.

8. In the Self Test dialog box, select *Test All*.

   You can also run individual tests by selecting them. Tests that require you to do something must be run this way.

If any test fails, contact your local Agilent Technologies Sales Office or Service Center for assistance.

See Also   Self Test (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

Agilent Technologies 16760A 1500 Mb/s State/800 MHz Timing Service Guide

Saving and Loading Logic Analyzer Configurations

The Agilent Technologies 16760A logic analyzer settings and data can be saved to a configuration file.

The configuration file will include references to any custom trigger libraries you have created, but if the configuration is loaded into an analyzer on a system that does not have the trigger libraries, they will not work correctly.

You can also save any tools connected to the logic analyzer. Later, you can restore your data and settings by loading the configuration file into the logic analyzer.

The Agilent Technologies 16760A logic analyzer does not load configurations from any other logic analyzers.

Note that while the *eye finder* (automatic sampling position adjustment) selected sampling positions are saved and loaded with logic analyzer configuration files, *eye finder* measurement data is not. You can save *eye finder* data to separate files (see "To automatically adjust sampling positions" on page 49).

To save logic analyzer configurations

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

To load logic analyzer configurations

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

4

Using the Logic Analyzer in Eye Scan Mode

- "Setting Up and Running Eye Scan Measurements" on page 119

- "Displaying Captured Eye Scan Data" on page 133

- "Saving and Loading Captured Eye Scan Data" on page 152

Setting Up and Running Eye Scan Measurements

The Eye Scan tab lets you set up and run Eye Scan measurements.

Eye Scan measurements sample small windows of time and voltage on logic analyzer data channels. The time windows are relative to a clock signal from the device under test. Captured measurement data is displayed as eye diagrams in the Eye Scan display.

Eye Scan measurements let you validate and characterize the data valid windows of signals latched by the clock. When an Eye Scan measurement shows a signal with setup/hold problems or some other anomaly, you can use a high-speed or high-bandwidth oscilloscope to examine the signal in greater detail.

- "To select channels for the eye scan" on page 119

- "To set the eye scan range and resolution" on page 120

- "To run an eye scan measurement" on page 121

- "To set advanced eye scan options" on page 121

- "To set up qualified eye scan measurements" on page 122

- "To comment on the eye scan settings" on page 132

See Also   "To make measurements on the eye scan data" on page 142

"Selecting the Eye Scan Mode" on page 55

"Displaying Captured Eye Scan Data" on page 133

"Saving and Loading Captured Eye Scan Data" on page 152

"Understanding Eye Scan Measurements" on page 259

To select channels for the eye scan

You may want to run the eye scan on channel subsets, for example,

when certain bus signals transition in one of the device under test's operating modes and other bus signals transition in a different operating mode.

1. In the *Eye Scan* tab, select the *Labels* subtab.

2. Select the bus/signal labels for the next eye scan measurement.

See Also   "Labels Subtab" on page 204

To set the eye scan range and resolution

Eye Scan looks at selected logic analyzer channels for signals passing through small windows of time (relative to clock signal edges) and voltage. You define these windows by setting the time and voltage ranges and resolution.

1. In the *Eye Scan* tab, select the *Scan Settings* subtab.

2. Select the scan range option.

    Coarse specifies relatively large windows of time and voltage. Because there are fewer windows to scan, the measurement runs faster.

    Medium specifies moderately sized windows of time and voltage.

    Fine specifies relatively small windows of time and voltage. Because there are more windows to scan, the measurement takes longer to run.

3. If desired, you can change the time range and resolution settings.

4. If desired, you can change the voltage range and resolution settings.

    Selecting the *Scan voltage over entire range of signal activity* option causes the voltage range to be determined during the measurement.

If the range and resolution settings have been changed, you can use the *Default* button to return to the default settings for the selected option.

See Also   "To run an eye scan measurement" on page 121

"Scan Settings Subtab" on page 205

"To set advanced eye scan options" on page 121

To run an eye scan measurement

1. After selecting the scan ranges and resolution (see "To set the eye scan range and resolution" on page 120), select the *Use Signals from Device Under Test* option.

    The *Use Demo Data (No probes required)* option is primarily for demonstration purposes only. You can use this mode to familiarize yourself with the eye scan display and measurement tools.

2. Select the Run Single button ▶.

    The Eye Scan display window opens, and the captured measurement data begins to appear.

    While the eye scan measurement runs, the Stop button becomes available.

    The estimated time of the measurement is shown in the status field.

NOTE: A group run will not run an eye scan measurement.

See Also   "To select channels for the eye scan" on page 119

To set advanced eye scan options

Eye scan measurements look at selected logic analyzer channels for signals passing through small windows of time and voltage.

Advanced options let you specify the number of clocks to process in each window and whether measurement data should be accumulated or replaced.

1. In the *Eye Scan* tab, select the *Advanced* subtab.

   Select *Quick Scan* for a relatively fast Eye Scan.

   Select *Complete Scan* if you are concerned about capturing noise and signal anomolies that lie outside the area of the most regular signal activity. *Complete Scan* will take significantly longer than *Quick Scan*.

2. Select the number of clocks to be processed at each scan point option.

   You can select *Short* when there are frequent transitions on all channels, *Medium* when the channels transition at a normal rate, or *Long* when there are sporadic transitions on some of the channels.

   The *Custom* option lets you enter the number of clocks to be processed at each scan point.

3. Select whether results should be accumulated from scan to scan or whether they should replace the previously scanned data.

See Also   "To run an eye scan measurement" on page 121

"Advanced Subtab" on page 206

To set up qualified eye scan measurements

A bus may have different signal timing requirements based on some condition. For example, a bi-directional bus may have different timing requirements for read transfers versus write transfers.

Eye scan can be configured to use a qualifier signal to tell it when to collect data and when to stop collecting data.

For example: a read/write qualifier signal in a DDR (Double Data Rate) system could be used so that data is collected only on read cycles or only on write cycles.

Other situations in which a qualified eye scan may be necessary:

• Signal timing for reads is different than for writes.

• Different signal sources are used for reads than for writes. For example, a

memory device versus a memory controller.

Requirements   Qualified eye scans are performed with double edge clocks.

Qualified eye scans are available only in the 800 Mb/s mode of the 16760 logic analyzer.

When the qualifier is in use, there are six channels on the master card that cannot be measured by eye scan:

Pod 2, clock input (clock K) - the qualifier

Pod 2, channels 14 and 15

Pod 1, channels 0, 1, and 2

This limitation is due to the internal architecture of the analyzer.

You can not perform an eye scan on the J clock of the master logic analyzer card because this is the target system clock signal. It is used as the reference signal for the eye scan.

Signal Timing - Burst Transfers   The qualifier feature is designed for systems using double data rate bursted transfers. In these systems, data is transferred on each edge of the clock. In other words, there are two data transfers for each clock cycle - one on the rising edge and one on the falling edge. Figure 1 shows a burst with four clock cycles and eight data transfers. The clock cycle begins on the rising edge of the clock.

The direction of the transfer (read versus write) is typically set up via an earlier command sequence (not shown).
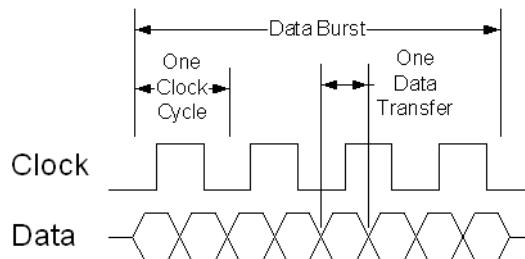
A suitable qualification signal is rarely available. Instead, an extra circuit (added to the SUT, in a probe adapter, or other convenient location) is usually required to decode read/write commands and generate the qualifier. The analyzer samples the qualification signal at the beginning of each clock cycle (i.e. at the first of each pair of data transfers). The analyzer can be configured to treat either the rising edge or the falling edge of the clock as the first edge of each clock cycle. The qualifier should remain stable for the entire duration of each burst.

Also, the qualifier must be pipelined (delayed) by one clock cycle before transmittal to the analyzer. The qualifier may be driven after the clock edge (i.e. you may use the output of a flip flop) since the analyzer has adjustable sampling positions on each input channel.

NOTE:

The time range available for eye scan when the qualifier is in use depends on the sampling position of the qualifier input in the analyzer. The range is increased as the position is made later (as the sample position moves from before the clock edge to after it).

The available eye scan range in 800 Mbs eye scan mode always begins at -4 ns and goes through +8 ns + tQualSample, where tQualSample is the qualifier's sample position as shown in the eye finder display (see Analyzer Setup, below). The adjustment range for tQualSample is from -2.5ns through +2.5 ns.

Putting it all together, the figure below shows the qualifier timing for a pair of eight-transfer data bursts, the first of which is to be qualified for eye scan and the second of which is to be ignored. The example shows a system with a clock cycle beginning with a rising edge and a high-true qualifier level. The qualifier signal to the analyzer is assumed to be the output of a pipeline register and is therefore sampled by the analyzer after the rising edge of the clock.
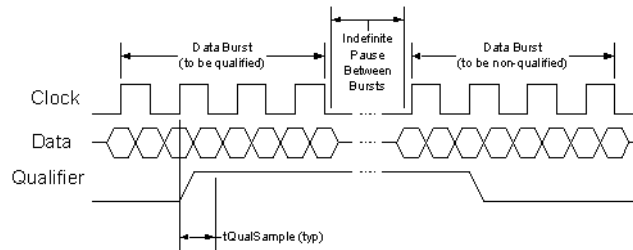
The qualifier is changed on the second clock cycle of the burst. It remains stable until after the next clock cycle following the end of the burst. If the clock pauses between bursts (as shown above), then the qualifier level for the previous burst remains in place until the second clock cycle of the next burst.

If the clock free-runs between bursts, then the qualifier should be taken false on the cycle following the end of the burst, and returned true on the second cycle of the next burst (if that burst is to be qualified for eye scan). If each burst is only one clock cycle (two data transfers), then the qualifier will be changing on the first edge of each burst, but the level will be for the previous burst. Variable size bursts are supported.

NOTE: If you have more than one logic analyzer card, the following procedure applies to the master card. The master card is the one that receives the clock signal from the target system on its J clock input.

Procedure   **First, set up the sampling position for the qualifier**

1. Connect the qualification signal from your target system to the Pod A2 clock (the K clock) input of the master logic analyzer card.

2. On the *Sampling* tab, select the *State Mode* button, and select *800 Mb/s* acquisition mode.
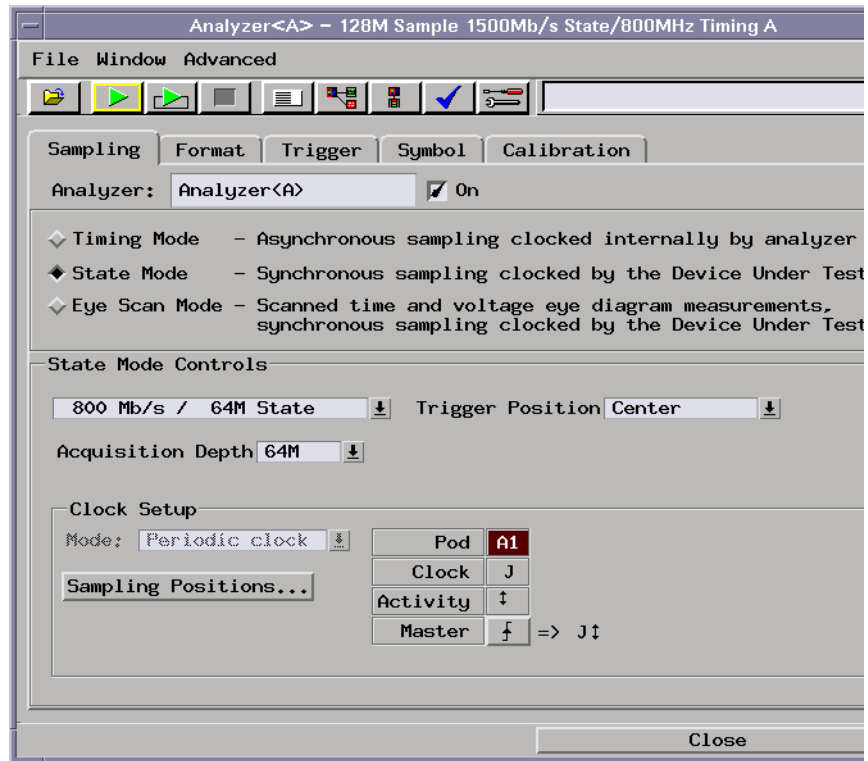
3. If you have only one 16760 logic analyzer card, on the *Trigger* tab, select
    the *Settings* subtab, and set *Count* to *Off*. This makes the second pod
    available for use by the qualifier. If you have a multi-card module (more
    than one 16760 card), an unassigned pod can be used to store time tags.
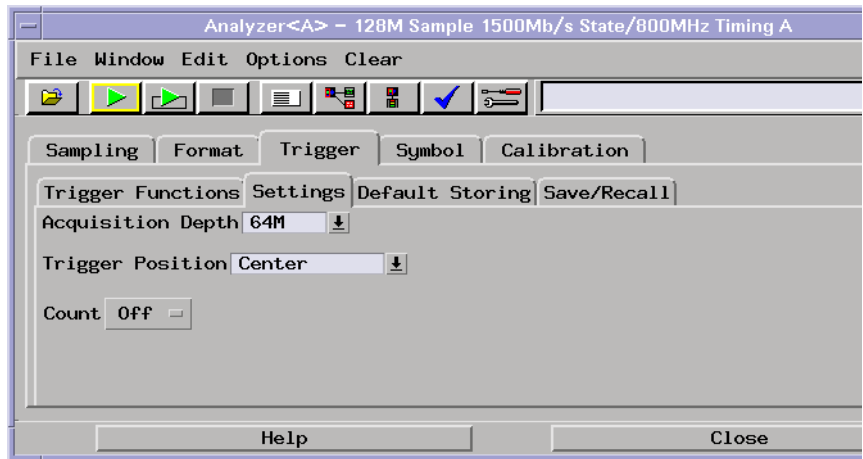
4. Assign pod A2 to the analyzer:

   - In the analyzer's *Format* tab select the *Pod Assignment* button.

   - Drag the *A2* pod from the *Unassigned Pods* section to the *Analyzer* section.

   - Select *Close*.

5. Define a label for the K clock on the master card (the qualifier input). This will make it easier to see which signal is used as the qualification signal. The qualifier can be in its own label and/or it can be part of a multi-channel label. The label will be displayed on the eye scan *Qualifier* tab.

NOTE: Be sure that the thresholds are set correctly!
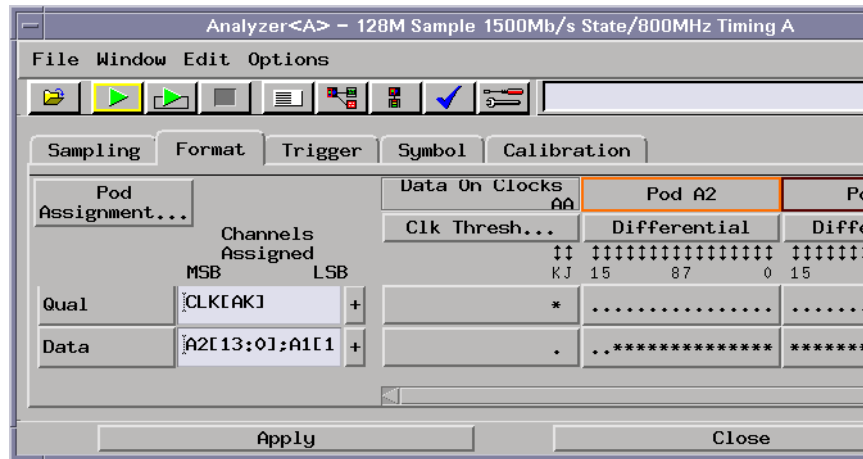
6. Select *Sampling Positions...* to open the Sampling Positions dialog. Select the label you created and choose *Expand* to display all the signals (if it is not already expanded).

7. Start your target system and select *Run Eye Finder*.

8. When Eye Finder completes its run, move the qualifier sampling position to the right of the qualifier transition region at or to the right of the clock position (t = 0). The sampling position is shown by the blue line in the display.

NOTE: The sampling positions for data are not used by eye scan. They should be set for correct state mode acquisitions.
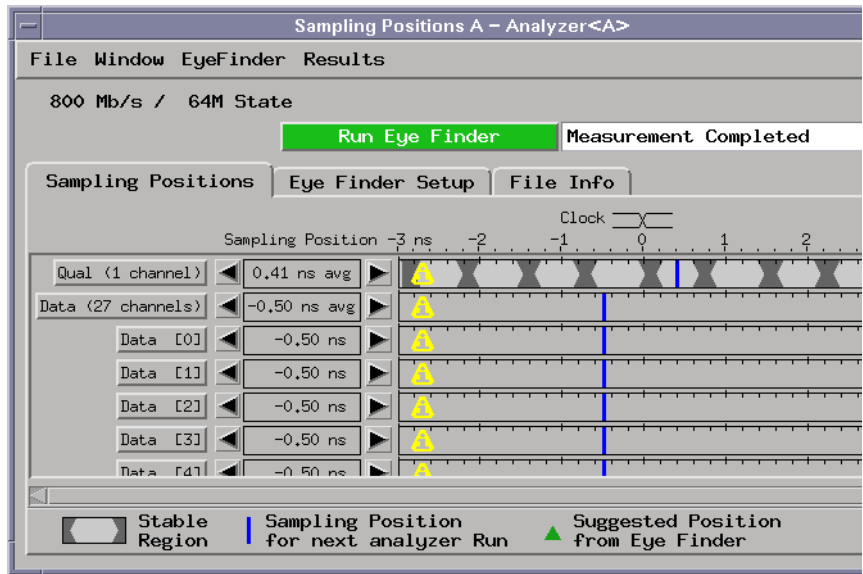
Now, set up eye scan mode

1. Select the *Eye Scan* button on the *Sampling* tab. Ensure that eye scan is set to *800 Mb/s / Eye Scan* and the *Master* clock is set to *Both Edges*.
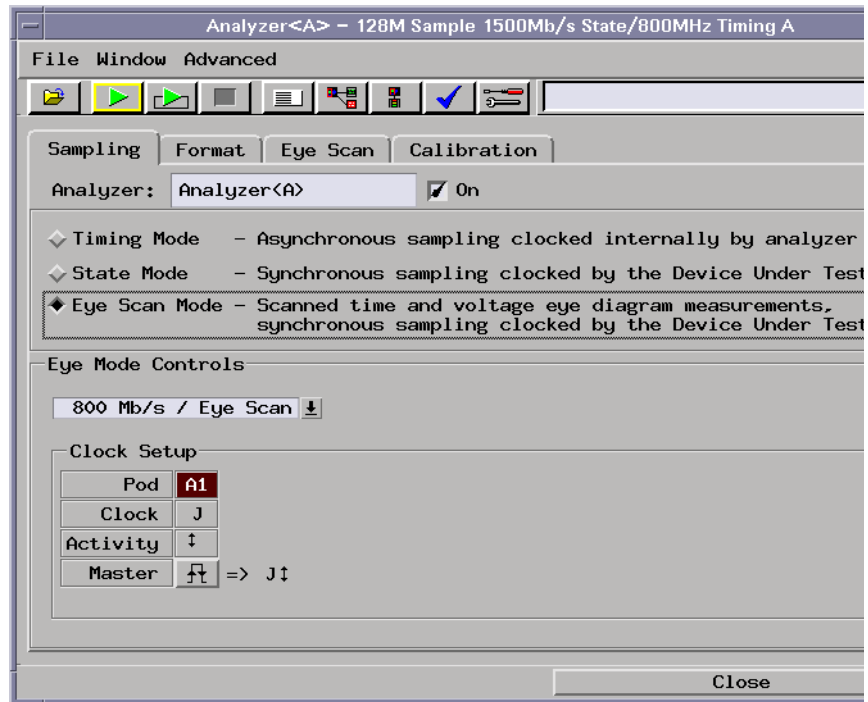
2. Select the *Eye Scan* tab.

3. Select the *Qualifier* subtab.
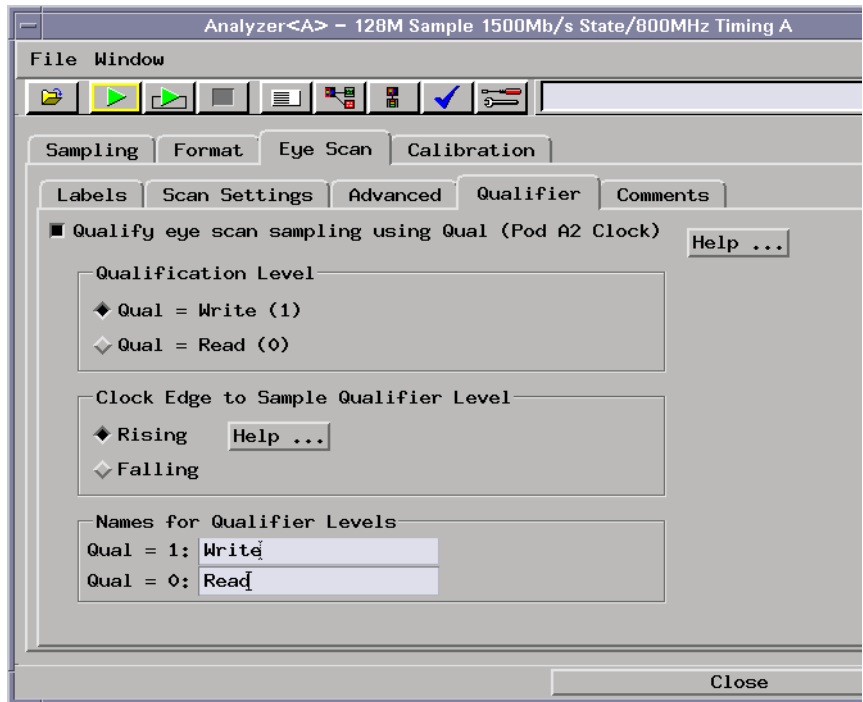
4. Select the *Qualify eye scan sampling using...* button.

5. Use the *Qualification Level* buttons to select whether eye scan data is collected when the qualifier signal is high or low. Note that the names "high" and "low" can be changed using the *Names for Qualifier Levels* section in this window.

6. Use the *Clock Edge to Sample Qualifier Level* buttons to select whether Eye Scan samples the qualifier signal level on the rising edge or the falling edge of the system clock.

7. (Optional) You can enter your own names for the qualifier signal levels using the *Names for Qualifier Levels* fields. For example: you can replace the word "high" with "read" and replace the word "low" with "write" if you are using a read/write line as your qualifier signal.

8. (Optional): You can enter comments on the *Comments* tab to describe the setup. The first comment line is included at the lower left of the eye scan display.

Make eye scan measurements

The logic analyzer is now ready to make qualified eye scan measurements.

Select the run icon to run the eye scan.

All other eye scan settings are available and are the same as they are in non-qualified measurements. To make non-qualified eye scan measurements, turn off the qualifier by clicking the box in the upper left of the "Qualifier" tab.

Qualification settings can be stored in a logic analyzer configuration file. You may want to save several configuration files, one for each qualification mode. For example, you could save one for writes, another for reads, and another for non-qualified, unidirectional signals such as address and command lines. Use the *Comments* tab to document the purpose of each configuration file.

See Also   "Qualifier Subtab" on page 208

"Displaying Captured Eye Scan Data" on page 133

To comment on the eye scan settings

You can enter your comments on the eye scan settings. When the logic analyzer configuration is saved, comments are saved along with the eye scan settings.

1. In the *Eye Scan* tab, select the *Comments* subtab.

2. Select the text entry box and type in your comments.

3. Select the *Apply comment to existing eye scan data* button.

The first line of the comment is included at the lower left of the Eye Scan display.

See Also   "To comment on the eye scan data" on page 151

Displaying Captured Eye Scan Data

Once you have run an eye scan measurement, captured measurement data is displayed as eye diagrams in the Eye Scan display.

The Eye Scan display has the following measurement tools: slope, limits, 4 point, 6 point, diamond, histogram, and cursors. You can also display measurement information in text format.

The Eye Scan display can be scaled, and there are several display options.

- "To open the Eye Scan display" on page 133

- "To select the channels displayed" on page 134

- "To scale the Eye Scan display" on page 135

- "To set Eye Scan display options" on page 136

- "To make measurements on the eye scan data" on page 142

- "To display information about the eye scan data" on page 149

- "To comment on the eye scan data" on page 151

See Also   "Selecting the Eye Scan Mode" on page 55

"Setting Up and Running Eye Scan Measurements" on page 119

"Saving and Loading Captured Eye Scan Data" on page 152

"Understanding Eye Scan Measurements" on page 259

To open the Eye Scan display

The Eye Scan display shows the results of the eye scan measurement.

