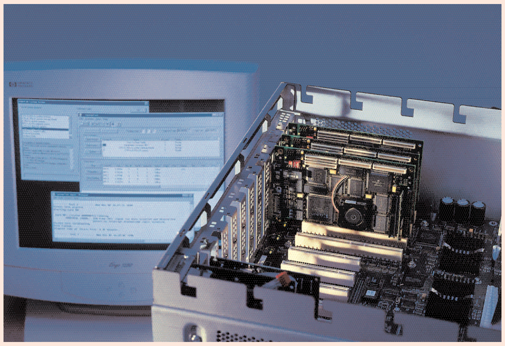
5

5 Measurements of Agilent E2920 PCI/PCI-X Series



PCI
LOCAL BUS

PCI-X

Agilent Technologies

5 Measurements...
... that you need for the verification of your device

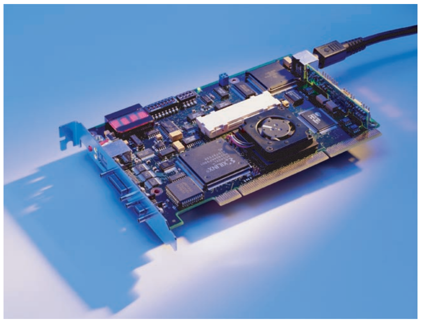
Introduction

Whether you are a project manager or a design engineer developing peripheral chips, plug-in cards, systems or firmware, you need debugging tools that help to evaluate, optimize and validate the behavior of your prototype under its final operating conditions as early and as completely as possible.

The Agilent E2920 Verification Tools, PCI and PCI-X Series, your "window into the system" during product development, give you access to almost all of the system components located on the PCI or PCI-X bus, as well as devices and adapters on secondary buses such as ISA.
The E2920 PCI and PCI-X Series, offers you a convenient, cost effective and plug-in solution, which, being a testcard itself, requires almost no time for setting up.

This brochure shows how to set up five major measurements used for verifying devices with the Agilent E2920 Verification Tools. You can use it to learn how to use the tools, as you need them.

Content



We help you to solve problems successfully

Agilent, one of the leading