1. From the Window menu, select your logic analyzer and choose the *Eye Scan...* menu item.

```
File  Window
      System                    ▷
      Slot A: Analyzer<A> ▷  Setup and Trigger...
      Activate Modules    ▷  Eye Scan...
                             Waveform<1>...
                             Listing<1>...
                             Source Viewer<1>...
```
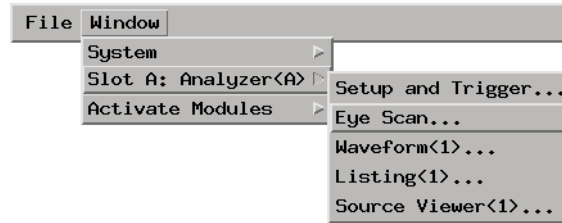
NOTE: There is no Eye Scan display tool icon in the Workspace window because the
                     Eye Scan display doesn't work with the standard form of captured logic
                     analyzer data (like, for example, the Waveform and Listing display tools do).
                     Instead, the Eye Scan display window is a sub-window of the instrument icon
                     (much like the Source Viewer window is a sub-window of the Listing display
                     tool).

To select the channels displayed

1. At the bottom left of the *Eye Scan* display window, select the label(s)
   whose channels should appear in the display.

```
Label <All Labels> ▾ ◀ ▶    Channel [15] Pod A1 Channel 15 ▾ ◀ ▶
```

2. At the bottom right of the *Eye Scan* display window, select the channel(s)
   whose data should appear in the display.

```
Label <All Labels> ▾ ◀ ▶    Highlight [15] Pod A1 Channel 15 ▾ ◀
```
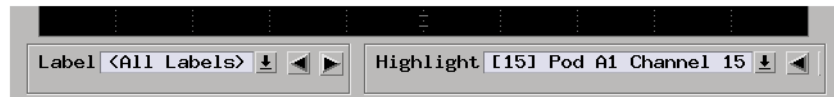
When channel highlighting has been selected (see "To set Eye Scan display
options" on page 136), this option chooses the channel that is highlighted
within the data captured on the bus.

See Also

To scale the Eye Scan display

The Eye Scan display's scaling options on the *Scale* tab let you change the display scale of the captured eye scan data. Also, you can click-and-drag to zoom in on a portion of the Eye Scan display.

To use the scaling options

1. In the *Eye Scan* display window, select the *Scale* tab.
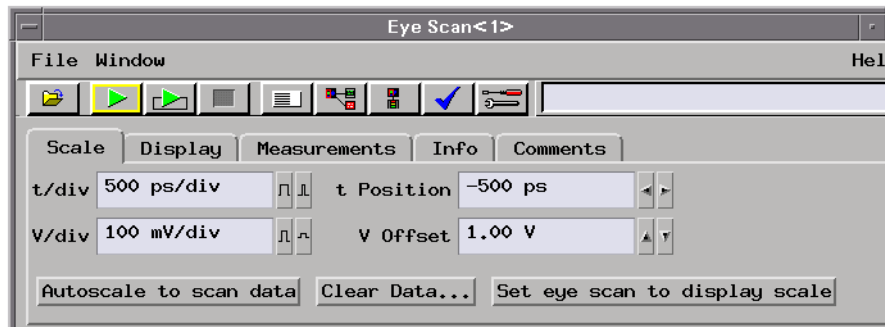
In the *Scale* tab, you can:

*   Select the *Autoscale to scan data* button to automatically scale the display so it shows the data that has been captured.

*   Select the *Set eye scan to display scale* button to display the *Scan Settings* dialog. The Eye Scan time and voltage display settings are copied to the *Scan Settings* window, and subsequent Eye Scan runs will use these settings.

*   Enter values in the *t/div* field or select the widen or narrow waveform buttons.

*   Enter values in the *V/div* field or select the heighten or shorten waveform buttons.

*   Enter values in the *t Offset* field or select the increase or decrease value buttons.

*   Enter values in the *V Offset* field or select the increase or decrease value buttons.

To zoom-in on the displayed data using the click-and-drag method

To zoom in on a selected portion of the Eye Scan display, select the upper left corner of the area you want to enlarge, drag down and to the right, then release the selection tool (mouse button or touch screen). The selected area is resized to fill the display and the Eye Scan time and voltage settings in the *Scan Settings* dialog are changed to new values based on the display.

The *Undo Zoom* button will appear in the upper left corner of the display, and will disappear after approximately ten seconds.

To clear the captured eye scan data

1. Select the *Clear Data* button.

To quickly set up another measurement using the scale

1. Select the *Set eye scan to display scale* button.

This button lets you quickly set up another eye scan measurement on a particular area of interest.

See Also

"To open the Eye Scan display" on page 133

"To set the eye scan range and resolution" on page 120

To set Eye Scan display options

The Eye Scan display options let you change the characteristics of the display.

1. In the *Eye Scan* display window, select the *Display* tab.
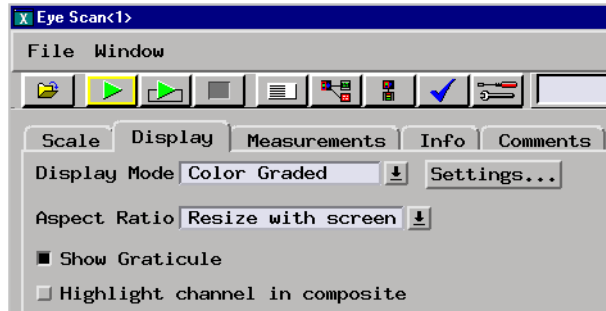
In the *Display* tab, you can:

- Change the *Display Mode*.

The *Gray Scale* option shows the measurement data in gray-scale where the brightness of a region indicates the number of transitions detected in that region.
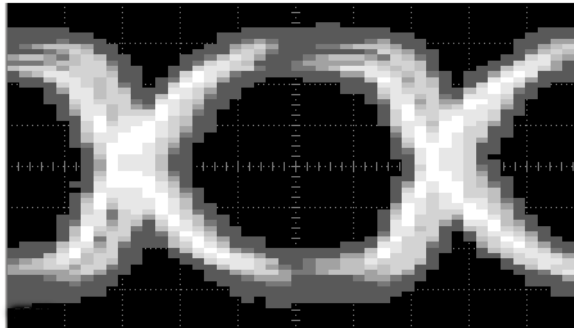
The *Color Graded* option shows the measurement data in color where the color of a region indicates the number of transitions detected in that region.
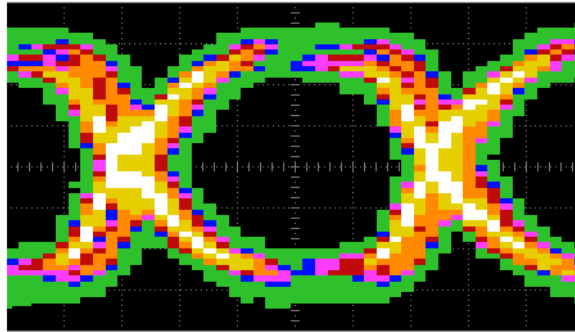
The *Solid Color* option shows measurement data as a solid color in all regions where transitions were detected.
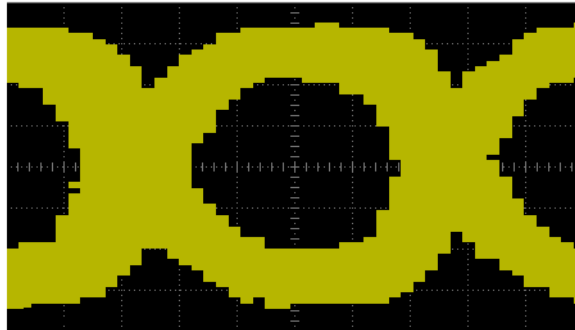
The *Ch Density Gray* option shows the measurement data in shades of gray where the brightness of a region indicates the number of logic analyzer channels which have transistioned in that region.

The *Ch Density Color* option shows the measurement data in color where the color of a region indicates the number of logic analyzer channels which have transitioned in that region.
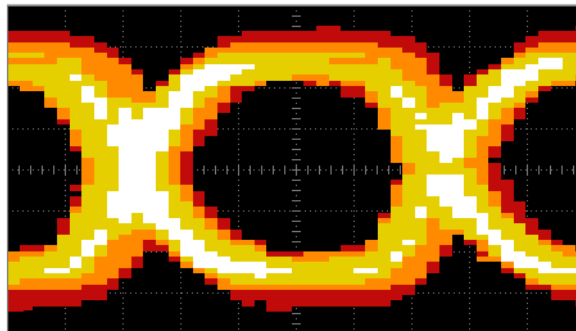
2. Change the Display Mode settings. See "To change the Eye Scan color scale" on page 140.

3. Change the display's *Aspect Ratio*. The options let you choose from several fixed aspect ratios or a display that is resized with the window. To make eye diagram shape comparisons easier, you can choose an aspect ratio that matches your oscilloscope.

4. Choose to *Show Graticule*. Suppressing the graticule makes it easier to see the on-screen indicators used in the measurement tools.

5. Choose to *Highlight channel in composite*. In this mode, the selected channel is displayed in solid white over the composite data from all channels in the label.
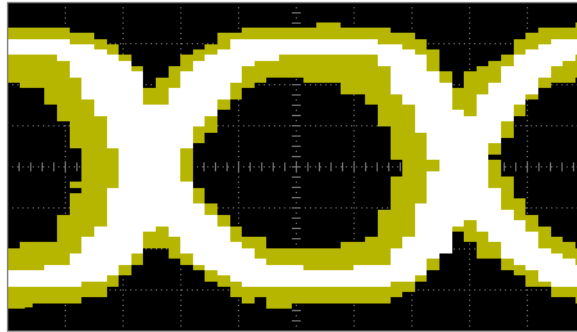
See Also

"To open the Eye Scan display" on page 133

"To scale the Eye Scan display" on page 135

"To change the Eye Scan color scale" on page 140

"To select the channels displayed" on page 134

To change the Eye Scan color scale

Color can be used to differentiate aspects of the eye scan display for increased readability.
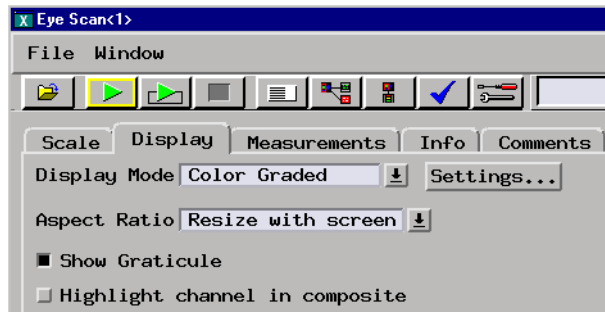
1. In the *Eye Display* window, select the *Display* tab.

2. Select the *Settings...* button.

   The *Color Scale* dialog will appear.
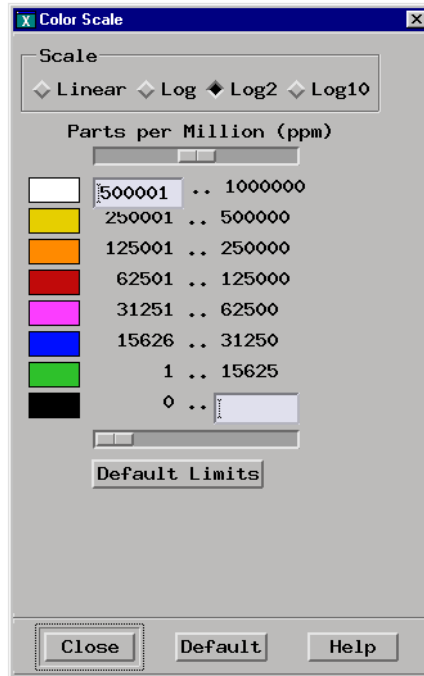
3. Select the *Linear* button to equally divide the color scale so that each color represents an equal increment of transitions.

   Select the *Log*, Log2, or Log10 scale buttons to separate the display of transitions detected using a logarithmic scale.

4. Use the *Parts per Million (ppm)* slider bars above and below the color bars to bring out detail in the eye diagram. The bottom slider bar controls the black level. Set the slider bar or enter a number in the text box associated with the black color bar to hide noise. The upper limit for black is zero by default.

NOTE: Measurement points that are excluded from the display using the value entered for the black color bar will not be reported by the measurement tools. To cause the measurement tools to report all measurement points, enter 0 (zero) in the box for the black color bar in the *Color Scale* window. See "To make measurements on the eye scan data" on page 142.

Similarly, you can use the top slider bar or enter a number in the text

box associated with the white color bar to bring out detail in the eye diagram.

The *Default Limits* button can be used to return the black and white limit values to settings that should produce a typical eye diagram.

The *Default* button at the bottom of the window can be used to reset both the limits and the *Scale* mode at once.

To make measurements on the eye scan data

The Eye Scan display's measurement tools show detailed information about the captured measurement data.

NOTE: Measurement points that are excluded from the display using the *Color Scale* dialog will not be reported by the tools (4 Pt Box, Diamond, etc.) in the *Measurements* tab. In other words, measurement points that have been filtered out of the display are not included in the measurement statistics. See "To change the Eye Scan color scale" on page 140 for more information.

1. In the *Eye Scan* display window, select the *Measurements* tab.

   In the *Measurements* tab, you can:

   • Turn on the *Eye Limits* tool. This tool is for measuring the inner eye limits detected at the time and voltage coordinates of the cursor. This tool is a single point that can be repositioned.
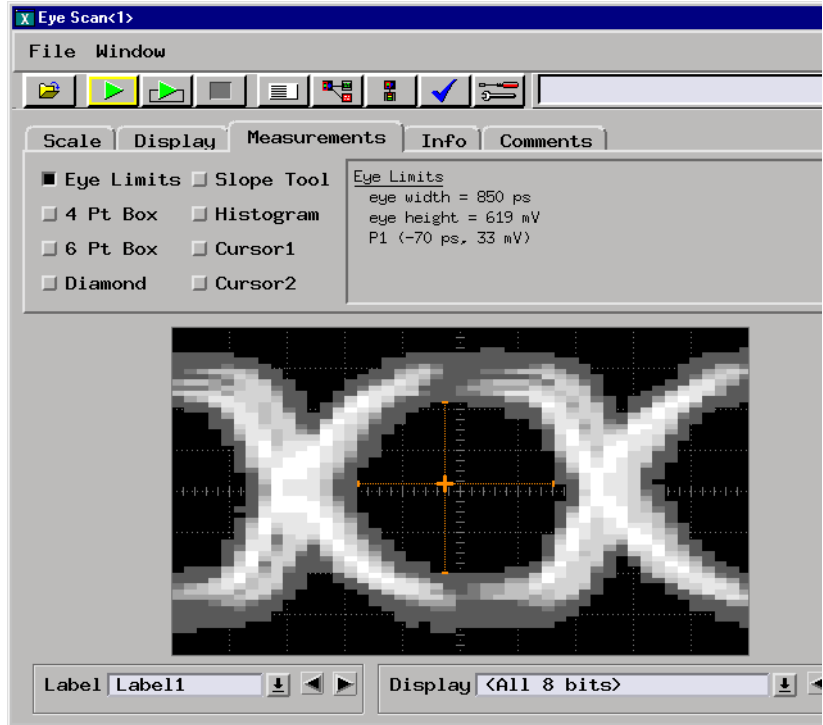
- Turn on the *4 Pt Box* tool. This tool is a box that displays time (box width), voltage (box height), and number of transitions detected. Each point of the box can be repositioned.
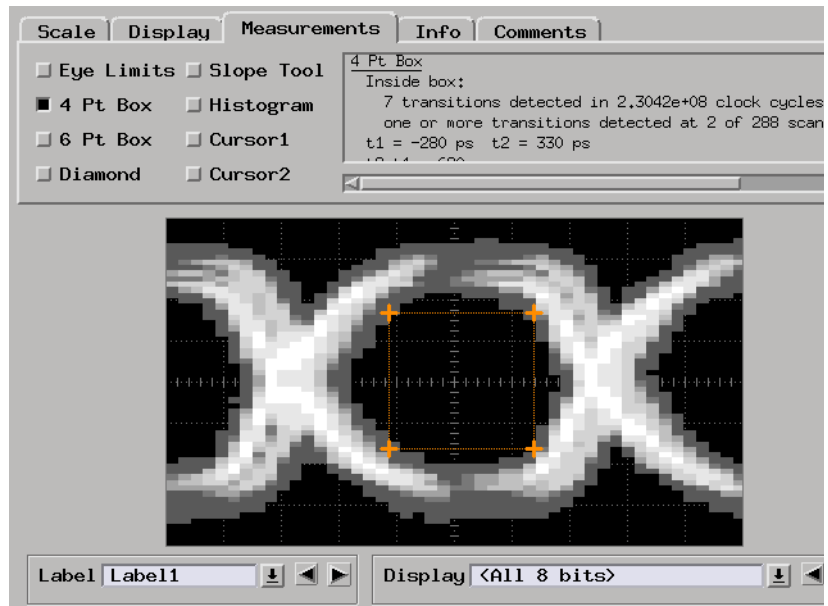
- Turn on the *6 Pt Box* tool. This tool is also for measuring time, voltage, and the number of transitions detected within the area. The *6 Pt Box* has 6 points that can be repositioned to match the shape of an eye. Because of the box's symmetry, it's easiest to position the box's upper and lower points before the middle points.
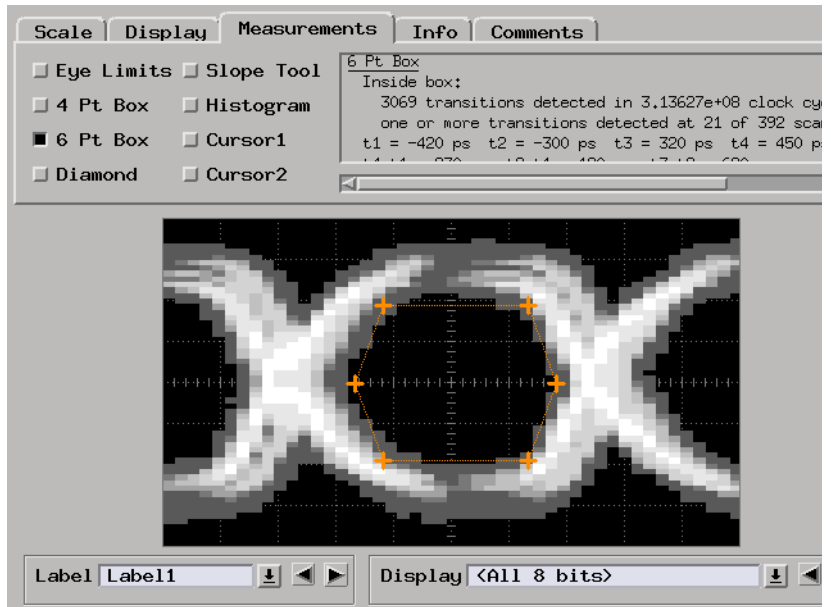
- Turn on the *Diamond* tool. This tool is also for measuring time, voltage, and number of hits. Number of hits refers to the number of transitions detected within the area. The *Diamond* tool has 4 points that can be repositioned to match the shape of an eye.
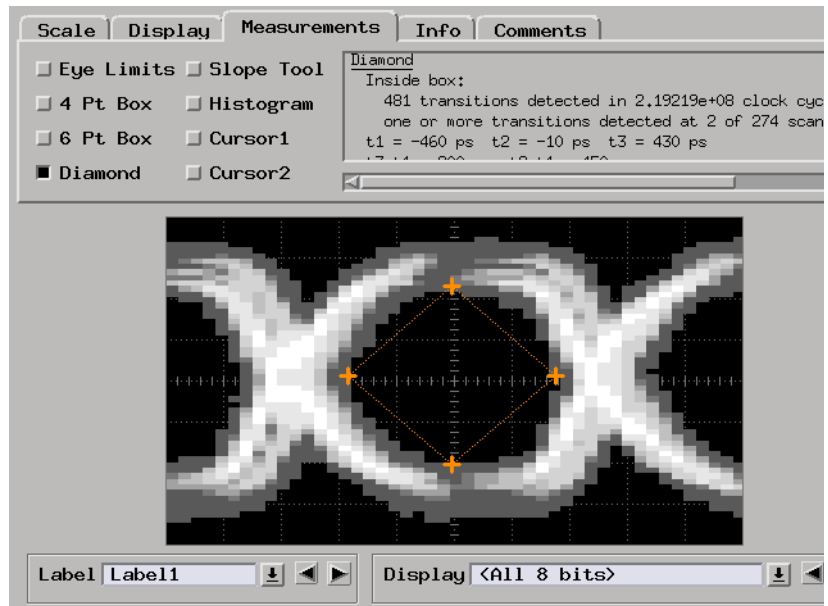
- Turn on the *Slope Tool* tool. This tool is for measuring the slope of rising and falling edges in an eye. It also shows the change in time and voltage between two points. The slope tool is a line with two endpoints that can be repositioned.
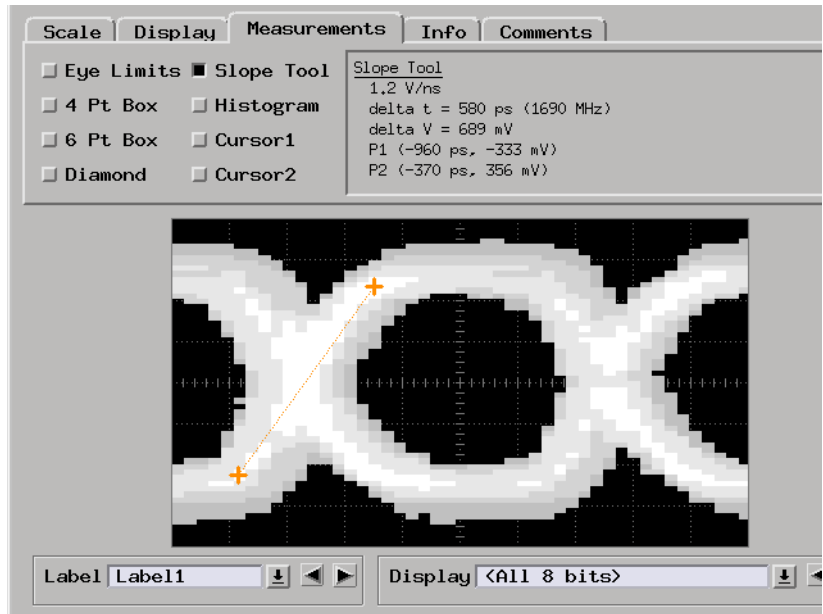
- Turn on the *Histogram* tool. This tool displays a histogram of the relative number of transitions at a particular voltage between two times. The histogram tool is a horizontal line in the display with two endpoints that can be repositioned. There is a slider bar on the line that can be positioned for exact measurements at a particular time. The histogram can be displayed using a linear or logarithmic scale. Note that the logarithmic scale is disabled when the display is in *Channel Density* mode (*Ch Density Gray* or *Ch Density Color.*)
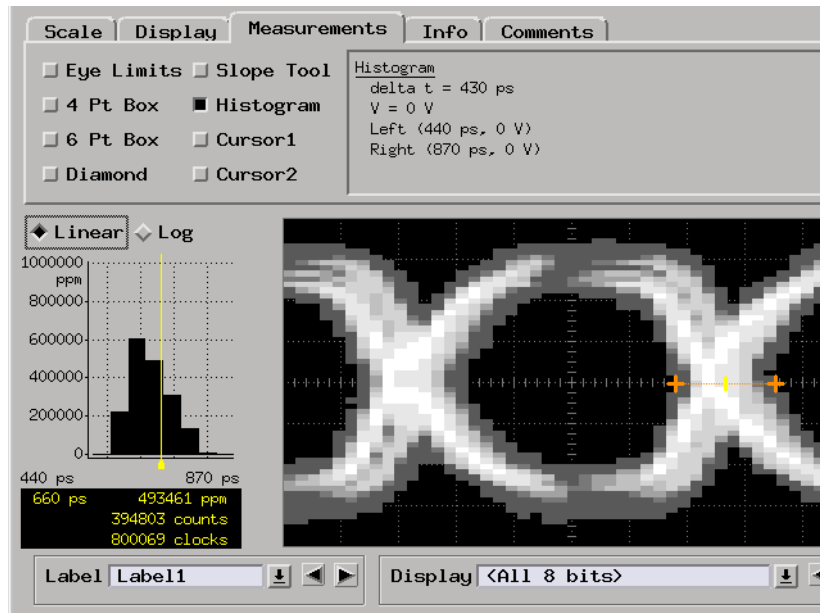
- Turn on the *Cursor1* or *Cursor2* tools. Cursors are single points that display the time and voltage where they are positioned. (To display the *change* in time and voltage between two points, use the *Slope* tool instead of two cursors.) Cursors also show the number of channels that contribute to the data at a particular point.
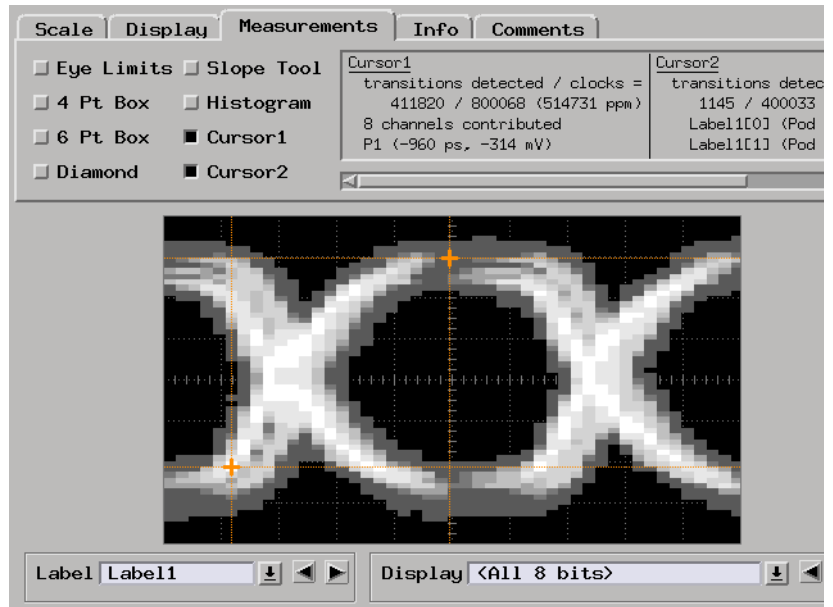
Any combination of the measurement tools can be used at the same time. Information for all selected tools is displayed. You can use the scroll bars to navigate through the information.

See Also

To display information about the eye scan data

1. In the *Eye Scan* display window, select the *Info* tab.
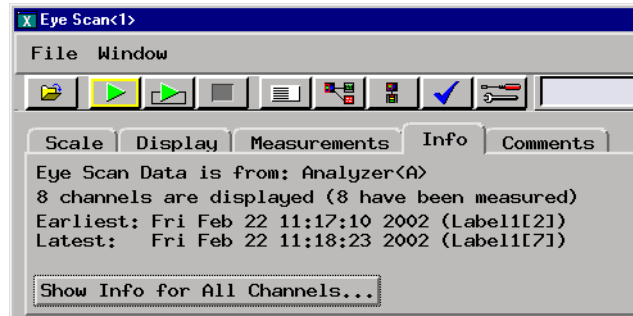
In the *Info* tab, you can:

• View information about which logic analyzer the Eye Scan data is from, how many channels are displayed, and when the data was acquired.

• Select the *Show Info for All Channels* button to display detailed information about the Eye Scan measurement.

Included in the detailed information is: the date and time of the last eye scan measurement, the time and voltage range and resolution settings that were used (see "To set the eye scan range and resolution" on page 120), the number of clocks that were processed at each scan

point (see "To set advanced eye scan options" on page 121), the number of scan points that were measured, and the amount of system memory required for the data.

When you save a configuration that includes data, *Mem size* also shows the amount of disk space required for the eye scan data.

See Also   "To open the Eye Scan display" on page 133

To comment on the eye scan data

You can enter your comments on the captured eye scan data. When the logic analyzer configuration is saved with data, comments are saved along with the captured eye scan data.

1. In the *Eye Scan* display window, select the *Comments* subtab.

2. Select the text entry box and type in your comments.

See Also   "To comment on the eye scan settings" on page 132

Saving and Loading Captured Eye Scan Data

When the logic analyzer configuration is saved *with data*, captured eye scan data is included. Eye scan data can be restored by loading a logic analyzer configuration file. For more information, see "Saving and Loading Logic Analyzer Configurations" on page 116.

See Also

"Selecting the Eye Scan Mode" on page 55

"Setting Up and Running Eye Scan Measurements" on page 119

"Displaying Captured Eye Scan Data" on page 133

"Understanding Eye Scan Measurements" on page 259

5

Reference

- "The Sampling Tab" on page 155
- "The Format Tab" on page 174
- "The Trigger Tab" on page 176

- "The Symbols Tab" on page 193

- "Error Messages" on page 211

- "Specifications and Characteristics" on page 227

The Sampling Tab
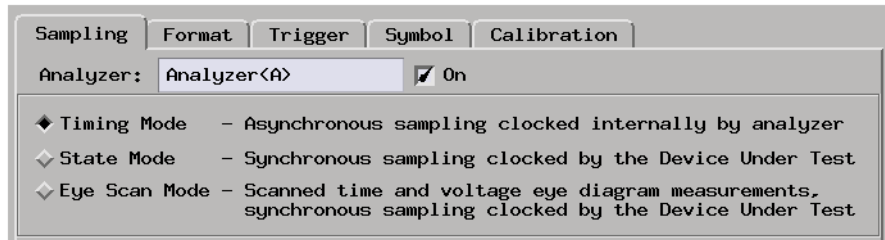
The Sampling tab lets you choose between the logic analyzer's:

- Asynchronous sampling Timing Mode

- Synchronous sampling State Mode

- Time and voltage scanning (relative to a clock from the device under test) Eye Scan Mode.

This tab also lets you set controls for the selected mode and adjust the logic analyzer sampling positions.

- "Timing Mode" on page 156

- "State Mode" on page 158

- "Sampling Positions Dialog" on page 159

    - "Eye Finder Setup Tab" on page 169

    - "Sampling Positions Tab" on page 160

    - "File Info Tab" on page 171

- "Eye Scan Mode" on page 172

Timing Mode

```
Sampling | Format | Trigger | Symbol | Calibration |

Analyzer:  Analyzer<A>          ☑ On

◆ Timing Mode   – Asynchronous sampling clocked internally by analyzer
◇ State Mode    – Synchronous sampling clocked by the Device Under Test
◇ Eye Scan Mode – Scanned time and voltage eye diagram measurements,
                  synchronous sampling clocked by the Device Under Test

┌Timing Mode Controls──────────────────────────────────────┐
│  800 MHz / 64M Sample Conventional                    ⬦ │
│                                                          │
│  Trigger Position Center            ⬦                    │
│                                                          │
│  Acquisition Depth 64M     ⬦                             │
│                                                          │
│  Sample Period:  1.25 ns                                 │
└──────────────────────────────────────────────────────────┘
```
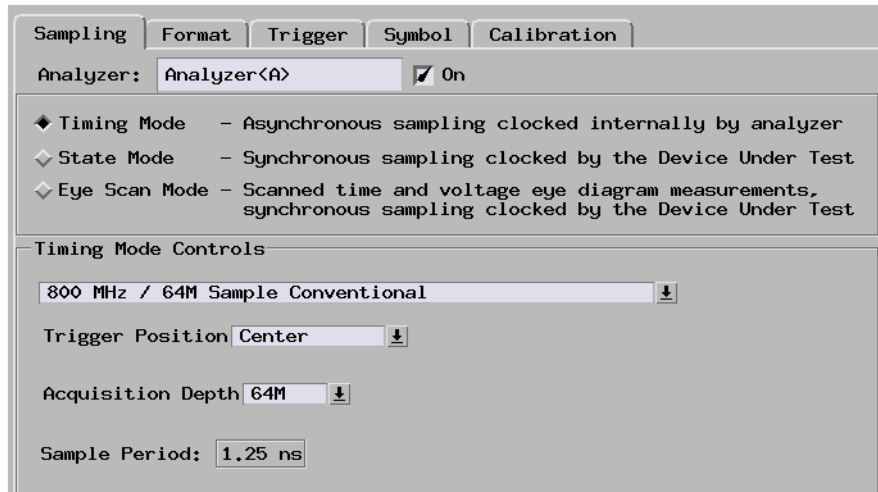
When you select Timing Mode, the Timing Mode Controls area appears.

**Conventional/
Transitional
Configuration**   Lets you configure the timing analyzer to use memory for
all samples (conventional) or just samples that are
different than previously stored sample (transitional).

Trigger Position Lets you specify where the sample that triggered the
analyzer should appear among all the other samples that
are stored in acquisition memory.

**Acquisition
Depth**   Lets you use a smaller portion of the acquisition memory
and speed up processing of the captured data.

Sample Period   Lets you specify how often the logic analyzer samples
signals from the device under test. In the *800 MHz / 64M
Sample Conventional* configuration, the sample period is
fixed at 1.25 ns.

See Also

How Samples are Stored in Transitional Timing

In the timing mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration, with the exception of the first sample, the logic analyzer must store two samples (four when sampling at 2.5 ns) so that no data is lost during the time it takes for the edge detectors to reset. This affects the number of transitions that can be stored in a given amount of logic analyzer memory.

When transitions occur more than two sample periods apart, only half of the logic analyzer memory will contain samples with transitions. This is illustrated below at time tags 2, 5, 7, 14, and 17. This is the minimum number of transitions that will be stored.
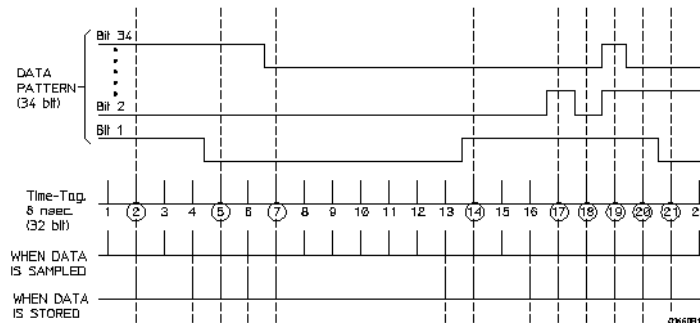
When transitions occur at every sample period, all of the logic analyzer memory contains samples with transitions (even though edges were only detected at every other sample). This is illustrated above at time tags 18, 19, 20, and 21. This is the maximum number of transitions that will be stored.

When transitions occur at a varied rate, sometimes at every sample period and other times more than two sample periods apart, somewhere between half and all of the logic analyzer memory contains samples with transitions.

See Also
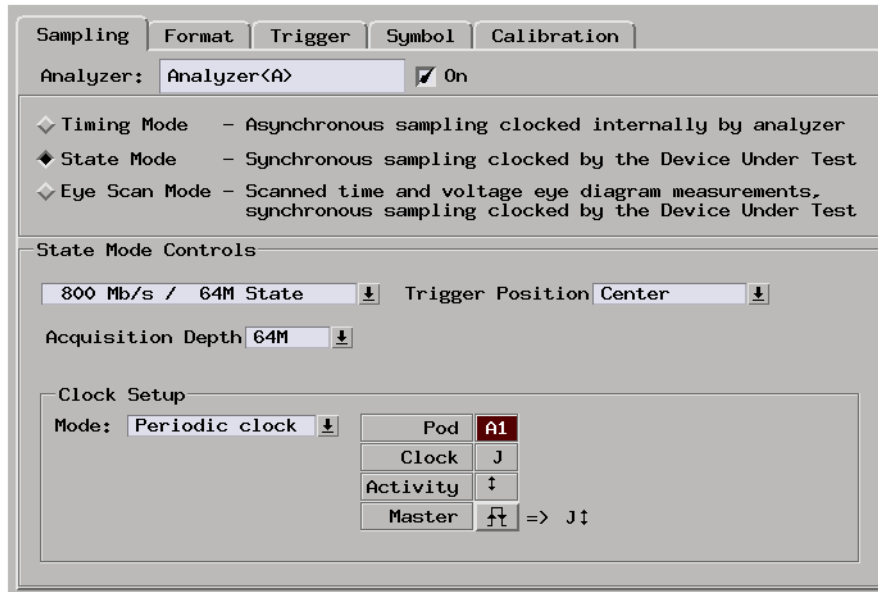
State Mode

When you select State Mode, the State Mode Controls area appears.

**State Speed
Configuration**   Lets you configure the state analyzer for faster sampling, but with half of the channels.

Trigger Position Lets you specify where the sample that triggered the analyzer should appear among all the other samples that are stored in acquisition memory.

**Acquisition
Depth**   Lets you use a smaller portion of the acquisition memory and speed up processing of the captured data.

Clock Setup   Lets you specify the clock mode. Set the mode to *Periodic* if the clock is a fixed frequency clock that is always running. Set the mode to *Aperiodic* if clock frequency varies or is bursted. The clock setup also lets you specify the clock edge from the device under test that will be used

as the sampling clock.

Generally, the state mode sampling clock is taken from the signals that clock valid data in the device under test.

See Also   "Selecting the State Mode (Synchronous Sampling)" on page 46

"In Either Timing Mode or State Mode" on page 53

Sampling Positions Dialog

The Sampling Positions dialog lets you position the logic analyzer's setup/hold window (or sampling position) so that data on high-speed buses is captured accurately, in other words, so that data is sampled when it is valid.

When the device under test's data valid window is less than 2.5 ns (roughly, for clock speeds >= 200 MHz), it's easiest to use *eye finder* to locate the stable and transitioning regions of signals and to automatically adjust sampling positions.

When the device under test's data valid window is greater than 2.5 ns (roughly, for clock speeds < 200 MHz), it's easiest to adjust the sampling position manually, without using the logic analyzer to locate the stable and transitioning regions of signals.

- "Sampling Positions Tab" on page 160
  - "Eye Finder Run Messages" on page 162
  - "Eye Finder Info Messages" on page 165
  - "Eye Finder Load/Save Messages" on page 167
- "Eye Finder Setup Tab" on page 169
  - "Eye Finder Advanced Settings Dialog" on page 170
- "File Info Tab" on page 171

See Also   "Understanding State Mode Sampling Positions" on page 256

"Selecting the State Mode (Synchronous Sampling)" on page 46

Sampling Positions Tab

The Sampling Positions display is a digital "eye" diagram in that it represents many samples of data captured in relation to the sampling clock. The transitioning edges measured before and after the sampling clock result in a picture that is eye-shaped.

You should have already specified the logic analyzer threshold voltage, but you may adjust it to maximize the width of the measured stable regions.

Eye finder measures the location of the stable region boundaries and places the logic analyzer's sampling position in the center of the stable region.
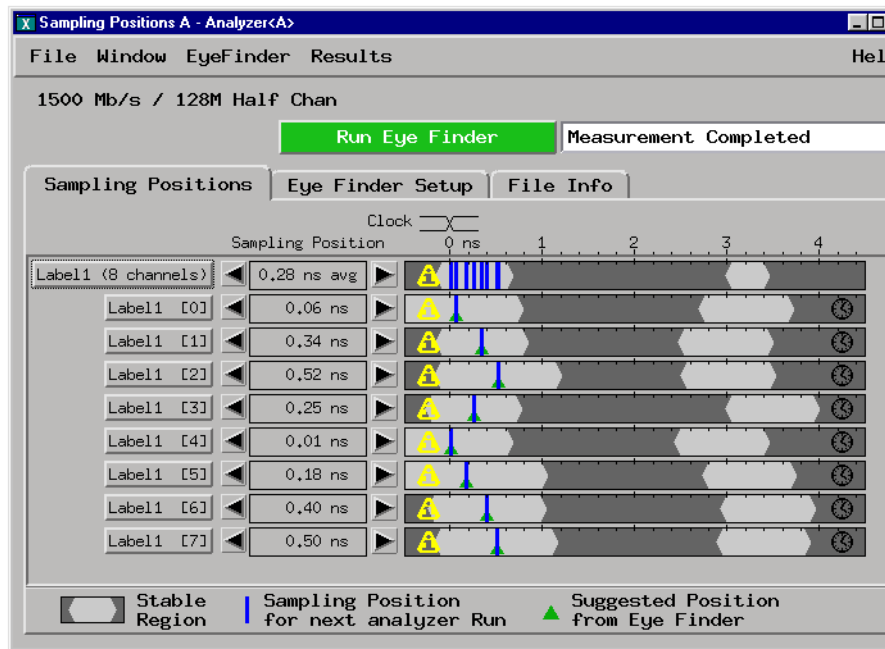
File menu   Lets you save/load *eye finder* data.

EyeFinder menu Lets you run *eye finder*, choose the run mode, and access the "Eye Finder Advanced Settings Dialog" on page 170.

Results menu   Let you expand/collapse the signals in a label, set the bus

view, set the sampling positions to the suggested sampling positions, and remove all *eye finder* data.

Label buttons   Let you expand/collapse the signals in a label, set the bus view, choose the suggested sampling position, and show message or time stamp information.

Display Area   Shows:

- Transitioning (dark) and stable (light) regions on the signals.

- Suggested sampling positions (green triangles).

- The current sampling positions (blue lines).

- Informational message icons . You can move the mouse pointer over the icon to cause the message to pop up.

- Time stamp icons . You can move the mouse pointer over the icon to see when the last *eye finder* measurement was run.

    To give you more information about the signals, the display covers +/-5 ns even though the sampling position may only be set to +/-3.25 ns.

**Sampling
Position**   Lets you adjust the sampling position. You can also drag the sampling position bar to a new location.

See Also   "Understanding State Mode Sampling Positions" on page 256

"How Selected/Suggested Positions Behave" on page 161

"Eye Finder Run Messages" on page 162

"Eye Finder Info Messages" on page 165

"Eye Finder Load/Save Messages" on page 167

"Eye Finder Setup Tab" on page 169

"To manually adjust sampling positions" on page 52

How Selected/Suggested Positions Behave. The *eye finder*'s selected and suggested sampling positions behave as follows:

How the Selected Position Behaves

1. When *eye finder* is enabled, the selected position (blue line) is set based on the manual setup/hold value.

2. Whenever the selected position is moved, the manual setup/hold value is also updated. They always track each other.

3. When the manual setup/hold is enabled again, the position changes made while *eye finder* was enabled can be kept or discarded.

NOTE: If *eye finder* changes are discarded, this includes any setup/hold settings loaded from a configuration file while *eye finder* was enabled.

4. The selected position is "snapped" to the suggested position (green triangle) each time the channel is measured.

How the Suggested Position Behaves

1. There is only a suggested position (green triangle) on channels that have been measured.

2. The suggested position is always in the center of the stable region closest to the selected position (blue line).

3. If the selected position is moved to a different stable region, the suggested position "hops" to the center of that region.

4. If a stable region is open-ended, the suggested position is placed 1.25 ns from the closed end (the visible boundary). If more than 1 clock edge is active, the suggested position is placed 1.5 ns from the closed end.

Eye Finder Run Messages. These messages can appear in the status area after you run a *eye finder* measurement.

"XX% complete"

The indicated percentage of the channels selected for the current *eye finder* measurement are completed.

"Cannot run the Eye Finder at this time."

A logic analyzer measurement is currently running. Stop the logic analyzer or wait for it to complete before running *eye finder*.

An *eye finder* measurement is currently running. Stop the *eye finder* or wait for it to complete before running the *eye finder*.

The *eye finder* is already running on the other machine defined for this analyzer. *Eye finder* cannot run on both machines at the same time.

"Cannot run the Logic Analyzer at this time."

An *eye finder* measurement is currently running. Stop the *eye finder* measurement or wait for it to complete before running the logic analyzer measurement.

Eye finder uses the same hardware as an ordinary logic analyzer measurement uses. Therefore, they cannot be performed at the same time.

"Characterizer cannot be loaded"

This is an internal error. It means that a software unit responsible for measuring one of the channels did not fit the hardware setup and the other characterizers which were already loaded. Try running *eye finder* on just one label (or one channel) at a time to attempt to clear it up. Contact support if this persists.

"Complete: DATE"

The measurement completed successfully on the date and time given. This time may also be accessed by expanding a label in the results display and selecting the clock icon or selecting "Show Time Stamp" on the popup menu raised by selecting the channel name in the results display.

"Eye Finder only operates when the analyzer clocking is master clock (slave and/or demux clocking are not supported)."

See the "Clock Setup" section of the "Sampling" tab in the analyzer setup window.

"Eye Finder only operates when the analyzer is setup for state analysis."

See the "Pod Assignment" dialog accessed from the "Format" tab in the analyzer setup window.

"From Eye Finder: After hardware calibration, the sampling positions for the following channels may have shifted out of the selected stable region by the amount shown: CHANNEL: AMOUNT ps ... (NNN more)"

Each time a measurement is started, the hardware is re-calibrated. The new calibration values are checked against those used when the *eye finder* measurements were taken. This message indicates that the sampling positions for the given CHANNELs may have drifted out of the stable region. The measurement is taken anyway, but you may want to treat the results with caution and run *eye finder* again (or manually adjust the sample position away from the indicated unstable region).

"Hardware calibration failed"

Something isn't as expected about the hardware and/or the cables and connections between boards. To get detailed messages, start an ordinary run or run PV.

"Measurement Canceled"

The measurement was stopped. No change was made to the results displayed. A run is stopped by user request or when the Sampling Positions dialog is closed or iconified.

"No labels defined with channels for the analyzer."

Define one or more labels with channels in the Format tab of the analyzer's Setup window.

No labels are defined for the analyzer. *Eye finder* cannot be run until one or more labels are defined with one or more channels assigned.

"No labels or channels selected for running Eye Finder."

Select one or more labels in the "Eye Finder Setup" tab.

All labels defined for the analyzer are listed in the Eye Finder Setup page. None are currently selected (selected labels are highlighted). Select one or more labels for measurement by *eye finder*.

"Repetitive runs stopped"

The measurement was stopped. Data from the last measurement which completed fully are retained. Repetitive runs are stopped by user

request or when the Sampling Positions dialog is closed or iconified.

"Timeout: < N K clocks in 5 sec"

Eye finder requires stimulus at a minimum rate to perform its measurements. Too few state clocks were seen in the time allotted. Check clock inputs, clock definition, threshold voltage settings, and the operation of the device under test.

Eye Finder Info Messages. These messages appear in the Eye Finder Results tab after an *eye finder* measurement is run.

"Example measurement for demo. Results and settings will not be used for analysis."

This channel was measured when "Use demo data (no probes required)" was selected in the Settings tab. The data shown are typical of *eye finder* operation, but the sample position setting shown is NOT used. (The manual setting is still in use.)

"No activity present. Confirm connection, stimulus, and threshold."

This channel appears to be completely quiet.

- Check the probe connection between the analyzer and the device under test.

- Check the threshold voltage setting in the Format tab.

- Check that the device under test is turned on and is running the appropriate diagnostic or other stimulus program.

If all these things are set up correctly, activity will be shown in the Format tab.

"No stable regions. Is this the correct clock for this channel?"

Two common possibilities exist:

1. The signal on this channel is asynchronous to the clock defined for the logic analyzer. If this is the case, there is no stable relationship between the times when the signal switches and when the clock arrives.

   If you expect the signal to be sampled synchronously you must redefine the clock for this signal.

2. The stable region(s) are too small for *eye finder* to detect.

   In this case you must resort to adjusting the sample position manually and checking its validity by running an ordinary analyzer measurement to see if the data values you expect are sampled. You can adjust the sample position manually by selecting the arrow buttons or by dragging the blue sampling position indicator in the display.

"Only a few transitions detected. Change stimulus or increase measurement duration (Advanced Settings)."

The signal on this channel was observed to toggle fewer than 500 times. The characterization may be accepted as it stands or you may wish to change the stimulus program or diagnostic in the device under test to increase the toggle rate.

Another option is to select "Long" in the Eye Finder Advanced Settings dialog (accessed from the "Advanced..." button on the Eye Finder Setup tab or the "Advanced Settings..." menu pick under the EyeFinder pull down menu). Using the "Long" setting won't necessarily make the message go away, but it will ensure that *eye finder* has the opportunity to observe a more significant number of transitions on the channel.

"Run Eye Finder to characterize this channel."

Select this channel (or a label that contains it) and run *eye finder* to characterize the channel. Only channels selected for a measurement are updated when the measurement finishes. Information about other channels is not changed.

"See individual channels for message(s)"

There is a message for one or more the the channels assigned to this bus label. Expand the label (using either the popup menu on the label in the display or the "Results" pulldown menu), then scroll down to the channel with a message icon and display its message (either by selecting the yellow message icon or by using the popup menu).

"The stable region extends beyond the limits of the display."

This channel is active, but the signal does not switch within 5 nsec before or after the clock. For example, this could occur if the propagation delay in the device under test from clock to data is greater

than 5 nsec and the clock period is greater than 10 nsec (slower than
100 MHz).

Eye Finder Load/Save Messages. These messages can appear when
saving or loading *eye finder* data.

"... (at line XX in the file)"

Indicates where the error occurred in the file being read. Since *eye
finder* data files are ASCII text, you can use a text editor to examine
the file at the indicated line to determine how to repair the problem.

"Bad assignment for XXX"

The numerical value for the item XXX could not be read.

**"Cannot load Eye Finder data for a different module type. Data in file is from a
OTHERMODEL. This module is a THISMODEL."**

**"Cannot load Eye Finder data for a module installed in different slots. Data in file is
from a module installed in slots B A C. This module is installed in slots E D F".**

**"Cannot load Eye Finder data from a different instrument. Data in file is from
OTHERINSTRUMENT. This instrument is THISINSTRUMENT."**

**"Channel name in the data file ("OTHERNAME") does not match the expected
channel name ("THISNAME")."**

**"Channel number in the data file (NN) does not match the expected channel number
(MM).",**

"Did not find: XXX (Search began at line LLL)"

The *eye finder* data file has a nested block structure. The next block
(or item) to be read was XXX, but it was not found before the end of
the file. The line after the last item successfully read was LLL.

"Error in writing"

Disk is probably full.

"Failed to open file for reading/writing: NAME"

The selected file could not be opened. Check access and file permissions.

"File NAME already exists. Overwrite?"

The selected file exists. Answering "Yes" will cause the existing contents of the file to be replaced with current *eye finder* information.

"Invalid true/false flag"

The boolean value for the item could not be read. Boolean values start with either a 't' or 'f' ('T' or 'F'are also accepted).

"Master slot in the data file ('j') does not match the expected master slot ('E')."

"The number of channels in the data file (NNN) does not match the number of channels in the analyzer (XXX)."

Eye finder characterizes the unique combination of the instrument, module, cabling, probes, and Device Under Test. Results are not transferable from one situation to another.

"Unsupported revision level (AA.BB)"

Something else is probably wrong because there aren't any unsupported versions.

Eye Finder Setup Tab
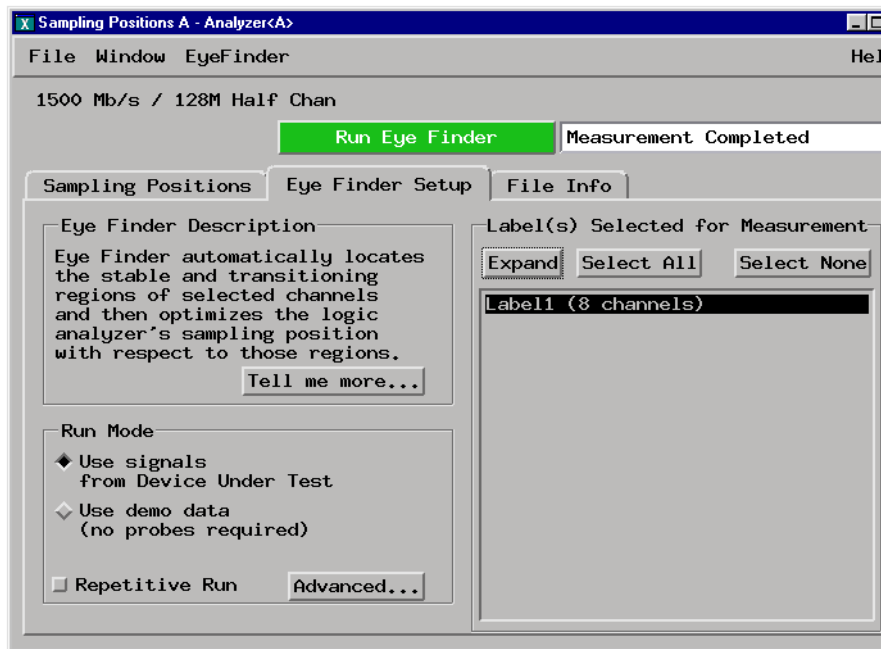
File menu   Lets you save/load *eye finder* data.

EyeFinder menu Lets you run *eye finder*, choose the run mode, and access the "Eye Finder Advanced Settings Dialog" on page 170.

Run Mode   Lets you look at *eye finder* with demo data or in normal operating mode by sampling signals from the device under test.

Repetitive Run Runs the *eye finder* repetitively, so you can see how stable and transitioning signals vary over time.

Advanced   See "Eye Finder Advanced Settings Dialog" on page 170.

Label Selection Lets you choose the labels (channels) to run *eye finder* on. You can expand or overlay the signals in a label, select all or select none of the signals, select individual signals, or select multiple signals.

                    If a channel appears in multiple labels, selecting that

channel will select it in each of those labels.

Eye Finder Advanced Settings Dialog.
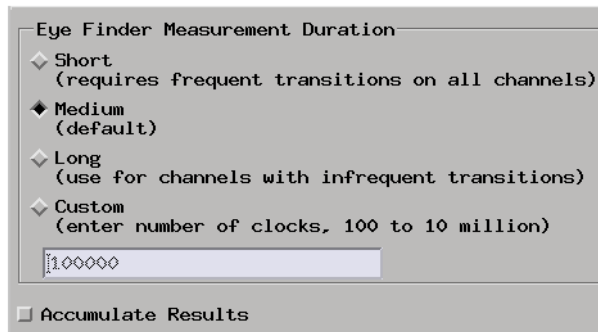
Short   *Eye finder* looks at 100,000 clock cycles on each channel
to determine the suggested sampling positions. This setting
requires frequent transitions on all channels.

Medium   *Eye finder* looks at 500,000 clock cycles on each channel
to determine the suggested sampling positions. Use this for
channels that transition at a normal rate.

Long   *Eye finder* looks at 2.5 million clock cycles on each
channel to determine the suggested sampling positions.
Use this setting if some channels have sporadic transitions.

Custom   Lets you specify the number of clock cycles to sample on
each channel at each sampling position (each point in
time).

**Accumulate
Results**   When selected, *eye finder* data is accumulated from run-
to-run. When unselected, *eye finder* data is erased before
each run.

Some things to consider when selecting among the *eye finder*
advanced settings are:

• Upper address bits that don't transition as frequently as lower address bits.

- Data buses that are driven by different circuitry at different times.

When different channels require different settings, you can run *eye finder* on channel subsets to avoid using the Long setting on a large number of channels.

File Info Tab

When the Eye Finder option is selected, the File Info subtab lets you save and load eye finder data.
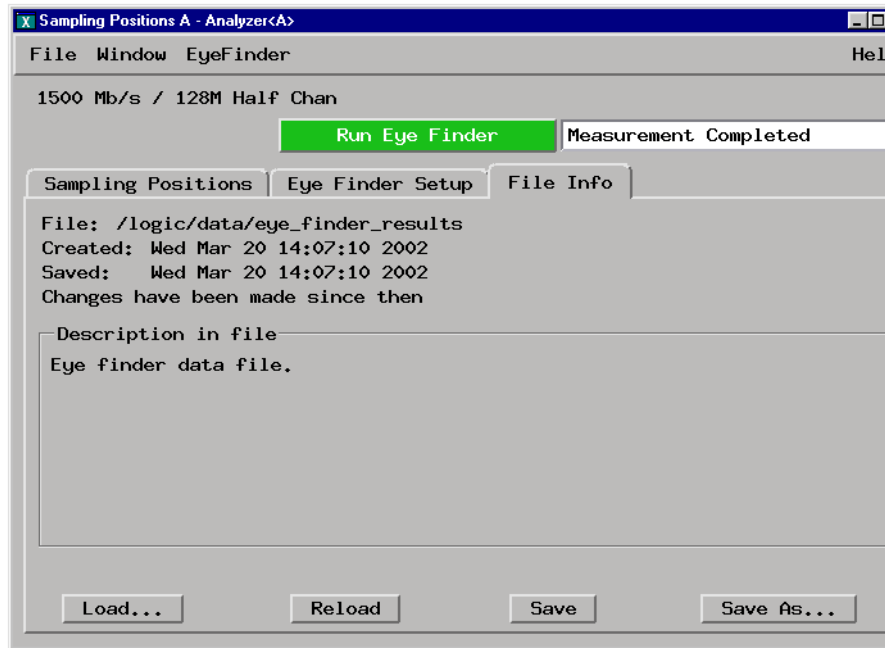
File: Name of the *eye finder* data file.

Created: Date and time the *eye finder* data file was created.

Saved: Date and time the *eye finder* data file was last saved.

                 You are notified if the eye finder data has changed since the last time it was saved.

Load...   Loads eye finder data that was previously saved to a file.

Reload	Reloads eye finder data from the named file, deleting unsaved changes.
Save	Saves current eye finder data to the named file.
Save As...	Saves current eye finder data to a new file.

See Also   "To automatically adjust sampling positions" on page 49

Eye Scan Mode
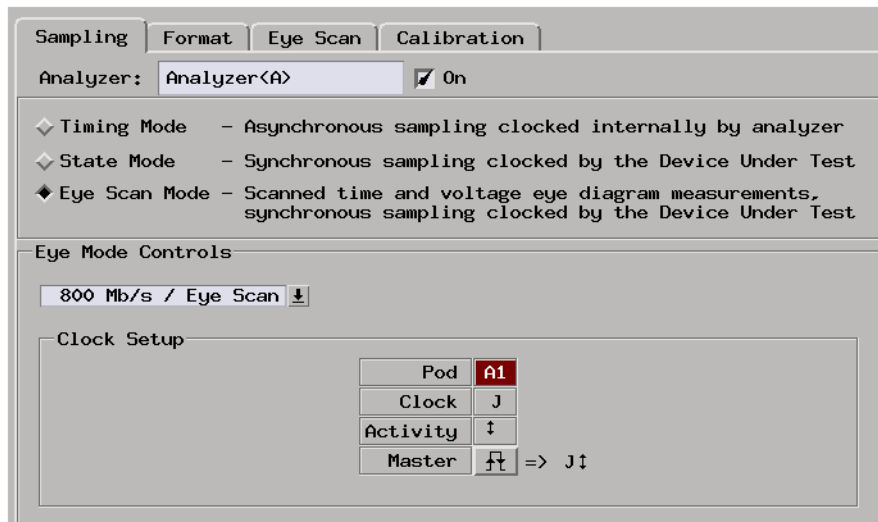
When you select Eye Scan Mode, the Eye Scan Mode Controls area appears.

Eye Scan Speed
Configuration Lets you configure the analyzer for faster signals, but with only half of the channels.

Clock Setup Lets you specify the clock edge from the device under test that will be used as the reference clock.

Generally, the eye scan mode reference clock is taken from the signals that clock valid data in the device under test.

See Also   "Selecting the Eye Scan Mode" on page 55

"Understanding Eye Scan Measurements" on page 259

The Format Tab
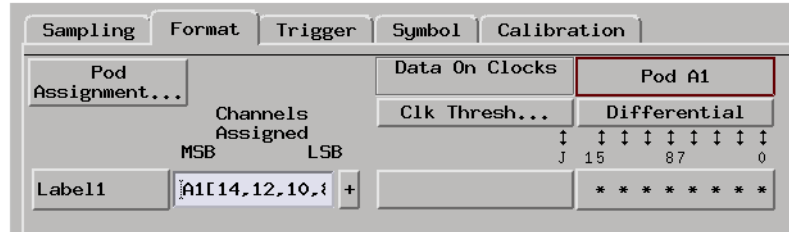
The Format tab is used to assign bus and signal names (from the device under test), to logic analyzer channels. These names are called *labels*. Labels are also used when setting up triggers and displaying captured data.

The Format tab also lets you assign pods to the logic analyzer and specify the logic analyzer pod and clock threshold voltages.

The Data On Clocks display column shows the clock inputs available as data channels in the present configuration, including the clocks on expander *cards* which cannot be used in the clock setup.

The Format tab has activity indicators that show whether the signal on a channel is above the threshold voltage (high), below the threshold voltage (low), or transitioning.

NOTE: Activity indicated on channels may appear inaccurate or missing while a measurement is still actively running in the state mode's 800, 1250, and 1500 Mb/s configurations.

• "Pod Assignment Dialog" on page 175

See Also   "Formatting Labels for Logic Analyzer Probes" on page 57

"To assign pods to the analyzer" on page 57

"To set pod threshold voltages" on page 58

"To set clock threshold voltages" on page 59

Pod Assignment Dialog
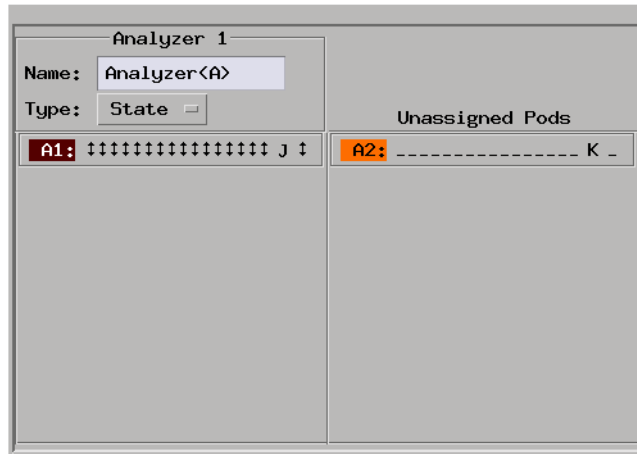
Name: Lets you name the analyzer.

Type: Lets you select the timing (asynchronous) sampling mode, the state (synchronous) sampling mode, or turn the analyzer off.

Pods Can be dragged-and-dropped under the analyzer to assign those channels to the analyzer or can be left unassigned.

The Trigger Tab
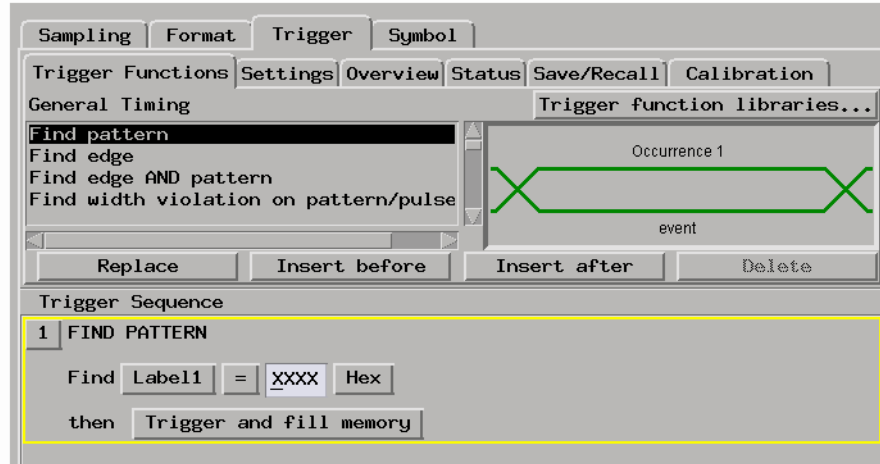
The Trigger tab is used to tell the analyzer when to capture data. The key event is the *trigger*.

In the Agilent Technologies 16760A logic analyzer, you can insert multiple trigger actions. When you insert multiple trigger actions, the trigger marker in the display windows is placed on the first sample whose evaluation caused a branch through an associated trigger action.

The Trigger tab has two main areas: On top, tabs of functions and controls to build your trigger; and beneath the tabs, the current trigger sequence. Some controls are also located in the logic analyzer window's menu bar.

- "Trigger Functions Subtab" on page 177

- "Settings Subtab" on page 188

- "Overview Subtab" on page 189

- "Default Storing Subtab" on page 190

- "Save/Recall Subtab" on page 191

See Also   "Understanding Logic Analyzer Triggering" on page 240

"Setting Up Triggers and Running Measurements" on page 69

"Editing the Trigger Sequence (Timing or 200, 400 Mb/s State Only)" on page 78

Trigger Functions Subtab
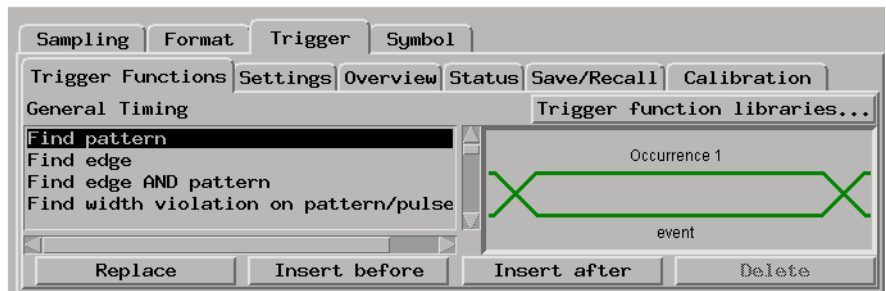
Trigger functions provide a simple way to set up the analyzer to *trigger* on common events and conditions.

Libraries of functions are available for each of the timing and state sampling mode configurations.

In addition, in the timing sampling mode and in the state sampling mode's *200 Mb/s / 32M State* configuration, you can create and load your own trigger function libraries, as well as unload the default library.

- "General Timing Trigger Functions" on page 178

- "General State Trigger Functions" on page 180

- "Advanced Trigger Functions" on page 183

- "Turbo State Trigger Functions" on page 185

- "16760 State Trigger Functions" on page 185

- "16760 Half Channel State Trigger Functions" on page 187

General Timing Trigger Functions

When the timing sampling mode is selected, the following *General Timing* trigger functions are found in the *Trigger Functions* tab.

You can edit most of these trigger functions to specify particular pattern and edge events or boolean combinations of events.

You can use the following types of events in these trigger functions: patterns, ranges, flags, counters, timers, and Wait for arm in (see "To cross-trigger with another instrument" on page 112).

You can use the following actions in these trigger functions: counters, timers, store control (in the *400 MHz / 32M Sample Transitional or Store Qualified* configuration when *Transitions* are not being stored), goto (for arbitrary sequencing), and trigger.

You can expand these trigger functions to see how they are constructed with the underlying advanced trigger functions.

You can break down these trigger functions to directly edit the underlying advanced trigger functions. Sometimes you need to break down a trigger function in order to add other event types such as counters, timers, flags, and arming from other instruments.

- *Find pattern*

  Becomes true when a pattern occurs one time.

- *Find edge*

  Becomes true when the specified edge occurs in one sample.

- *Find edge AND pattern*

  Becomes true when the specified pattern and the specified edge occurs in one sample.

- *Find width violation on a pattern/pulse*

  Becomes true when the width of a pattern violates minimum and

maximum width specifications.

- Find Nth occurrence of an edge

  Becomes true when the specified edge occurs in the specified number of samples.

- Find pattern present/absent for > duration

  Becomes true when the specified pattern is present or absent for greater than the amount of time specified.

- Find pattern present/absent for < duration

  Becomes true when the specified pattern is present or absent for less than the amount of time specified.

- Run until user stop

  Sets up to never trigger. You must select the stop button to view the captured data.

- Find 2 edges too close together

  Becomes true when the second specified edge occurs within a specified time after the first specified edge.

- Find 2 edges too far apart

  Becomes true when the second specified edge does not occur within a specified time after the first specified edge.

- Find pattern occurring too soon after edge

  Becomes true when a specified pattern occurs within a specified time after the first specified edge.

- Find pattern occurring too late after edge

  Becomes true when a specified pattern does not occur within a specified time after the first specified edge.

- Find glitch

  Becomes true when the specified glitch occurs in one sample.

- Wait t seconds

  Becomes true when the specified amount of time has expired.

- Wait for arm in

  When the logic analyzer is armed by another instrument (as specified in the Intermodule window), this trigger function becomes true when the arm signal is received.

- Wait for flag

  Becomes true when the specified flag has the specified value. This trigger function tests for a flag event.

- OR Trigger

  When the logic analyzer is armed by another instrument (as specified in the Intermodule window), this trigger function becomes true when a pattern occurs a specified number of times OR when the arm signal is received.

- "Advanced Trigger Functions" on page 183

See Also   "To specify a label pattern event (Timing only)" on page 71

"To specify a label edge event" on page 72

"To break down a trigger function (timing or 200 Mb/s state only)" on page 72

"To cross-trigger with another instrument" on page 112

General State Trigger Functions

When the state sampling mode's *200 Mb/s / 32M State* configuration is selected, the following *General State* trigger functions are found in the *Trigger Functions* tab.

You can edit most of these trigger functions to specify particular pattern events or boolean combinations of events.

You can use the following types of events in these trigger functions: patterns, ranges, flags, counters, timers, and Wait for arm in (see "To cross-trigger with another instrument" on page 112).

You can use the following actions in these trigger functions: flags, counters, timers, store control, goto (for arbitrary sequencing), and trigger.

You can expand these trigger functions to see how they are constructed with the underlying advanced trigger functions.

You can break down these trigger functions to directly edit the underlying advanced trigger functions. Sometimes you need to break down a trigger function in order to add other event types such as counters, timers, flags, and arming from other instruments.

• Find pattern n times

  Becomes true when the specified pattern occurs in the specified number of samples (eventually).

• Store range until pattern occurs

  Becomes true when the specified pattern occurs in the specified number of samples (eventually) and only stores samples in the specified range until then.

• Store pattern2 until pattern1 occurs

  Becomes true when the first specified pattern occurs in the specified number of samples (eventually) and only stores samples with the second specified pattern until then.

• While storing pattern2, find pattern1

  This trigger function has been replaced by the "Store range until pattern occurs" and "Store pattern2 until pattern1 occurs" trigger functions.

• Store nothing until pattern occurs

  Becomes true when the specified pattern occurs one time and doesn't store any samples until then.

• Run until user stop

  Sets up to never trigger. You must select the stop button to view the captured data.

• Find pattern2 occurring immediately after pattern1

  Becomes true when the second specified pattern occurs in the sample immediately after a sample in which the first specified pattern occurs.

• Find pattern1 eventually followed by pattern2

Becomes true when the second specified pattern occurs in a sample (eventually) after a sample in which the first specified pattern occurs.

- Find pattern2 occurring too soon after pattern1

  Becomes true when the second specified pattern occurs within a specified time after the first specified pattern.

- Find pattern2 occurring too late after pattern1

  Becomes true when the second specified pattern does not occur within a specified time after the first specified pattern.

- Find too few states between pattern1 and pattern2

  Becomes true when the second specified pattern occurs within a specified number of samples (states) after the first specified pattern.

- Find too many states between pattern1 and pattern2

  Becomes true when the second specified pattern does not occur within a specified number of samples (states) after the first specified pattern.

- Find n-bit serial pattern

  Becomes true when a specified serial pattern of N bits is found on the analyzed line.

- Find pattern n consecutive times

  Becomes true when the specified pattern occurs in the specified number of samples consecutively.

- Find pattern2 n times after pattern1, before pattern3 occurs

  Becomes true when the second specified pattern occurs in a specified number of samples after the the first specified pattern but without the third specified pattern occurring anywhere in between.

- Store n samples

  Becomes true when the specified number of samples are stored.

- Wait n external clock states

  Becomes true when the specified number of external clocks have occurred.

- Wait for arm in

When the logic analyzer is armed by another instrument (as specified in the Intermodule window), this trigger function becomes true when the arm signal is received.

- Wait for flag

  Becomes true when the specified flag has the specified value. This trigger function tests for a flag event.

- Set/clear/pulse flag

  Becomes true on any sample and sets, clears, pulse sets, or pulse clears the specified flag. This trigger function inserts a flag action.

- OR Trigger

  When the logic analyzer is armed by another instrument (as specified in the Intermodule window), this trigger function becomes true when a pattern occurs a specified number of times OR when the arm signal is received.

- "Advanced Trigger Functions" on page 183

See Also

"To specify a label pattern event (Timing only)" on page 71

"To break down a trigger function (timing or 200 Mb/s state only)" on page 72

"To cross-trigger with another instrument" on page 112

Advanced Trigger Functions

The advanced trigger functions let you create a custom *trigger sequence* level using events, comparison functions, and up to 4 branches.

The advanced trigger functions are available in the timing sampling mode and in the state sampling mode's *200 Mb/s / 32M State* configuration. In the timing sampling mode, you can specify edges on labels and event durations.

The types of events include labels, timers, flags, and counters.

- Advanced - If/then

  This trigger function has only one branch. If the events in the "If" event list

are true, it executes the actions after "then".

- Advanced - 2-way branch

  This trigger function has two branches, of the form "If - then; else if - then".

  For each sample, the events in the first "If" branch are checked. If all events are true, the "then" portion is executed. If they are not true, the events in the "else if" branch are checked. If the "else if" events are true, its "then" portion is executed.

  If neither branch is true, the logic analyzer remains in this sequence level and repeats the comparison with the next sample.

- Advanced - 3-way branch

  This function has three branches, of the form

```
If (events1)
  then (actions1)
Else if  (events2)
  then (actions2)
Else if (events3)
  then (action3)
```

  The logic analyzer evaluates each sample against the clauses in the order they are specified. The logic analyzer executes the set of actions in the "then" clause associated with the first listed "if" or "else if" clause that becomes true.

  This trigger function is not available in the timing sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration.

- Advanced - 4-way branch

  Like the 3-way branch, but with 3 "Else if" clauses.

  This trigger function is not available in the timing sampling mode's *400 MHz / 32M Sample Transitional or Store Qualified* configuration.

- Advanced - pattern1 AND pattern2

  Searches for two different patterns occurring in the same sample. If you set it to look for more than 1 occurrence, you can specify whether occurrences are consecutive or not. You can also add other events, including labels, to be searched for.

- Advanced - pattern1 OR pattern2

  Finds either pattern1 or pattern2 or both in a sample. If you set it to look for more than 1 occurrence, you can specify whether the occurrences are consecutive or not. You can also add other events, including labels, to be searched for.

See Also   "Editing Advanced Trigger Functions (Timing or 200 Mb/s State Only)" on page 83 for more information on using the advanced trigger functions.

Turbo State Trigger Functions

When the state sampling mode's *400 Mb/s / 32M State* configuration is selected, the following *Turbo State* trigger functions are found in the *Trigger Functions* tab.

You can edit these trigger functions to specify particular pattern and edge events or boolean combinations of events.

You can use the following types of events in these trigger functions: patterns, ranges, flags, and Wait for arm in (see "To cross-trigger with another instrument" on page 112).

These trigger functions provide goto and trigger actions. You cannot modify these actions or insert any other actions.

- Find pattern n times

  Becomes true when the specified pattern occurs the specified number of times.

- Find pattern1, or reset on pattern2

  Becomes true when the specified pattern1 occurs. If pattern2 occurs before pattern1, then evaluation resets (starts over).

16760 State Trigger Functions

When the state sampling mode's *800 Mb/s / 64M State* configuration is selected, the following *16760 State* trigger functions are found in the *Trigger Functions* tab.

You can edit these trigger functions to specify particular pattern events or boolean combinations of events.

You can use the following types of events in these trigger functions: patterns, ranges, flags, and Wait for arm in (see "To cross-trigger with another instrument" on page 112).

These trigger functions provide trigger actions. You cannot modify these actions or insert any other actions.

Only one of these trigger functions can be used at a time; in other words, they cannot be sequenced to provide more sophisticated triggering.

- Find pattern

  Trigger when the specified pattern occurs.

- Find 2 patterns in immediate sequence

  Trigger when the second specified pattern occurs in the next sample after a sample in which the first specified pattern occurs.

- Find 3 patterns in immediate sequence

  Trigger when all three specified patterns occur in the specified order. The occurrence of each pattern must be in contiguous samples.

- Find 4 patterns in immediate sequence

  Trigger when all four specified patterns occur in the specified order. The occurrence of each pattern must be in contiguous samples.

- Find 2 patterns in eventual sequence

  Trigger when the second specified pattern occurs, eventually, after the first specified pattern has already occurred.

- Find 3 patterns in eventual sequence

  Trigger when all three specified patterns occur, eventually, in the specified order.

- Find 4 patterns in eventual sequence

  Trigger when all four specified patterns occur, eventually, in the specified order.

- Run until user stop

  Sets up to never trigger. You must select the stop button to view the

captured data. For maximum data capture the trigger position should be set to *end*.

A warning message concerning no trigger action will be displayed in the *Run Status* window. This warning message can be ignored since the intent of the trigger function is not to trigger. The run status window can be prevented from coming up with this warning by setting the *Popup On Severity* level to errors only.

16760 Half Channel State Trigger Functions

When the state sampling mode's *1250 Mb/s / 128M Half Chan* or *1500 Mb/s / 128M Half Chan* configuration is selected, the following *16760 Half Channel State* trigger functions are found in the *Trigger Functions* tab.

You can edit these trigger functions to specify particular pattern events or boolean combinations of events.

You can use the following types of events in these trigger functions: patterns, ranges, flags, and Wait for arm in (see "To cross-trigger with another instrument" on page 112).

These trigger functions provide trigger actions. You cannot modify these actions or insert any other actions.

Only one of these trigger functions can be used at a time; in other words, they cannot be sequenced to provide more sophisticated triggering.

• Find pattern

  Trigger when the specified pattern occurs.

• Find 2 patterns in immediate sequence

  Trigger when the second specified pattern occurs in the next sample after a sample in which the first specified pattern occurs.

• Find 2 patterns in eventual sequence

  Trigger when the second specified pattern occurs, eventually, after the first specified pattern has already occurred.

• Run until user stop

Sets up to never trigger. You must select the stop button to view the captured data. For maximum data capture the trigger position should be set to *end*.

A warning message concerning no trigger action will be displayed in the *Run Status* window. This warning message can be ignored since the intent of the trigger function is not to trigger. The run status window can be prevented from coming up with this warning by setting the *Popup On Severity* level to errors only.

Settings Subtab
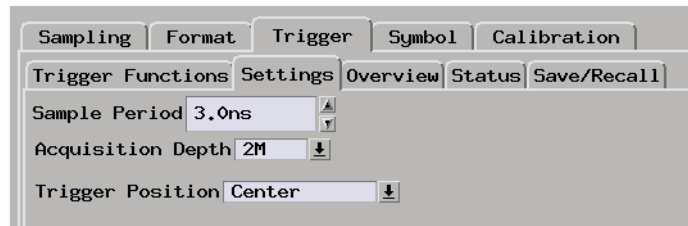
Sample Period (Timing mode only). Lets you specify how often the logic analyzer samples signals from the device under test. In the timing mode's *800 MHz / 64M Sample Conventional* configuration, the sample period is fixed at 1.25 ns.

Acquisition Depth Lets you use a smaller portion of the acquisition memory and speed up processing of the captured data.

Trigger Position Lets you specify where the sample that triggered the analyzer should appear among all the other samples that are stored in acquisition memory.

Count Lets you turn off counting, count time for each stored sample, or, in the state mode's *200 Mb/s / 32M State* configuration, count the occurrence of some state with each stored sample.

Intermodule Control Opens the Intermodule window which lets you configure multiple instrument measurements, adjust the order of

trigger arming, and compensate for timing skew between modules.

See Also   "To specify the sample period" on page 45

"To set acquisition memory depth" on page 53

"To specify the trigger position" on page 53

"To count time, states, or turn counting off" on page 75

"To cross-trigger with another instrument" on page 112

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

Overview Subtab
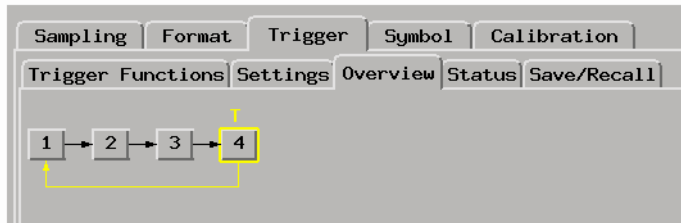
This tab gives a picture of the trigger sequence.

See Also   "To view a picture of the trigger sequence" on page 83

Default Storing Subtab

Store by default Lets you specify that all events (*Anything*), no events (*Nothing*), *Custom* (user defined) events, or *Transitions* be stored by default.

At start of acquisition Lets you choose whether default storing is initially *On* or *Off*.

Event specification list When you choose *Custom* (user defined) events, this area lets you define which events should be stored by default.

Group events When you choose *Custom* events, you can group events in the list to specify the evaluation order.

See Also "To specify default storing" on page 76

"To specify whether default storing is initially on or off" on page 78

"To group events" on page 89

"To specify a label pattern event (Timing only)" on page 71

Status Subtab
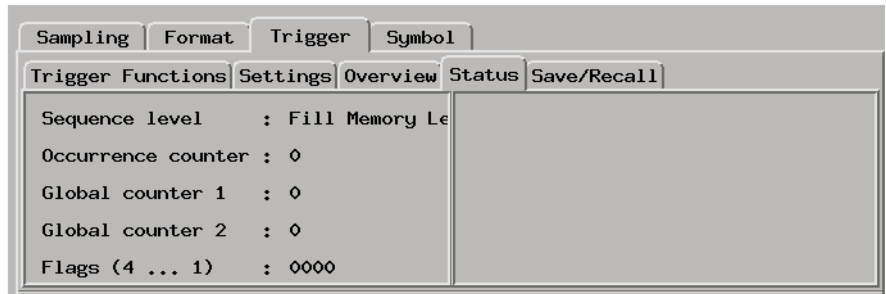
The Status subtab shows you the sequence level that is evaluating captured data, occurrence and global counter values, and flag values.

See Also   "To view the trigger status" on page 93

Save/Recall Subtab
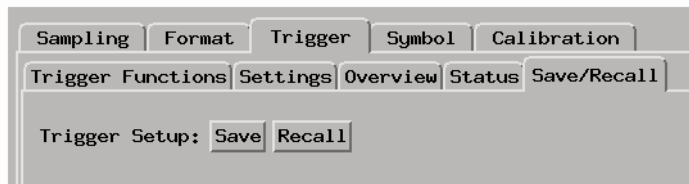
The Save/Recall subtab lets you save trigger setups within a session.

The Agilent Technologies 16760A logic analyzer provides memory locations to store up to 15 trigger sequences for both state and timing sampling modes. Five of the 15 memory positions are reserved for the 5 most recent runs.

When you exit your Agilent Technologies 16700 *session*, the trigger save/recall list is cleared. However, the trigger save/recall list can be saved as part of a configuration file.

You can also save trigger sequences outside of configuration files by creating trigger function libraries.

See Also   "Saving/Recalling Trigger Setups" on page 90

"Saving and Loading Logic Analyzer Configurations" on page 116

"To create a trigger function library (timing or 200 Mb/s state only)" on page 73

The Symbols Tab
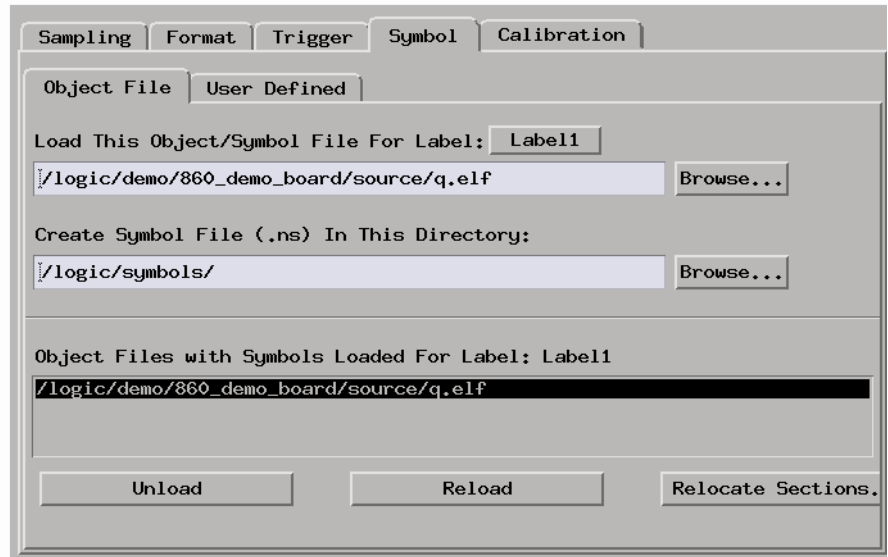
The Symbols tab lets you load symbol files or define your own symbols. Symbols are names for particular data values on a label.

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.

- "Symbols Selector Dialog" on page 195

- "Symbol File Formats" on page 197

- "General-Purpose ASCII (GPA) Symbol File Format" on page 198

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" on page 101

Symbols Selector Dialog

```
┌─────────────────────────────────────────────────────────────┐
│  ─                    Symbol Selector — ADDR                  │
├─────────────────────────────────────────────────────────────┤
│                                                               │
│                                                               │
│   Search Pattern: *                        │  Recall  │      │
│  ┌─Find Symbols of Type────────────────────────────────┐     │
│  │ ■ Function       ■ Variable        ■ Label          │     │
│  │ ■ Source Files   ■ User Defined    ■ Case Sensitive │     │
│  └─────────────────────────────────────────────────────┘     │
│                                                               │
│  Matching Symbols                        202 Symbols Found    │
│  ┌──────────────────────────────────────────────────────┐    │
│  │ INPUT                    Function       FFF06C9C-FFF │▲│   │
│  │ Init_IO                  Function       FFF06BA8-FFF │█│   │
│  │ ME_add_to_history        Variable            41CD    │ │   │
│  │ ME_clear_hist_buff       Variable            41BD    │ │   │
│  │ ME_do_sort               Variable            41C1    │ │   │
│  │ ME_first_marker          Variable            41B8    │ │   │
│  │ ME_get_targets           Variable            41C3    │ │   │
│  │ ME_proc_specific         Variable            41BF    │ │   │
│  │ ME_read_conditions       Variable            41C5    │ │   │
│  │ ME_save_points           Variable            41CB    │ │   │
│  │ ME_set_outputs           Variable            41C7    │ │   │
│  │ ME_update_display        Variable            41BB    │ │   │
│  │ ME_update_system         Variable            41B9    │ │   │
│  │ ME_write_hdwr            Variable            41C9    │ │   │
│  │ MX_add_to_history        Variable            41CE    │▽│   │
│  └──────────────────────────────────────────────────────┘    │
│  ◁                                                     ▷      │
│                                                               │
│  Offset By     Align to                                       │
│  0x 00000000    │ 1 Byte  ▔│   │ Beginning ▔│                 │
│                                                               │
│  ┌──────────────┐   ┌──────────────┐   ┌──────────────┐       │
│  │      OK      │   │    Cancel    │   │     Help     │       │
│  └──────────────┘   └──────────────┘   └──────────────┘       │
└─────────────────────────────────────────────────────────────┘
```

Search Pattern: Lets you enter partial symbol names and the asterisk wildcard character (*) to limit the symbols to choose from (see "Search Pattern" on page 196). Use the Recall button to select from previous search patterns.

**Find Symbols of
Type** Lets you limit the types of symbols to choose from.

**Matching
Symbols** Lists the symbols that match the search pattern. You choose a symbol from this list.

Offset By Lets you add an offset value to the starting point of a symbol. This can be useful when compensating for microprocessor prefetches (see "Offset By Option" on page 196).

Align to Lets you mask the lower order bits of a symbol's value. This can be useful for triggering on odd byte boundaries (see "Align to x Byte Option" on page 197).

Beginning/End/ Range When a symbol represents a range of addresses, you can choose the beginning address of the range, the end address of the range, or the whole range.

See Also "To enter symbolic label values" on page 105

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*".

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. 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 label value. 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 label value. You would set the value of the "Offset By" field to 10 hex. Now, when you specify *func2* as your label value, the logic analyzer will trigger on address *func2+10*.

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.

Symbol File Formats

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

- OMF96

- OMFx86

- IEEE-695

- ELF/DWARF

- ELF/stabs

- TI COFF

For ELF/DWARF1, 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 107) file.

See Also

"To load object file symbols" on page 102

"To create an ASCII symbol file" on page 106

"To create a readers.ini file" on page 107

General-Purpose ASCII (GPA) Symbol File Format

General-purpose ASCII (GPA) format files are loaded into a logic analyzer just like other object files.

If your compiler does not produce object files in a supported format, or if you want to define symbols that are not included in the object file, you can create an ASCII format symbol file.

Typically, ASCII format symbol files are created using text processing tools that convert the symbol table information from a compiler or linker map output file.

Different types of symbols are defined in different records in the GPA file. Record headers are enclosed in square brackets, for example, [VARIABLES]. For a summary of GPA file records and associated symbol definition syntax, refer to the "GPA Record Format Summary" on page 199 that follows.

Each entry in the symbol file must consist of a symbol name followed by an address or address range.

While symbol names can be longer, the logic analyzer only uses the first 16 characters.

The address or address range must be a hexadecimal number. It must appear on the same line 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
```

The following example defines two symbols that correspond to address ranges and one symbol that corresponds to a single address.

```
main      00001000..00001009
test      00001010..0000101F
var1      00001E22      #this is a variable
```

For more detailed descriptions of GPA file records and associated symbol definition syntax, refer to the following topics:

- "SECTIONS" on page 200

- "FUNCTIONS" on page 201

- "VARIABLES" on page 202

- "SOURCE LINES" on page 202

- "START ADDRESS" on page 203

- "Comments" on page 203

GPA Record Format Summary

Format

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

[FUNCTIONS]
func_name  start..end

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

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

```
[START ADDRESS]
address
```

```
#comment text
```

Lines without a preceding header are assumed to be symbol definitions in one of the [VARIABLES] formats.

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

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

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

[VARIABLES]
total     40002000  4
value     40008000  4

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

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

SECTIONS

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

NOTE: To enable section relocation, section definitions must appear before any other definitions in the file.

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

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
```

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
```

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 specify 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
```

SOURCE LINES

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
14        0000100A
22        0000101E
```

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

START ADDRESS

Format

```
[START ADDRESS]
address
```

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

Example

```
[START ADDRESS]
00001000
```

Comments

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

Format

```
#comment text
```

Example

```
#This is a comment
```

The Eye Scan Tab

The Eye Scan tab is used to set up and run eye scan measurements.

- "Labels Subtab" on page 204
- "Scan Settings Subtab" on page 205
- "Advanced Subtab" on page 206
- "Qualifier Subtab" on page 208
- "Comments Subtab" on page 209

See Also   "Selecting the Eye Scan Mode" on page 55

"Setting Up and Running Eye Scan Measurements" on page 119

"Understanding Eye Scan Measurements" on page 259

Labels Subtab
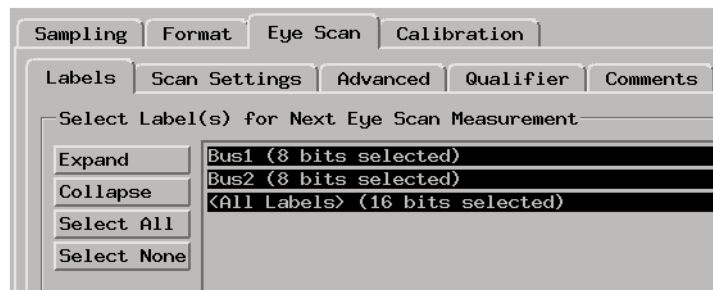
In the label/channel list, selected channels are highlighted. Double-clicking a label expands and collapses the channels in the label.

Expand   Shows all channels within labels.

Collapse   Shows labels only.

Select All   Selects all channels.

Select None   De-selects all channels.

Scan Settings Subtab
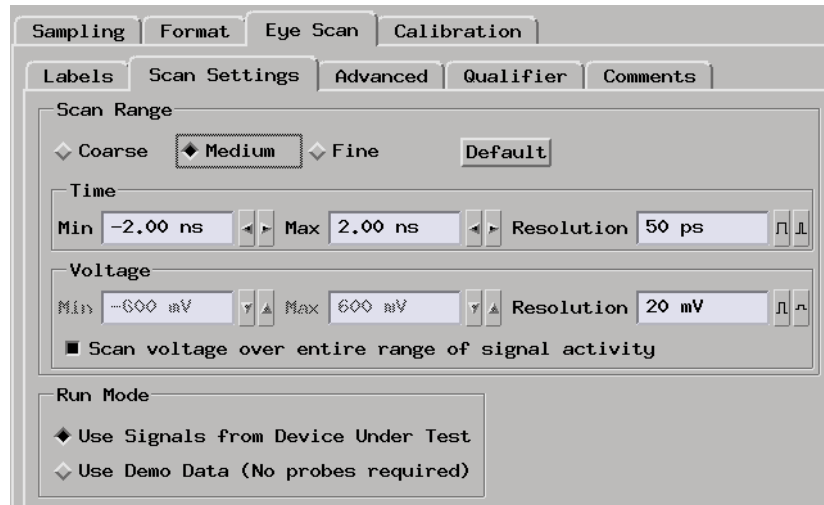
Scan Range   Lets you choose from *Coarse*, *Medium*, or *Fine* time and
              voltage settings.

              The *Default* button returns to the defaults for the selected
              scan range option.

Time   Lets you enter the minimum and maximum times (relative
              to clock signal edges) and the size of the time windows for
              the scan.

Voltage   Lets you enter the minimum and maximum voltages
              (relative to the threshold voltage) and the size of the time
              windows for the scan.

              If the *Scan voltage over entire range of signal activity*
              option is selected, the voltage range is determined
              automatically during the measurement.

Run Mode   Lets you look at eye scan results with demo data or in

normal operating mode by sampling signals from the device under test.

See Also   "To set the eye scan range and resolution" on page 120

Advanced Subtab

Eye scan measurements look at selected logic analyzer channels for signals passing through small windows of time and voltage.

Advanced options let you specify the number of clocks to process in each window and whether measurement data should be accumulated or replaced.
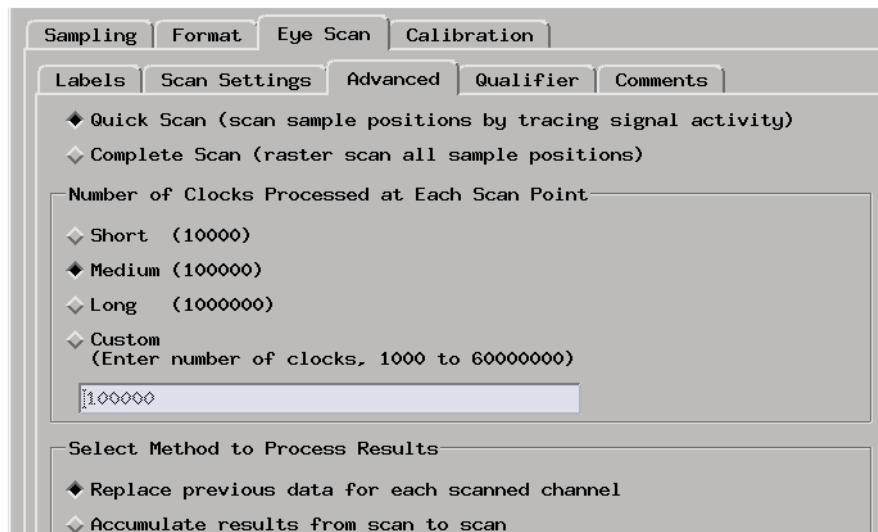
Quick Scan   Provides a faster eye scan than *Complete Scan*. Scanning is limited to regions that exhibit significant signal activity. It is possible that noise or other signal anomalies may not be displayed because areas that don't exhibit activity are not repeatedly scanned.

Complete Scan   This is a raster scan of all pixels in the Eye Scan display. This scanning method is slower, but it ensures that noise

and other anomalies will be displayed if present.

Short   Processes fewer clocks at each scan point. This setting requires frequent transitions on all channels.

Medium   Processes a moderate number of clocks at each scan point. Use this for channels transition at a normal rate.

Long   Processes more clocks at each scan point. Use this setting if some channels have sporadic transitions.

Custom   Lets you enter the number of clocks to process at each scan point.

**Select Method
to Process
Results**   You can choose to process measurement results by replacing previous data or by accumulating data.

**Replace
previous data
for each
scanned channel** Eye Scan data will be erased at the beginning of each run. This only applies to data for channels which are selected for Eye Scan (using a label or discretely). Data for channels that are not selected will not be erased at the beginning of the run.

**Accumulate
results from
scan to scan**   Scan data from each successive run is overlaid on existing data.

See Also   "To set advanced eye scan options" on page 121

Qualifier Subtab
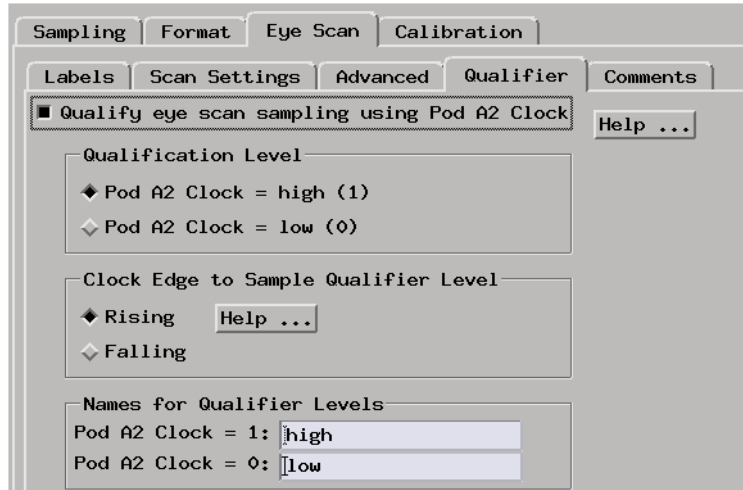
**Qualification
Level**   Select the button that describes the state in which you
                   want Eye Scan to collect data.

**Clock Edge to
Sample
Qualifier Level** Qualified eye scans may only be performed with double
                   edge clocks. Use *Clock Edge to Sample Qualifier Level*
                   setting to specify whether the qualifier signal is checked on
                   the rising edge or the falling edge of the system clock.

**Names for
Qualifier Levels** You can rename qualifier levels from "high" and "low" to
                   names that are meaningful for your system (for example,
                   read and write).

See Also   "To set up qualified eye scan measurements" on page 122

Comments Subtab
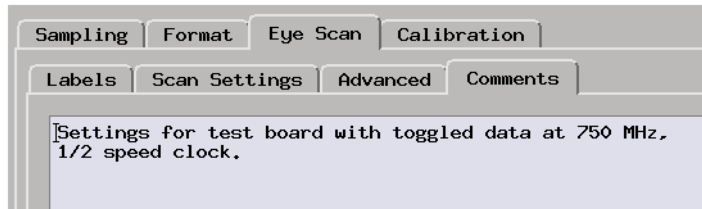
You can enter your comments on the eye scan settings. When the logic analyzer configuration is saved, comments are saved along with the eye scan settings.

See Also   "To comment on the eye scan settings" on page 132

The Calibration Tab
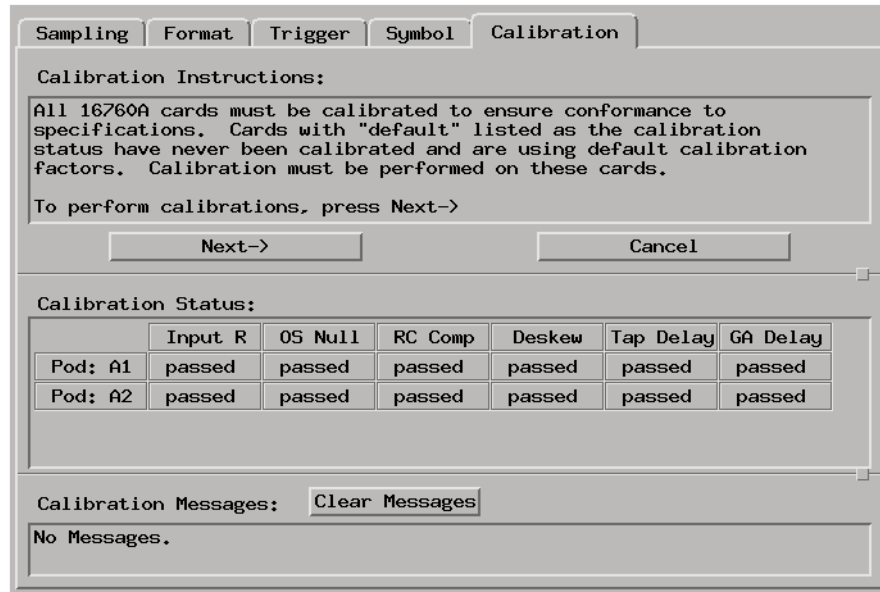
All 16760A logic analyzer cards must be calibrated to ensure conformance to specifications.

**Calibration
Instructions**   Follow these instructions to calibrate 16760A logic analyzer cards.

The *Cancel* button exits the calibration procedure.

**Calibration
Status**   Displays whether calibration has passed or failed.

**Calibration
Messages**   Displays any messages returned during the calibration.

The *Clear Messages* button removes all messages.

See Also   "What is a Specification?" on page 236

Error Messages

- "Analyzer armed from another module contains no "Arm in from IMB" event" on page 212

- "Branch expression is too complex" on page 212

- "Cannot specify range on label with clock bits that span pod pairs" on page 217

- "Counter value checked as an event, but no increment action specified" on page 218

- "Goto action specifies an undefined level" on page 218

- "Hardware Initialization Failed" on page 218

- "Maximum of 32 Channels Per Label" on page 219

- "Must assign another pod pair to specify actions for flags" on page 219

- "Must assign Pod 1 on the master card to specify actions for flags" on page 219

- "No more Edge/Glitch resources available for this pod pair" on page 219

- "No more Pattern resources available for this pod pair" on page 220

- "No Trigger action found in the trace specification" on page 221

- "Slow or Missing Clock" on page 221

- "Timer value checked as an event, but no start action specified" on page 222

- "Trigger function initialization failure" on page 222

- "Trigger inhibited during timing prestore" on page 223

- "Trigger Specification is too complex" on page 223

- "Waiting for Trigger" on page 225

Analyzer armed from another module contains no "Arm in from IMB" event

This warning is displayed when a 16715A and newer analyzer machine is in the group run arming tree, armed from another module (not directly from the group run), and no sequencer event list in the analyzer contains an "Arm in from IMB" event or no level in a "Wait for arm in" trigger function.

In this case the analyzer will not wait for the module above it to trigger before it triggers.

Branch expression is too complex

The "Branch expression is too complex" message means that the event list expression for the indicated branch contains more event terms to logically combine than the hardware is capable of combining on a single branch.

Other branches in the sequence may also be too complex. The trigger sequence compiler stops compiling at the first convenient place after it encounters a fatal error.

Because the trigger sequence compiler tries to optimize the event list expression to best fit the capabilities of the hardware, a precise description of the event list limits cannot be easily enumerated, but listed below are some general guidelines for all acquisition modes and some specific suggestions for particular modes.

General Guidelines

- Labels that span multiple pod pairs (split labels) greatly increases the compiled hardware expression complexity as compared with labels that are entirely contained within a single pod pair.

  Whenever possible try to arrange the probing such that labels to not span pod pairs. This is the single most effective way to reduce the complexity required to implement the event list expression.

NOTE:

For labels that do span pod pairs, the complexity can be reduced to the same as that of the non-split label case if all bits in the label on all but one pod pair can be set to Xs in the event list expression for the measurement.

For example, if label ADDR has its 16 most significant bits on pod A3 and 16 least significant bits on pod A2 (spanning pod pairs A4/A3 and A2/A1), the complexity of the compiled expression will be reduced if all 16 most significant bits or all 16 least significant bits are set to Xs in the pattern event.

- Inequality compares (<,<=,>,>=) of split labels increases the expression complexity compared to equality (=,!=) compares of split labels. There is no difference in complexity for non-split labels.

- Ranges are implemented as two inequality compares which doubles the required complexity for non-split labels but compounds the complexity to an even greater extent for ranges on split patterns.

- Equivalent event list expressions compile to a MUCH greater hardware complexity in 400/800/1250/1500 Mb/s state modes than in 200 Mb/s state mode. This is due to the way the hardware implements these faster state modes. The hardware parallelizes the data to allow the internal sequencer to still run at <= 200 Mb/s. However, this requires the trigger compiler to allocate additional sequence levels, branches, and pattern resources and combine them in complex expressions to de-parallelize the trigger expression. Using split labels in these faster modes further multiplies the complexity of these compiler generated expressions.

- The trigger compiler first expands all expression lists to sum-of-products form (for example, A(B+C) is expanded to AB+AC). The trigger compiler then does rudimentary boolean reduction on the expanded expression. However, the compiler does make some trade-offs between complete reduction and compile speed. Manually expanding and reducing a complex expression may help the trigger compiler to better fit the expression into the hardware resources.

Specific Guidelines - 200 Mb/s State and all Timing Modes

- Cannot OR more than 16 non-split pattern events if the pattern events are all on the same pod pair.

- Cannot OR more than 4 non-split pattern events if each pattern event is on a different pod pair. You can, however, OR 4 patterns together on each of 4 different pod pairs to make a total of 16 patterns ORed across 4 pod pairs.

- Cannot AND more than 16 non-split pattern events if the pattern events are all on the same pod pair.

- Can AND up to 160 non-split pattern events if the pattern events are evenly distributed across all 10 pod pairs on a 5 card set (16 pattern events per pod pair).

Specific Guidelines - 400 Mb/s State Modes

- Cannot AND or OR more than 8 non-split pattern events if the pattern events are all on the same pod pair.

- Cannot OR more than 4 non-split pattern events if each pattern event is on a different pod pair. You can, however, OR 2 patterns together on each of 4 different pod pairs to make a total of 8 patterns ORed across 4 pod pairs.

- Cannot AND or OR more than 4 non-split ranges if the pattern events are all on the same pod pair.

- Cannot AND or OR more than 2 split equality (=,!=) pattern events.

- Cannot specify more than 1 split inequality (<,<=,>,>=) pattern events.

- Cannot specify any range on a split label.

- In 400 Mb/s State Mode, the trigger sequence compiler must combine elements of the trigger events of the previous sequence level and the next sequence with the current sequence level, thereby increasing the total complexity of the current level. A sequence level that may compile fine when its the only level in the sequence, may be too complex to compile another level is inserted before or after it.

    One possible work-around to this problem is to insert a simple "If anything" sequence level in between two complex levels. The disadvantage to this approach, of course, is that the trigger sequence will miss one state in between the two complex sequence levels.

If the following sequence does not compile:

```
 1 If (complex event list)
   occurs 1 time
   then Goto Next
 2 If (complex event list)
   occurs 1 time
   then Trigger and fill memory
```

This one may:

```
1 If (complex event list)
  occurs 1 time
  then goto next
2 If anything
  occurs 1 time
  then Goto Next
3 If (complex event list)
  occurs 1 time
  then Trigger and fill memory
```

- In 400 Mb/s State Modes, the trigger sequence compiler must always add some additional complexity to the compiled expression for the first sequence level that is not needed in subsequent sequence levels. Additional complexity is also required in the first sequence level for the following 2 conditions:

  a. When using the "Find pattern1, or reset on pattern2" trigger function in 400 Mb/s State Modes, the event list of the first sequence level must be combined with reset branch of each subsequent sequence level by the trigger compiler in order to evaluate the parallelized samples.

  b. When using double edge clocking mode (J Clk rising and falling edges) in 400 Mb/s State Mode, an additional pattern resource is allocated and combined with the event list in the first sequence level by the trigger compiler to prevent triggering on an initial garbage state.

  Inserting an "If anything" state as the first state can simplify the complexity of the compiled event list in the first sequence level and subsequent "If/Else" sequence levels. The disadvantage is that the sequence will then miss the first state after the reset condition is met in an "If/Else" sequence level.

If the following sequence does not compile:

```
1 If (complex event list)
  occurs 1 time
  then Goto Next
3 If (complex event list)
  then Trigger and fill memory
  Else if (complex event list)
  then Goto 1
```

This one may:

```
1 If anything
  occurs 1 time
  then Goto Next
2 If (complex event list)
  occurs 1 time
  then Goto Next
3 If (complex event list)
  then Trigger and fill memory
  Else if (complex event list)
  then Goto 1
```

Specific Guidelines - 800 Mb/s State Mode

- Labels that span pods (split labels) require more combiner resources than labels with bits that all belong to a single pod. Whenever possible, define labels that do not span pods. In some cases, the compiler will be able to combine 2 non-split labels that are ANDed together even though it fails to compile a pattern on a single label that spans pods.

- Cannot specify more than 4 patterns or 2 ranges per pod.

     Non-split patterns may use operations: =, !=, <, <=,>, >=, In range, Not in range.

     Split patterns may only use operations: =, !=.

     Patterns on labels with re-ordered bits may only use operations: =, != (same as Timing and 200 and 400 Mb/s modes).

- Restrictions on the way event resources can be used and combined vary with the type of trigger function selected.

     The "Find Pattern" and "Find n Patterns in Immediate Sequence" trigger functions can use all of the available pattern events in any sequence level. However, some combinations (ANDing and ORing) of pattern events, while not exceeding the maximum number of pattern resources, can use too many combiner resources. Any combination of 3 of the flag events and "Wait for arm in from IMB" event can also be combined with the pattern events in each sequence level.

     When using the "Find n Patterns in Eventual Sequence" trigger functions, only non-split labels may be used and only one pattern event per sequence level may be used. The interface allows insertion of other events in each sequence level to allow ANDing or ORing of the pattern event with other event types (flags or arm in) or replacement of the pattern event with another event type. However, if more than one pattern event is specified the trigger compiler will be unable to compile it.

- Pattern events used in the Default Storing Control count against the number of available resources.

Specific Guidelines - 1250/1500 Mb/s State Mode

- Labels that span pods (split labels) require more combiner resources than labels with bits that all belong to a single pod. Whenever possible, define labels that do not span pods. In some cases, the compiler will be able to

combine 2 non-split labels that are ANDed together even though it fails to compile a pattern on a single label that spans pods.

- Cannot specify more than 3 patterns or 1 range per pod.

  Non-split patterns may use operations: =, !=, <, <=,>, >=, In range, Not in range.

  Split patterns may only use operations: =, !=.

  Patterns on labels with re-ordered bits may only use operations: =, != (same as Timing and 200 and 400 Mb/s modes).

- Restrictions on the ways event resources can be used and combined vary with the type of trigger function selected.

  The "Find Pattern" and "Find 2 Patterns in Immediate Sequence" trigger functions can use all of the available pattern events in any sequence level. However, some combinations (ANDing and ORing) of pattern events, while not exceeding the maximum number of pattern resources, can use too many combiner resources. Any combination of 3 of the flag events and "Wait for arm in from IMB" event can also be combined with the pattern events in each sequence level.

  When using the "Find 2 Patterns in Eventual Sequence" trigger function, only non-split labels may be used and only one pattern event per sequence level may be used. The interface allows insertion of other events in each sequence level to allow ANDing or ORing of the pattern event with other event types (flags or arm in) or replacement of the pattern event with another event type. However, if more than one pattern event is specified the trigger compiler will be unable to compile it.

- Pattern events used in the Default Storing Control count against the number of available resources.

Cannot specify range on label with clock bits that span pod pairs

A label that contains clock bits being used as data bits, can only be included in a range term if the clock bits are confined to a single pod pair.

Counter value checked as an event, but no increment action specified

This warning occurs because you have used a counter in your *trigger sequence*, but do not have *Counter Increment* as an action. You do not need to increment the counter in the same sequence level. The counter event will still function, but will not change value. The default value for both counters is 0.

Goto action specifies an undefined level

The "undefined level" messages mean that the trigger sequence contains goto statements that point to non-existent levels. This is detected when the trigger sequence is evaluated. The logic analyzer will not run if there are undefined levels, even if there is no possibility of the goto sequence being called.

Possible Causes

• The last sequence level calls "goto next"

To check this, select the *Overview* subtab under the Trigger tab.

To fix, find the "goto next" statement and change it to point to an existing level.

Hardware Initialization Failed

Please go to System Administration Tools and run the Self-Test Utility (see page 114) on the logic analyzer. If you have failures, contact your Agilent Technologies Sales Office for service or software upgrades.

Maximum of 32 Channels Per Label

The logic analyzer can only assign up to 32 channels for each label. If you need more than 32 channels, assign them to two labels and use the labels in conjunction.

Must assign another pod pair to specify actions for flags

In state sampling mode, when there is only one *pod pair* assigned to an analyzer, flags are not available. You must assign another pod pair to the analyzer in order for flags to be available.

In the timing sampling mode, flags are are always available.

See Also   "To assign pods to the analyzer" on page 57

Must assign Pod 1 on the master card to specify actions for flags

When using a 16760A analyzer in 200Mb/s state mode, Pod 1 on the master card must be assigned in order to add actions for the flags in a branch action list. To assign the pod, go to the format tab and select the "Pod Assignment" button. Drag Pod 1 of the master card from the unassigned pods list to the analyzer assigned list.

No more Edge/Glitch resources available for this pod pair

This error occurs when you have used more than 2 edges or glitches per *pod pair* in the *trigger specification*.

Possible Solutions

• Phrase some of the edges as patterns.

For example, if you are looking for a rising edge on a read/write line, you can check for R/W = 0 in one level followed by R/W = 1 in the next level.

• Move some of the edges to another pod pair.

Even if a label spans pod pairs, only the edge resources of the pod pair the specific channel is on are used.

No more Pattern resources available for this pod pair

This error occurs when you have used up all the pattern resources available. Each *pod pair* has about 28 pattern resources. Some pattern events use more than 1 resource.

Possible Solutions

• Keep labels within a pod pair

If a label (bus) spans pod pairs (for example, pods 2 and 3) then when you use the label in a *trigger sequence* it will use up at least one pattern resource on both pod pairs. If you hook up your probes in such a way that the signals are on a single pod pair, you can free up a pattern resource on the other pod pair.

• Move some labels to another pod pair

Each pod pair has its own set of pattern resources. Putting your two most-used labels on different pod pairs can improve your resource usage.

You may find more information in the Branch expression is too complex (see page 212) error message help for the 800 and 1250 Mb/s modes.

No Trigger action found in the trace specification

This warning occurs when the trigger sequence you specified does not have at least one *trigger and fill memory* or *trigger and goto* action. The analyzer will still acquire data, but you will need to manually stop it. For example, when you use the *Run until user stop* function, to avoid popping up the status window, set the popup level to errors only.

Slow or Missing Clock

The message "Slow or Missing Clock" only appears in *state measurement*s. However, if you have another instrument armed by the state analyzer, a slow or missing clock on the state analyzer will prevent the other instrument from triggering also.

Possible Causes

- Target system is not running properly

  Check that the system is running properly. The logic analyzer and other probing fixtures such as pin extenders can place too much capacitive load on a system.

- Incorrect clock specification

  Make sure the device under test clock matches the clock specified under *Sampling*.
  Also check that the probe's clock channels are attached to the device under test's clock lines either directly or through an analysis probe.
  If you are using an analysis probe, the probe's User's Guide should show the correct connections and settings.

- Bad probe connection

  Check that the probe is securely attached to the clock line and is receiving a signal. The logic analyzer shows activity indicators under the *Sampling* and *Format* tabs.

- Incorrect signal level

The clock's threshold level is set by the pod threshold. For the logic analyzer's J clock, check the pod threshold of pod 1 of the *master card*.

Timer value checked as an event, but no start action specified

This warning occurs because you have used a timer in your *trigger sequence*, but have not started it with either *Start from reset* or *Resume* in any action. You do not need to start the timer in the same sequence level. The timer will still function if not started, but will not change value.

Trigger function initialization failure

The "trigger function initialization failure" messages mean that you tried to insert a trigger function which required a change to the trigger function default state.

Possible Causes

• Tried to insert "Wait for arm in" trigger function

  A "Wait for arm in" trigger level causes the logic analyzer to wait for a signal from another module or Port In. These signals are passed through the Intermodule Bus. To prevent the logic analyzer from hanging, it must be added to the Group Run Arming Tree. To do this, open the Intermodule window by selecting the Intermodule icon in the System window. In the Intermodule window, select the analyzer icon and select any option except for Independent.

• Tried to insert "Wait for other machine to trigger" function

  A "Wait for the other machine to trigger" trigger level causes the logic analyzer to wait for a signal from the other logic analyzer *machine* in the module. If this machine is not on, the current logic analyzer will hang. To turn the other machine on, select *Pod Assignment* under *Format*. Set the type of the other machine to *State* or *Timing*.

• In *Format*, no *labels* have bits assigned to them.

When you insert a trigger function, the logic analyzer sets up a field for you to enter values. The field length is based on the number of bits assigned to the first active label, or the label you specify. If there are no bits assigned to the label, the logic analyzer cannot complete the value field.

See Also   Using the Intermodule Window (see the *Agilent Technologies 16700A/B-Series Logic Analysis System* help volume)

"To assign probe channels to labels" on page 60

Trigger inhibited during timing prestore

The "trigger inhibited" informational message appears when you have a logic analyzer making a conventional *timing measurement*, and it is set to a slow sample rate. The logic analyzer will fill the designated amount of pre-trigger memory before checking for the trigger condition.

To calculate how long this should take, multiply the sample rate by the percentage of pre-trigger memory and the acquisition depth. For example, if

sample period = 1.0 ms (sample rate = 10^3 samples/sec.)
trigger position = center (percentage of pre-trigger memory = 50%)
acquisition depth = 64K (roughly 64 x 10^3 samples)

then the approximate time is 32 seconds.

Trigger Specification is too complex

The "Trigger Specification is too complex" message means that the trigger sequence contains more unique event list expressions than can be allocated to the available combiner resources in the analyzer hardware.

The analyzer has a maximum limit of 16 event list combiner resources. Each unique event list expression requires the use of at least one of these combiner resources. A complex event list may require more than one combiner resource.

The message does not mean that any single event list expression was too complex to combine (see "Branch expression is too complex" on page 212), but that the overall number of unique branch expressions specified has exceeded the limit of 16.

In order to compile and run, the total number of unique event list expressions must be reduced to 16, and the complexity of some of the expressions may have to also be reduced.

Branch expressions that are identical (and simple enough to be combined by a single combiner resource) share the same combiner resource. Reusing identical event list equations where possible will optimize the use of combiner resources (see page 224).

You may find more information in the "Branch expression is too complex" on page 212 error message help for the 800 and 1250/1500 Mb/s modes.

Combiner resource allocation guidelines:

- Labels that span multiple pod pairs (split labels) increases the number of required combiner resources as compared with labels that are entirely contained within a single pod pair.

    Whenever possible try to arrange the probing such that labels do not span pod pairs. This is the single most effective way to reduce the number of required combiner resources.

NOTE:
For labels that do span pod pairs, the complexity can be reduced to the same as that of the non-split label case if all bits in the label on all but one pod pair can be set to Xs in the event list expression for the measurement.

For example, if label ADDR has its 16 most significant bits on pod A3 and 16 least significant bits on pod A2 (spanning pod pairs A4/A3 and A2/A1), the complexity of the compiled expression will be reduced if all 16 most significant bits or all 16 least significant bits are set to Xs in the pattern event.

- Event lists with up to 4 unique pattern events can be combined in any combination of ANDs and ORs by a single combiner resource if all of the pattern labels are non-split and contained on the same pod pair.

  Combining more than 4 labels on the same pod pair will require another combiner resource.

- Non-split label pattern events from different pod pairs that are ORed together require an additional combiner resource for each additional pod pair included in the event list. (ANDing on non-split patterns from different pod pairs does not increase the required number of combiner resources).

- An inequality compare (<,<=,>,>=) with a split label pattern event requires 2 combiner resources.

- A range on a split label pattern event requires 4 combiner resources.

- The event list in the custom store qualification dialog also allocates combiner resources from the same pool of 16 resources. If the store qualification event list equation is the same as one of the branch event list equations in the trigger sequence, the combiner resource will be shared. A unique store qualification event list requires the allocation of 1 (or more) of the combiner resources.

- 400/800/1250/1500 Mb/s state modes requires many more combiner resources to implement the same trigger sequence as compared to 200 Mb/s state modes and all timing modes. Refer the discussion of complexity in the "Branch expression is too complex" on page 212 help topic.

Waiting for Trigger

This message indicates that the specified trigger pattern has not occurred. This may be expected, as when you are waiting to trigger on an unusual event.

Possible Causes

- Misaligned boundaries for addresses

  When the device under test is a microprocessor that fetches only from long-word aligned addresses, if the trigger is set to look for an opcode

fetch at an address that is not properly aligned, the trigger will never be found.

- Trigger set incorrectly

Some strategies you can use when verifying or debugging trigger sequence levels are:

- Look at the run status message line or open the Run Status window. It will tell you what level of the sequence the logic analyzer is in.

- Stop the measurement and look at the data that was captured. This is particularly useful when you use *store qualifiers* to store "no states" (or only the states you are interested in) and the branches taken are stored.

- Save the trigger setup, then simplify it to see what part of the sequence does get captured. When you learn what needs to be changed, you can recall the original trigger setup and make changes to it.

See Also

"To specify default storing" on page 76

"To save a trigger setup" on page 90

Specifications and Characteristics

NOTE: For a complete comparison of all logic analyzer specifications and characteristics refer to the *Agilent Technologies 16700 Series Logic Analysis System Product Overview* which can be downloaded from the 16760A product page on the Agilent web site.

Probes

- "E5378A Single-Ended Probe Specifications and Characteristics" on page 228

- "E5379A Differential Probe Specifications and Characteristics" on page 228

- "E5380A MICTOR-Compatible Probe Specifications and Characteristics" on page 229

Synchronous State Analysis

- "1500 Mb/s Sampling Mode Specifications and Characteristics" on page 230

- "1250 Mb/s Sampling Mode Specifications and Characteristics" on page 231

- "800 Mb/s Sampling Mode Specifications and Characteristics" on page 232

- "400 Mb/s Sampling Mode Specifications and Characteristics" on page 233

- "200 Mb/s Sampling Mode Specifications and Characteristics" on page 234

Asynchronous Timing Analysis

- "Conventional Timing Mode Specifications and Characteristics" on page 235

- "Transitional Timing Mode Specifications and Characteristics" on page 235

Environmental   Operating temperature: 0 to 45 degrees C

See Also   "What is a Specification?" on page 236

"What is a Characteristic?" on page 237

E5378A Single-Ended Probe Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Input resistance and capacitance:        (refer to figure below)
Maximum state data rate supported:       1250 Mb/s
Mating connector:                        Agilent part # 1253-3620 (see note 1)
Minimum voltage swing:                   250 mV p-p
Input dynamic range:                     -3.0 Vdc to +5 Vdc
Threshold accuracy*:                     +/- (30 mV +1% of setting) (see note 3)
Threshold range:                         -3.0 V to +5 V
User-supplied threshold range:           -3.0 V to +5 V
User-supplied threshold input
   resistance:                           >= 100K ohms
Threshold control options:               User-provided input,
                                         Adjustable from user interface
Maximum non-destructive input voltage: +/- 40 Vdc
Maximum input slew rate:                 5 V/ns
Clock input:                             Differential
Number of inputs (see note 5):           34 (32 data and 2 clock)

Note 1:  A support shroud, Agilent part # 16760-02302 (for
         boards up to 0.062 in. thick) or 16760-02303 (for
         boards up to 0.120 in. thick) is recommended.  A kit
         containing 5 shrouds and 5 connectors is available
         as Agilent part # 16760-68702 (for boards up to
         0.062 in. thick) or 16760-68703 (for boards up to
         0.120 in. thick).

Note 3:  User-supplied threshold input grounded.

Note 5:  Refer to the specific modes of operation for
         details on how inputs can be used.
```
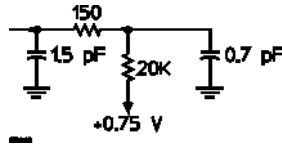

E5378A Input Equivalent Circuit

E5379A Differential Probe Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Input resistance and capacitance:        (refer to figure below)
```

```
Maximum state data rate supported:    1250 Mb/s
Mating connector:                     Agilent part # 1253-3620 (see note 1)
Minimum voltage swing:                Vpos - Vneg >= 200 mV p-p
Input dynamic range:                  -3.0 Vdc to +5 Vdc
Threshold accuracy:                   +/- (30 mV +1% of setting) (see note 4)
Threshold range:                      -3.0 V to +5 V
User-supplied threshold range:        N/A
User-supplied threshold input
    resistance:                       N/A
Threshold control options:            If operated single-ended (minus inputs
                                      grounded) the threshold can be adjusted
                                      from the user interface.
Maximum non-destructive input voltage: +/- 40 Vdc
Maximum input slew rate:              5 V/ns
Clock input:                          Differential
Number of inputs (see note 5):        17 (16 data and 1 clock)

Note 1:   A support shroud, Agilent part # 16760-02302 (for
          boards up to 0.062 in. thick) or 16760-02303 (for
          boards up to 0.120 in. thick) is recommended.  A kit
          containing 5 shrouds and 5 connectors is available
          as Agilent part # 16760-68702 (for boards up to
          0.062 in. thick) or 16760-68703 (for boards up to
          0.120 in. thick).

Note 4:   The threshold specified in the user interface
          is added to the difference between the negative
          and positive differential inputs.

Note 5:   Refer to the specific modes of operation for
          details on how inputs can be used.
```
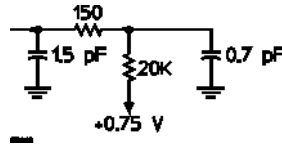

E5379A Input Equivalent Circuit

E5380A MICTOR-Compatible Probe Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Input resistance and capacitance:     (refer to figure below)
Maximum state data rate supported:    600 Mb/s
Mating connector:                     AMP MICTOR 38 (see note 2)
Minimum voltage swing:                300 mV p-p
Input dynamic range:                  -3.0 Vdc to +5 Vdc
Threshold accuracy:                   +/- (30 mV +1% of setting)
Threshold range:                      -3.0 V to +5 V
User-supplied threshold range:        N/A
User-supplied threshold input
    resistance:                       N/A
Threshold control options:            Adjustable from UI
Maximum non-destructive input voltage: +/- 40 Vdc
```

```
Maximum input slew rate:              5 V/ns
Clock input:                          Single-ended
Number of inputs (see note 5):        34 (32 data and 2 clock)

Note 2:  A kit containing 5 AMP MICTOR connectors and
         5 support shrouds is available, Agilent part
         # E5346-68701.  A support shroud is available
         separately, Agilent part # E5346-44701.

Note 5:  Refer to the specific modes of operation for
         details on how inputs can be used.
```
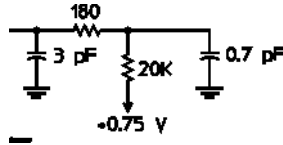

E5380A Input Equivalent Circuit

1500 Mb/s Sampling Mode Specifications and Characteristics

NOTE:

All specifications are marked by "*" (asterisk).

```
Maximum data rate:                    1500 Mb/s
Minimum clock interval (active edge
   to active edge)*:                  0.667 ns
Clock periodicity:                    Clock must be periodic
Number of clocks/qualifiers:          1
Clock polarity:                       Both edges
Minimum data pulse width:             600 ps
Number of Channels (see note 1)
    With time tags:                   16 x (number of modules) - 8
    Without time tags:                16 x (number of modules)
Maximum channels on single time
   base and trigger:                  80 (5 modules)
Maximum memory depth:                 128 M samples
Setup and Hold Time Range (manual
   adjustment),
       single edge:                   N/A
       multiple edge:                 +0.5 ns / 0 ns to -3.5 ns / +4.0 ns
Window width after running eye finder: 500 ps
Adjustment resolution:                10 ps
Time tag resolution:                  4 ns (see note 2)
Maximum time count between states:    17 sec
Trigger resources:                    2 patterns; 4 flags; Arm In
Trigger actions:                      Trigger and fill memory
Maximum trigger sequence levels:      2
Maximum trigger sequencer speed:      1500 MHz
Store qualification:                  Default
Maximum global counter:               N/A
Maximum occurrence counter:           N/A
Timer value range:                    N/A
Timer resolution:                     N/A
Timer accuracy:                       N/A
Timer reset latency:                  N/A
```

```
Data in to BNC port out:            150 ns
Flag set/reset to evaluation:       N/A

Note 1:  In the 1500 Mb/s mode, only the even-numbered
         channels (0,2,4,...etc.) are acquired.

Note 2:  The resolution of the hardware used to assign
         time tags is 4 ns.  Times of intermediate states
         are calculated.
```

1250 Mb/s Sampling Mode Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Maximum data rate:                  1250 Mb/s
Minimum clock interval (active edge
   to active edge)*:                0.800 ns
Clock periodicity:                  Clock must be periodic
Number of clocks/qualifiers:        1
Clock polarity:                     Both edges
Minimum data pulse width:           750 ps
Number of Channels (see note 1)
   With time tags:                  16 x (number of modules) - 8
   Without time tags:               16 x (number of modules)
Maximum channels on single time
   base and trigger:                80 (5 modules)
Maximum memory depth:               128 M samples
Setup and Hold Time Range (manual
   adjustment),
      single edge:                  N/A
      multiple edge:                +2.5 ns / -1.5 ns to -1.5 ns / +2.5 ns
Window width (manual adjustment),
   single edge:                     N/A
   multiple edge*:                  1 ns
Window width after running eye finder: 500 ps
Adjustment resolution:              10 ps
Time tag resolution:                4 ns (see note 2)
Maximum time count between states:  17 sec
Trigger resources:                  2 patterns; 4 flags; Arm In
Trigger actions:                    Trigger and fill memory
Maximum trigger sequence levels:    2
Maximum trigger sequencer speed:    1250 MHz
Store qualification:                Default
Maximum global counter:             N/A
Maximum occurrence counter:         N/A
Maximum pattern/range term width:
Timer value range:                  N/A
Timer resolution:                   N/A
Timer accuracy:                     N/A
Timer reset latency:                N/A
Data in to BNC port out:            150 ns
Flag set/reset to evaluation:       N/A

Note 1:  In the 1250-Mb/s mode, only the even-numbered
         channels (0,2,4,...etc.) are acquired.

Note 2:  The resolution of the hardware used to assign
         time tags is 4 ns.  Times of intermediate states
         are calculated.
```

800 Mb/s Sampling Mode Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Maximum data rate:
   E5378A, E5379A probes:              800 Mb/s
   E5380A probe:                       600 Mb/s
Minimum clock interval (active edge
   to active edge)*:
      E5378A, E5379A probes:           1.25 ns
      E5380A probe:                    1.67 ns
Minimum state clock pulse width:
   E5378A, E5379A probes:              750 ps
   E5380A probe:                       1.5 ns
Clock periodicity:                     Periodic or Aperiodic
Number of clocks/qualifiers:           1
Clock polarity:                        Rising, falling, or both
Minimum data pulse width:
   E5378A, E5379A probes:              750 ps
   E5380A probe:                       1.5 ns
Number of Channels
   With time tags:                     34 x (number of modules) - 17
   Without time tags:                  34 x (number of modules) - 1
Maximum channels on single time
   base and trigger:                   170 (5 modules)
Maximum memory depth:                  64 M samples
Setup and Hold Time Range (manual
   adjustment),
      single edge:                     +2.5 ns / -1.5 ns to -1.5 ns / +2.5 ns
      multiple edge:                   +2.5 ns / -1.5 ns to -1.5 ns / +2.5 ns
Window width (manual adjustment),
   single edge:
      E5378A, E5379A probes*:          1 ns
      E5380A probe:                    1.5 ns
   multiple edge:
      E5378A, E5379A probes:           1 ns
      E5380A probe:                    1.5 ns
Window width after running eye finder:
   E5378A, E5379A probes:              500 ps
   E5380A probe:                       1 ns
Adjustment resolution:                 10 ps
Time tag resolution:                   4 ns (see note 2)
Maximum time count between states:     17 sec
Trigger resources:                     2 patterns; 4 flags; Arm In
Trigger actions:                       Trigger and fill memory
Maximum trigger sequence levels:       4
Maximum trigger sequencer speed:       800 MHz
Store qualification:                   Default
Maximum global counter:                N/A
Maximum occurrence counter:            N/A
Maximum pattern/range term width:
Timer value range:                     N/A
Timer resolution:                      N/A
Timer accuracy:                        N/A
Timer reset latency:                   N/A
Data in to BNC port out:               150 ns
Flag set/reset to evaluation:          N/A


Note 2:  The resolution of the hardware used to assign
         time tags is 4 ns.  Times of intermediate states
         are calculated.
```

400 Mb/s Sampling Mode Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Maximum data rate:                    400 Mb/s
Minimum clock interval (active edge
    to active edge)*:                 2.5 ns
Minimum state clock pulse width:      1.5 ns
Clock periodicity:                    Periodic or Aperiodic
Number of clocks/qualifiers:          1
Clock polarity:                       Rising, falling, or both
Minimum data pulse width:             1.5 ns
Number of Channels
    With time tags:                   34 x (number of modules) - 17
    Without time tags:                N/A
Maximum channels on single time
    base and trigger:                 153 (5 modules)
Maximum memory depth:                 32 M samples
Setup and Hold Time Range
    (manual adjustment),
        single edge:                  +4.5 ns / -2.0 ns to -2.0 ns / +4.5 ns
        multiple edge:                +5.0 ns / -2.0 ns to -1.5 ns / +4.5 ns
Window width (manual adjustment)*,
    single edge:                      2.5 ns
    multiple edge:                    3.0 ns
Window width after running eye finder: 1.25 ns
Adjustment resolution:                100 ps
Time tag resolution:                  4 ns
Maximum time count between states:    17 sec
Trigger resources:                    8 patterns (=,/=,<,>,<=,>=),
                                      4 ranges (in range, not in range),
                                      2 occurrence counters,
                                      4 flags
Trigger actions:                      Go to, trigger and fill memory
Maximum trigger sequence levels:      16
Maximum trigger sequencer speed:      400 MHz
Store qualification:                  Default
Maximum global counter:               N/A
Maximum occurrence counter:           N/A
Maximum pattern/range term width:     32 bits
Timer value range:                    N/A
Timer resolution:                     N/A
Timer accuracy:                       N/A
Timer reset latency:                  N/A
Data in to BNC port out:              150 ns
Flag set/reset to evaluation:         N/A


Note 2:  The resolution of the hardware used to assign
         time tags is 4 ns.  Times of intermediate states
         are calculated.
```

200 Mb/s Sampling Mode Specifications and Characteristics

NOTE:
All specifications are marked by "*" (asterisk).

```
Maximum data rate:                      200 Mb/s
Minimum clock interval (active edge
    to active edge)*:                   5 ns
Minimum state clock pulse width:        1.5 ns
Clock periodicity:                      Periodic or Aperiodic
Number of clocks/qualifiers:            1
Clock polarity:                         Rising falling or both
Minimum data pulse width:               1.5 ns
Number of Channels
    With time tags:                     34 x (number of modules)
    Without time tags:                  34 x (number of modules)
Maximum channels on single time
    base and trigger:                   180 (5 modules)
Maximum memory depth:                   32 M samples
Setup and Hold Time Range
    (manual adjustment),
        single edge:                    +4.5 ns / -2.0 ns to -2.0 ns / +4.5 ns
        multiple edge:                  +5.0 ns / -2.0 ns to -1.5 ns / +4.5 ns
Window width (manual adjustment)*,
    single edge:                        2.5 ns
    multiple edge:                      3.0 ns
Window width after running eye finder: 1.25 ns
Adjustment resolution:                  100 ps
Time tag resolution:                    4 ns
Maximum time count between states:      17 sec
Trigger resources:                      16 patterns (=,/=,<,>,<=,>=),
                                        15 ranges (in range, not in range),
                                        Timers = (number of pods assigned),
                                        2 global counters,
                                        1 occurrence counter per sequence level,
                                        4 flags
Trigger actions:                        Go to,
                                        Trigger and Go to,
                                        Trigger and fill memory,
                                        Store/don't store sample,
                                        Turn default storing on/off,
                                        Timer (start,stop,pause,resume),
                                        Global counter increment.reset,
                                        Occurrence counter reset,
                                        Flag set/clear
Maximum trigger sequence levels:        16
Maximum trigger sequencer speed:        200 MHz
Store qualification:                    Default and per sequence level
Maximum global counter:                 16,777,215
Maximum occurrence counter:             16,2777,215
Maximum pattern/range term width:       32 bits
Timer value range:                      100 ns to 4397 sec
Timer resolution:                       4 ns
Timer accuracy:                         +/- (10 ns + 0.01% of value)
Timer reset latency:                    60 ns
Data in to BNC port out:                150 ns
Flag set/reset to evaluation:           110 ns
```

Conventional Timing Mode Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Maximum timing analysis sample rate:
    Full channels:                      400 MHz
    Half channels:                      800 MHz
Number of channels:                     34 x (number of modules)
Maximum channels on a single time
    base and trigger:                   170 (5 modules)
Sample period:                          1.25 ns
Sample period accuracy:                 +/- (100ps + 0.01% of sample period)
Channel-to-channel skew:                < 1.5 ns
Time interval accuracy:                 +/- [sample period +
                                            (chan-to-chan skew) +
                                            (0.01% of time interval)]

Minimum data pulse width:
    At 800 MHz sampling:                1.5 ns
    At 400 MHz sampling:                3.0 ns, 5 ns for trigger sequencing
Maximum trigger sequencer speed:        200 MHz
Trigger resources:                      16 patterns (=,/=,<,>,<=,>=),
                                        15 ranges (in range, not in range),
                                        2 edge/glitch,
                                        Timers = (number of pods assigned) -1,
                                        2 global counters,
                                        1 occurrence counter per sequence level,
                                        4 flags
Trigger resource conditions:            Arbitrary boolean combinations
Maximum global counter:                 16,777,215
Maximum occurrence counter:             16,777,215
Timer value range:                      100 ns to 4397 sec
Timer resolution:                       4 ns
Timer accuracy:                         +/- (10 ns + 0.01%)
Greater than duration:                  5 ns to 83 ms in 5 ns increments
Less than duration:                     10 ns to 83 ms in 5 ns increments
Timer reset latency:                    60 ns
Data in to BNC port out delay:          150 ns
Flag set/reset to evaluation:           110 ns
```

Transitional Timing Mode Specifications and Characteristics

NOTE: All specifications are marked by "*" (asterisk).

```
Maximum timing analysis sample rate:    400 MHz
Number of channels:
    For sample rates < 400 MHz:         34 x (number of modules)
    For sample rates = 400 MHz:         34 x (number of modules) - 17
Maximum channels on a single time
    base and trigger:                   170 (5 modules)
Sample period:                          2.5 ns to 1 ms
Sample period accuracy:                 +/- (100ps + 0.01% of sample period)
Channel-to-channel skew:                < 1.5 ns
```

```
Time interval accuracy:              +/- [sample period +
                                          (chan-to-chan skew) +
                                          (0.01% of time interval)]
Minimum data pulse width:            3.7 ns, 5.0 ns for trigger sequencing
Maximum trigger sequencer speed:     200 MHz
Trigger resources:                   16 patterns (=,/=,<,>,<=,>=),
                                     15 ranges (in range, not in range),
                                     2 edge/glitch,
                                     Timers = (number of pods assigned) -1
                                     2 global counters,
                                     1 occurrence counter per sequence level,
                                     4 flags
Trigger resource conditions:         Arbitrary boolean combinations
Maximum global counter:              16,777,215
Maximum occurrence counter:          16,777,215
Timer value range:                   100 ns to 4397 sec
Timer resolution:                    4 ns
Timer accuracy:                      +/- (10 ns + 0.01%)
Greater than duration:               5 ns to 83 ms in 5 ns increments
Less than duration:                  10 ns to 83 ms in 5 ns increments
Timer reset latency:                 60 ns
Data in to BNC port out delay:       150 ns
Flag set/reset to evaluation:        110 ns
```

What is a Specification?

A *Specification* is a numeric value, or range of values, that bounds the performance of a product parameter. The product warranty covers the performance of parameters described by specifications. Products shipped from the factory meet all specifications. Additionally, the products sent to Agilent Technologies Customer Service Centers for calibration and returned to the customer meet all specifications.

Specifications are verified by *Calibration Procedures*.

What is a Calibration Procedure?
Calibration procedures verify that products or systems operate within the specifications. Parameters covered by specifications have a corresponding calibration procedure. Calibration procedures include both performance tests and system verification procedure. Calibration procedures are traceable and must specify adequate calibration standards.

Calibration procedures verify products meet the specifications by comparing measured parameters against a pass-fail limit. The pass-fail limit is the specification less any required guardband.

The term "calibration" refers to the process of measuring parameters and referencing the measurement to a calibration standard rather than the process of adjusting products for optimal performance, which is

referred to as an "operational accuracy calibration".

What is a Characteristic?

Characteristics describe product performance that is useful in the application of the product, but that is not covered by the product warranty. Characteristics describe performance that is typical of the majority of a given product, but not subject to the same rigor associated with specifications.

Characteristics are verified by *Function Tests*.

What is a Function Test?

Function tests are quick tests designed to verify basic operation of a product. Function tests include operator's checks and operation verification procedures. An operator's check is normally a fast test used to verify basic operation of a product. An operation verification procedure verifies some, but not all, specifications, and often at a lower confidence level than a calibration procedure.

6

Concepts

- "Understanding Logic Analyzer Triggering" on page 240
- "Understanding State Mode Sampling Positions" on page 256
- "Understanding Eye Scan Measurements" on page 259

Understanding Logic Analyzer Triggering

Setting up logic analyzer triggers can be difficult and time-consuming. You could assume that if you know how to program, you should be able to set up a logic analyzer trigger with no difficulty. However, this is not true because there are many concepts that are unique to logic analysis. The purpose of this section is to describe these key concepts and how to use them effectively.

- "The Conveyor Belt Analogy" on page 240

- "Summary of Triggering Capabilities" on page 242

- "Sequence Levels" on page 242

- "Boolean Expressions" on page 245

- "Branches" on page 246

- "Edges" on page 246

- "Ranges" on page 246

- "Flags" on page 247

- "Occurrence Counters and Global Counters" on page 247

- "Timers" on page 248

- "Storage Qualification" on page 249

- "Strategies for Setting Up Triggers" on page 251

- "Conclusions" on page 255

See Also   "Setting Up Triggers and Running Measurements" on page 69

The Conveyor Belt Analogy

The memory of a logic analyzer can be compared to a very long conveyor belt, and the samples acquired from the Device Under Test (DUT) as boxes on the conveyor belt. At one end, new boxes are

placed on the conveyor belt, and at the other end the boxes fall off. In other words, because logic analyzer memory is limited in depth (number of samples), whenever a new sample is acquired the oldest sample currently in memory is thrown away if the memory is full. This is shown in the following figure.
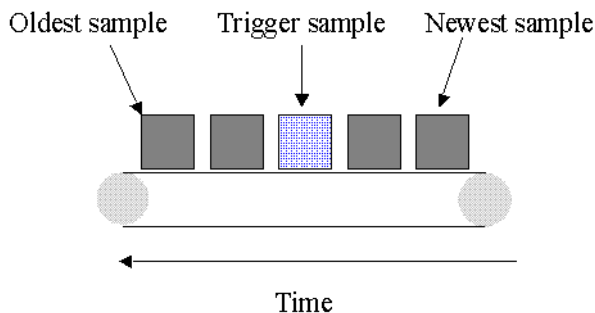
The conveyor belt analogy

A logic analyzer trigger is similar to someone standing at the beginning of the conveyor belt placing more boxes on it. They are told to "look for a special box and to stop the conveyor belt when that box reaches a particular position on the belt". Using this analogy, the special box is the trigger. Once a logic analyzer detects a sample that matches the trigger condition, this is the indication that it should stop acquiring more samples when the trigger is located appropriately in memory.

The location of the trigger in memory is known as the *trigger position*. Normally, the trigger position is set to the middle so that the maximum number of samples that occurred before and after the trigger are in memory. However, you can set the trigger position to any point in memory.

The concepts in this analogy are summed up in the following table.

```
Mapping of concepts in the Conveyor Belt Analogy
to a Logic Analyzer

Conveyor Belt Analogy      Logic analyzer
=====================      ============================
Boxes on the belt          Samples acquired from the
                           device under test
---------------------      ------------------------------
Number of boxes that       Memory depth
will fit on the belt
---------------------      ------------------------------
```

```
Special box          Trigger point
--------------------  -----------------------------
```

Summary of Triggering Capabilities

Because logic analyzer triggering provides a great deal of functionality, the following table provides a brief summary of the capabilities covered in this article. Each of these capabilities will be described.

```
Summary of Logic Analyzer Triggering Capabilities

Capability            Examples
====================  ==============================================
Edges                 If there is rising edge on SIG1 then Trigger
                      If there is falling edge on SIG1 then Trigger
--------------------  ----------------------------------------------
Boolean expressions   If ADDR = 1000 and DATA = 2000
--------------------  ----------------------------------------------
Ranges                If ADDR in range 1000 to 2000
--------------------  ----------------------------------------------
Storage qualification 1. If..
                         Else If ADDR in range 1000 to 2000 then
                            Store Sample
                            Go to 1
                         Else If ADDR not in range 1000 to 2000 then
                            Don't Store Sample
                            Go to 1
--------------------  ----------------------------------------------
Counters              1. If DATA = 1000 Then
                            Increment Counter 1
                            Go to 2
                      2. If Counter 1 > 2 Then
                            Trigger
--------------------  ----------------------------------------------
Timers                1. If DATA = 1000 Then
                            Start Timer 1
                            Go to 2
                      2. If Timer 1 > 500 ns Then
                            Trigger
--------------------  ----------------------------------------------
```

Sequence Levels

While logic analyzer triggers are often simple, they can require complex programming. For example, you may want to trigger on the rising edge of one signal that is followed by the rising edge of another signal. This means that the logic analyzer must first find the first rising

edge before it begins looking for the next rising edge. Because there is a sequence of steps to find the trigger, this is known as a *trigger sequence*. Each step of the sequence is called a *sequence level*.

Each sequence level consists of two parts; the conditions and the actions. The conditions are Boolean expressions such as "If ADDR = 1000" or "If there is a rising edge on SIG1". The actions are what the logic analyzer should do if the condition is met. Examples of actions include triggering the logic analyzer, going to another sequence level, or starting a timer. This is similar to an If/Then statement in programming.

Each sequence level in the trigger sequence is assigned a number. The first sequence level to be executed is always Sequence Level 1, but because of the Go To actions, the rest of the sequence levels can be executed in any order.

When a sequence level is executed and none of the Boolean expressions are true, the logic analyzer acquires the next sample and executes the same sequence level again. As a simple example, consider the following trigger sequence:

```
1. If DATA = 7000 then Trigger
```

If the following samples were acquired, the logic analyzer would trigger on sample #6.

```
Sample #    ADDR   DATA
    1       1000   2000
    2       1010   3000
    3       1020   4000
    4       1030   5000
    5       1040   6000
    6       1050   7000   <- This is where the logic analyzer triggers
    7       1060   2000
```

In essence, Sequence Level 1 is equivalent to "Keep acquiring more samples until DATA=7000, then trigger".

If a Boolean expression in a sequence level is met, another sample is always acquired before the next sequence level is executed. In other words, if a sample meets the condition in Sequence Level 1, another sample will be acquired before executing Sequence Level 2. This means that it is not possible for a single sample to be used to meet the conditions of more than one sequence level. Each sequence level can be thought of as representing events that occur at different points in

time. Two sequence levels can never be used to specify two events that happen simultaneously.

For example, consider the following trigger sequence:

```
1. If ADDR = 1000 then Go to 2
2. If DATA = 2000 then Trigger
```

If the following samples were acquired, the logic analyzer would trigger on sample #7.

```
Sample #    ADDR   DATA
    1       1000   2000   <- This sample meets the condition in Sequence level #1
    2       1010   3000
    3       1020   4000
    4       1030   5000
    5       1040   6000
    6       1050   7000
    7       1060   2000   <- This is where the logic analyzer triggers
```

Note that the logic analyzer will not trigger on Sample #1 because a new sample is acquired between the time that the condition in Sequence level 1 is met and when the condition in Sequence Level #2 is tested. A good way to think of this trigger sequence is "Find ADDR = 1000 followed by DATA = 2000 and then trigger". Multiple sequence levels in a trigger sequence imply a "followed by".

Once a logic analyzer triggers, it does not trigger again. In other words, even if more than one sample meets the trigger condition, the logic analyzer still only triggers once. For example, using "ADDR=1000" as our trigger, if the logic analyzer acquires the following samples, it will trigger on Sample #2 and only on Sample #2.

```
Sample #    ADDR
    1       0000
    2       1000   <- The logic analyzer triggers here
    3       2000
    4       1000   <- The logic analyzer does NOT trigger again here
    5       1040
```

A frequently asked question is "What happens if the conditions in a sequence level are not met?" For example, if there is a condition that says "If ADDR = 1000 Then Trigger", what happens if the current sample has ADDR = 2000? The logic analyzer simply acquires the next sample and tries to execute this sequence level again. In essence, if the trigger condition is "ADDR = 1000", this is equivalent to "Keep acquiring more samples until you find one that has ADDR=1000". Therefore, if you set up a trigger condition that is never met, the logic

analyzer will never trigger.

When the conditions are met in a sequence level, it is clear which sequence level will be executed next when a "Go To" action is used, but it is not necessarily clear if there is no "Go To". On some logic analyzers, if there is no "Go To", this means that the next sequence level should be executed. On other logic analyzers, it means the same sequence level should be executed again. Because of this confusion, it is good practice to always use a "Go To" action rather than relying on the default. The new Agilent Technologies 16715/16/17/18/19A state and timing modules deal with this problem by automatically including a "Go To" or "Trigger" action in every sequence level. For example:

```
If ADDR = 1000 and DATA = 2000 then
Go to 1   <- This is automatically added on the Agilent 16715/16/17/18/19A
```

Next: "Boolean Expressions" on page 245

Boolean Expressions

While multiple sequence levels imply a "followed by", within a sequence level Boolean expressions can be used. An example is:

```
If ADDR = 1000 and DATA = 2000
```

This expression means that for this expression to be met, ADDR must equal 1000 in the same sample that DATA equals 2000. In other words, ADDR equals 1000 at the same time that DATA equals 2000. Therefore, if you want to trigger on two events that occur at the same time, a Boolean expression should be used.

It's a common mistake to try to use two sequence levels when a Boolean expression should be used or to use a Boolean expression when two sequence levels should be used.

NOTE: Boolean expressions are used for events that happen at the same time, and multiple sequence levels are used when one event follows another.

Next: "Branches" on page 246

Branches

Branches are similar to the *Switch* statement in the C programming language and the *Select Case* statement in Basic. They provide a method for testing multiple conditions. Each branch has its own actions. An example of multiple branches is shown below:

```
1. If ADDR < 1000 then Go To 2      <- This is a branch of Level 1
   Else If ADDR > 2000 then Go To 3 <- This is a 2nd branch of Level 1
   Else If DATA = 2000 then Trigger <- This is a 3rd branch of Level 1
2. If DATA <= 7000 then Trigger
3. If there is a Rising Edge on SIG1, then Trigger
```

In sequence level 1, there are three branches, so there are three possible actions that can be taken.

When the condition of one branch is met, none of the branches below it are tested. In other words, there is no way for more than one branch to be executed based upon a single sample, even if the sample causes the conditions for more than one branch to be met. In other words, each branch is an "Else If".

Edges

Edges represent a transition from low to high or high to low on a single signal. Typically, edges are specified as "rising edge", "falling edge", or "either edge", where "rising edge" indicates a transition from a low to a high. On most logic analyzers, up to two edges can be included in the trigger sequence although some allow only one.

Ranges

Ranges are a convenient method for specifying a range of values, such as "ADDR in range 1000 to 2000". Most logic analyzers also support a

"not in range" function as well. Ranges are a convenient shortcut so that you don't have to specify "ADDR >= 1000 and ADDR <= 2000".

Flags

Flags are Boolean variables that are used to send signals from one module to another. They can be set when a condition occurs in one module and tested later by another module. In the example below, flag 1 is used to keep track of what happens in the trigger sequence of Module 1 so that this information can be used in Module 2.

Trigger Sequence for Module 1

```
1. If ADDR < 5000 then
      Set Flag 1
      Trigger and fill memory
```

Trigger Sequence for Module 2

```
1. If DATA = 5000 and Flag 1 is set then Trigger
   Else if DATA = 1000 and not Flag 1 then Trigger
```

Occurrence Counters and Global Counters

Occurrence Counters are used in situations where you want to find the Nth occurrence of an event. For example, if you want to trigger on the 5th time that ADDR = 1000, you could set up the trigger as:

```
If ADDR = 1000 occurs 5 times then Trigger
```

Global Counters are like integer variables. They are more flexible than Occurrence Counters because they can be used to count complex events such as an edge followed by another edge. Global Counters can be incremented, tested, and reset. By default, Global Counters begin with zero and don't need to be reset unless they have already been used in the trigger sequence. In general, Occurrence Counters should

be used in place of Global Counters, if possible, because they are easier to use and because there is a limited number of Global Counters.

Timers

Timers are used to check the amount of time that has elapsed between events. For example, if you want to trigger on one edge followed by another edge that occurs within 500ns, use a timer. The most critical point to remember in using timers is that they need to be started before they are tested. In other words, timers do not start automatically.

The key to setting up a timer is to identify where it should be started and where it should be tested. Consider the example in the following figure. The timer should be started when the rising edge on SIG1 is detected and it should be tested when the rising edge occurs on SIG2.
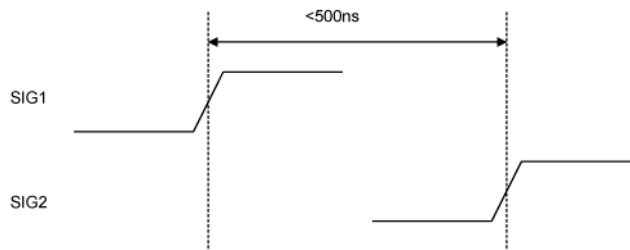
An edge followed by an edge with a time limit

An example trigger sequence to set up this measurement is:

```
1. If there is a Rising Edge on SIG1, then
     Start Timer1
     Go to 2
2. If there is a Rising Edge on SIG2 AND Timer1 < 500ns then
     Trigger
```

While the above trigger sequence seems correct, it actually has a critical flaw. What happens if there is a rising edge on SIG1 but SIG2 doesn't occur within 500ns? The logic analyzer will never trigger,

because timer1 will keep running and condition "Timer1 <500 ns" will never be met. There might be another rising edge on SIG1 that is followed within 500ns by the rising edge on SIG2 that occurs later on, so this situation is unacceptable.

To fix this problem, whenever the timer exceeds 500ns without triggering, the sequence should loop back to Level 1 to look for another rising edge on SIG1. The following shows an example of the correct sequence:

```
1. If there is a Rising Edge on SIG1, then
      Start Timer1
      Go to 2
2. If there is a Rising Edge on SIG2 AND Timer1 < 500ns then
      Trigger
   Else If Timer1 >= 500ns then
      Reset Timer1
      Go to 1
```

Occasionally, you may run out of timers. A counter can be used in place of a timer if the logic analyzer is sampling at regular intervals (that is, if it's in the timing sampling mode). A timer can be simulated by counting the number of samples that are acquired. For example, if the logic analyzer acquires a new sample every 10ns and seven samples are acquired, this represents 70ns.

Storage Qualification

Storage qualification is used to determine if an acquired sample should be stored (that is, placed in memory) or thrown away. This keeps the logic analyzer memory from being filled with samples that are not needed.

Default Storage The simplest method to set up storage qualification is by setting up the Default Storage. This is specified separate from the trigger sequence, such as in a separate tab or another dialog. Default Storage means "unless a sequence level specifies otherwise, this is what should be stored". As an example, you may want to only store samples if ADDR is in the range 1000 to 2000, so you should set the Default Storage to:

```
ADDR In Range 1000 to 2000
```

By default, the Default Storage is set to store all samples acquired. You can also set the Default Storage to store nothing, which means that no samples will be stored unless a sequence level overrides the default storage.

Sequence Level Storage

Sequence level storage qualification means that within a particular sequence level only certain samples will be stored. This means that until a "Go To" or "Trigger" action is used to leave this sequence level, the storage qualification applies. This is useful when you want different storage qualification for each sequence level. For example, you may want to store nothing until ADDR = 1000 and then store only samples with ADDR in the range 1000 to 2000 for the rest of the measurement.

Setting up sequence level storage requires the use of an additional branch. For example, if you want to store only samples with ADDR in the range 5000 to 6FFF while looking for DATA = 005E, the following sequence level could be used in some situations:

```
1. If DATA = 005E then Trigger
   Else If ADDR in range 5000 to 6FFF then
      Store Sample
      Go to 1
```

Note the use of the store sample action. This means "store the most recently acquired sample in memory now". It does *not* mean, "From now on, start storing". It should be noted that since the store sample action is never executed unless ADDR is in the range 5000 to 6FFF, this branch essentially means "While in this sequence level, store only samples with ADDR between 5000 and 6FFF".

The above example seems to imply that only samples with ADDR between 5000 and 6FFF will be stored. However, this depends upon how the default storage has been set up. Using the previous example, if the default storage is set to "Store Everything", and a sample is outside of the range 5000 to 6FFF, then the Else If branch is not executed and the Default Storage is applied. In essence, the sequence level has said what to do when a sample has a value in a particular range, but it doesn't say what to do for samples outside the range. Therefore, if you want to specify the sequence level storage unambiguously, use the following:

```
1. If DATA = 005E then Trigger
   Else If ADDR in range 5000 to 6FFF then
     Store Sample
     Go to 1
   Else If ADDR not in range 5000 to 6FFF then
     Don't Store Sample
     Go to 1
```

Alternatively, if the default storage is set to "Store Everything", use the following:

```
1. If DATA = 005E then Trigger
   Else If ADDR not in range 5000 to 6FFF then
     Don't Store Sample
     Go to 1
```

In summary, Sequence Level Storage always overrides the Default Storage, but only for the conditions specifically mentioned in the Sequence Level Storage. You must be very careful that you account for the interaction between Default Storage and Sequence Level Storage.

Next: "Strategies for Setting Up Triggers" on page 251

Strategies for Setting Up Triggers

- "Trigger Functions" on page 251

- "Setting Up Complex Triggers" on page 254

- "Document Your Trigger Sequences" on page 254

Trigger Functions

While setting up logic analyzer triggers can be difficult, *trigger functions* can greatly simplify the process. Trigger functions are commonly-needed building blocks that can be combined to set up a trigger. Because the functions cover most common triggers, you can set up your trigger simply by selecting the appropriate function and filling in the data. The Agilent Technologies 16715A logic analyzer trigger user interface is shown in the following figure. Note that trigger functions are prominently located at the top of the screen.
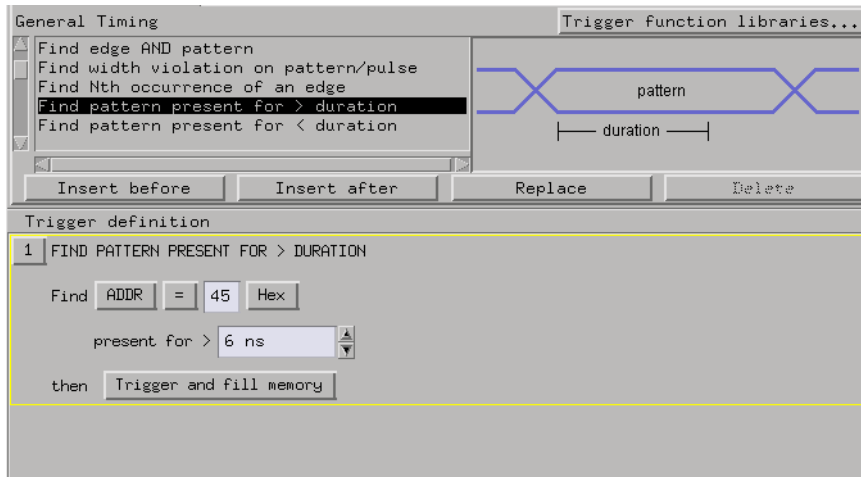
The Agilent 16715A trigger user interface

Note that a picture (which corresponds to the selected function) is provided to the right of the trigger function list.

For example, if you want to trigger when a bus pattern is immediately followed by another bus pattern, you can use the "Find Pattern2 occurring immediately after Pattern1" trigger function, shown in the following figure.
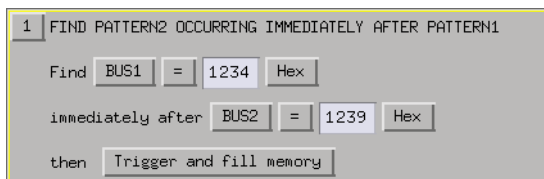
Pattern2 occurring immediately after Pattern1

Once you have selected this function, you simply fill in the names of the buses and the patterns. Contrast the previous figure with the following figure, which is the same trigger created using If/Then statements. The trigger function is easier to use because the additional details of the If/Then statements have been hidden. However, if you want to see the details, you can *break down* the function.

```
1  If  BUS2 = 1239  Hex
       occurs 1  time eventually
   then Goto  2
2  If  BUS1 = 1234  Hex
       occurs 1  time eventually
   then Trigger and fill memory
   Else if not  BUS2 = 1239  Hex  Or
                BUS1 = 1234  Hex
   then Goto  1
```

The same trigger as If/Then statements

Trigger functions can be modified. For example, if you start with the function "Find Edge", you can add another event, and it becomes the same as "Find Edge and Pattern". Therefore, a function that is not exactly correct can often be converted into the desired trigger. It is also possible to break down a function into the underlying If/Then statements and modify them.

The functions "Store range until pattern occurs" and "Store nothing until pattern occurs" make storage qualification much easier. These functions completely override the Default Storage. The "Store range until pattern occurs" function is shown in the following figure.

```
1 │ STORE RANGE UNTIL PATTERN OCCURS

   Store   BUS2    In range   5000   6FFF   Hex

   until   BUS1   =   0045   Hex  occurs

   then    Store sample

           Trigger and fill memory
```
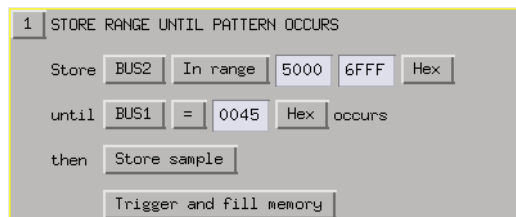
Store range until pattern occurs

Trigger functions are like building blocks because they can be used together in a trigger sequence. For example, if you want to set up a trigger as "Find edge followed by pattern", you can use a "Find Edge" function for Level 1 and a "Find Pattern" function for Sequence Level 2 (see the following figure). So, functions are useful both as an entire trigger sequence and as one step in a trigger sequence.
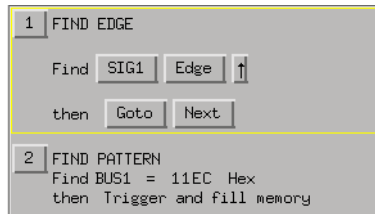
"Find Edge" and "Find Pattern" together

Setting Up Complex Triggers

Frequently, the most difficult part of setting up a complex trigger is breaking down the problem. In other words, how do you map a complex trigger into sequence levels, branches, and Boolean expressions? Here are step by step instructions:

1. Break down the problem into events that don't happen simultaneously. These correspond to the sequence levels.

2. Scan the list of trigger functions to try to find some that match the events identified in Step #1.

3. Within all remaining events, break them down into Boolean expressions and their corresponding actions. Each Boolean expression/Action pair corresponds to a separate branch within a sequence level. Remember that "Store" branches may exist that are used only to handle storage qualification for that sequence level.

Document Your Trigger Sequences

If a trigger sequence is important at one time, it is likely to be important again. This is why documenting trigger sequences is so valuable. Complex trigger sequences generally are too difficult to understand without some accompanying explanation. The following figure shows an example of the inline documentation on an Agilent Technologies logic analyzer. Inline means that the documentation is included in the trigger definition itself. This allows you to document

different parts of the trigger to describe how they work.
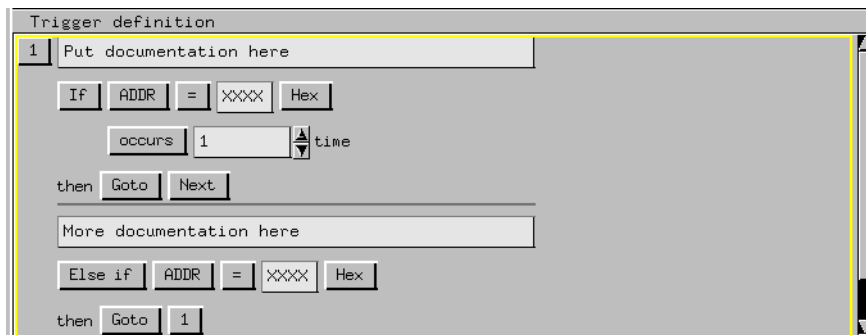
Inline documentation on an Agilent logic analyzer

Conclusions

Setting up logic analyzer triggers is very different than writing software. The job can be greatly simplified if other work can be leveraged by using pre-defined trigger functions and well-documented triggers that were written earlier. Only write your own trigger setup if there's nothing else available. Finally, when faced with a difficult trigger to set up, break down the problem into smaller chunks and deal with each one individually.

Understanding State Mode Sampling Positions

Synchronous sampling (state mode) logic analyzers are like edge-triggered flip-flops in that they require input logic signals to be stable for a period of time before the clock event (setup time) and after the clock event (hold time) in order to properly interpret the logic level. The combined setup and hold time is known as the setup/hold window.

A device under test (because of its own setup/hold requirements) specifies that data be valid on a bus for a certain length of time. This is known as the data valid window. The data valid window on most buses is generally less than half of the bus clock period.

To accurately capture data on a bus:

• The logic analyzer's setup/hold time must fit within the data valid window.
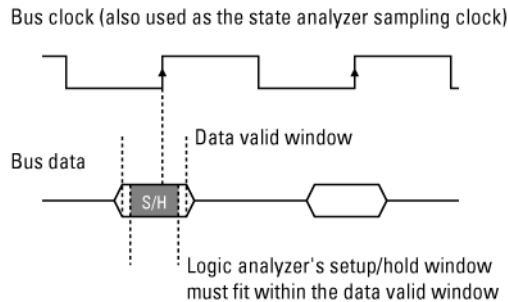
Bus clock (also used as the state analyzer sampling clock)

Data valid window

Bus data

S/H

Logic analyzer's setup/hold window
must fit within the data valid window

• Because the location of the data valid window relative to the bus clock is different for different types of buses, the position of the logic analyzer's setup/hold window must be adjustable (relative to the sampling clock, and with fine resolution) within the data valid window. For example:
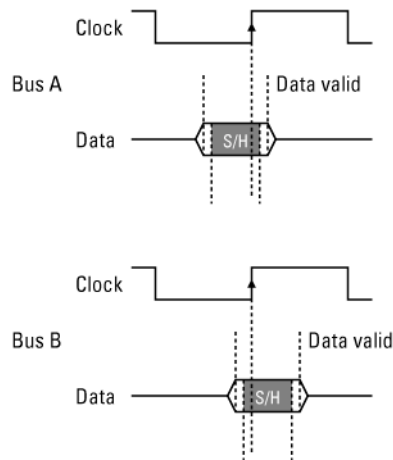
To position the setup/hold window (sampling position) within the data valid window, a logic analyzer has an adjustable delay on each sampling clock input (to position the setup/hold window for all the channels in a pod).

Sample Position Adjustments on Individual Channels

Some logic analyzers let you adjust the position of the setup/hold window (sampling position) on each channel. When you can make sampling position adjustments on individual channels, you can make the logic analyzer's setup/hold window smaller because you can correct for the delay effects caused by the probe cables and the logic analyzer's internal circuit board traces, and you are left with the setup/hold requirements of the logic analyzer's internal sampling circuitry.

However, the process of manually positioning the setup/hold window for each channel is time consuming. For each signal in the device under test and each logic analyzer channel, you must measure the data valid window in relation to the bus clock (with an oscilloscope), repeatedly position the setup/hold window and run measurements to see if the logic analyzer captures data correctly, and finally position the setup/hold window in between the positions where data was captured incorrectly.

In Agilent Technologies logic analyzers which have the *eye finder* feature, you can automatically position the setup/hold window on each

channel in a small fraction of the time (and without the extra test equipment) that it takes to make the adjustments manually. *Eye finder* is an easy way to get the smallest possible logic analyzer setup/hold window.

See Also   "To automatically adjust sampling positions" on page 49

"To manually adjust sampling positions" on page 52

"Selecting the State Mode (Synchronous Sampling)" on page 46

"Sampling Positions Dialog" on page 159

Understanding Eye Scan Measurements

Eye scan measurements are made possible by the logic analyzer's ability to double-sample each channel using slightly offset delays and by comparing the delayed samples using an exclusive-OR operation.
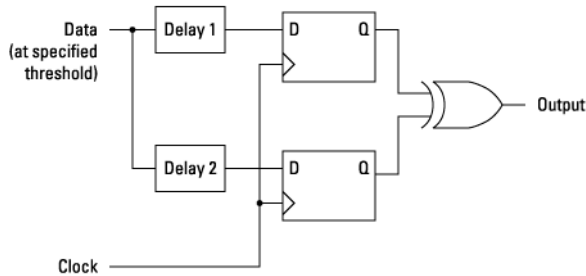
If output = 1, data transitioned between delay 1 and delay 2
If output = 0, data did not transition between delay 1 and delay 2.
Output is collected for many clocks.

When the exclusive-OR output is high, the delayed samples are different, and a transition is detected between the delay times.

Because of jitter and other variations in the sampled signal, an eye scan measurement checks many clocks for each pair of delay values so that it can report how often transitions occur between the two delay times.

Then, another pair of delay values is checked, and so on, until a whole range of time is scanned for transitions.
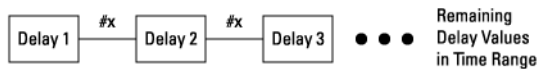
(#x = Number of Transitions)

Because the logic analyzer is able to adjust the threshold voltage for channels, an eye scan measurement is able to repeat the scan for transitions over time at many threshold voltage levels.
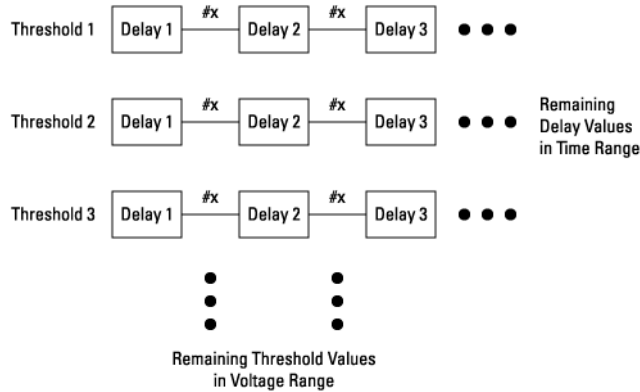
The result is a map of transitions detected in small windows of time and voltage over a range of time and voltage. Oscilloscope-like eye diagrams are used to display the measurement data. The number of transitions in each window is indicated by brightness or color.
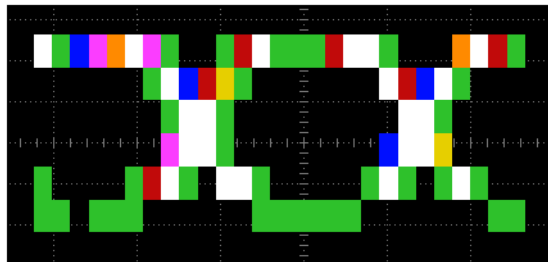
When setting up eye scan measurements, you can define the time and voltage ranges, as well as the size (in other words, resolution) of the time and voltage windows. There are coarse, medium, and fine settings, and you can make adjustments to these settings.

Coarse settings result in faster measurements because fewer samples are collected, but the eye diagram resolution is lower (see the picture above).

Coarse Settings
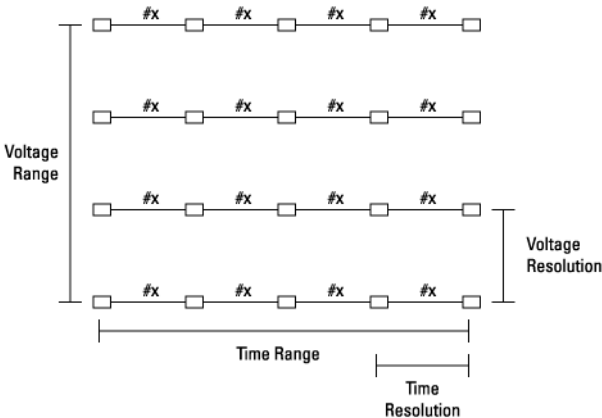
Fine settings result in higher eye diagram resolution, but because more samples are collected, measurements take longer to run.

Fine Settings
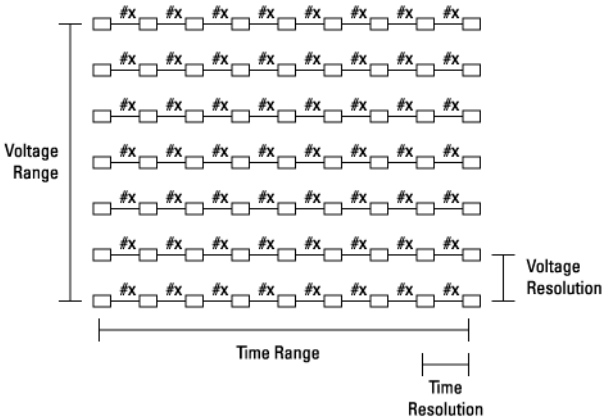
The picture below is an example of measurement results when the fine settings are used.
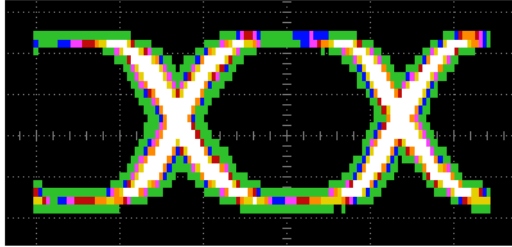
The smallest time resolution that can be set is 10 ps. The smallest voltage resolution that can be set is 1 mV.

The number of channels on which an eye scan measurement collects data also affects the measurement time. The exception is when there are multiple logic analyzer cards in a module; in this case, measurements run simultaneously in parallel.

See Also   "Selecting the Eye Scan Mode" on page 55

"Using the Logic Analyzer in Eye Scan Mode" on page 117

"Eye Scan Mode" on page 172

"The Eye Scan Tab" on page 204

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 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

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 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 or higher 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

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 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 would be referred to as *Slot C:*

Glossary

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.

Index

Index

Index

Index

Index

Index

Index

Index

Index

Index

Agilent Technologies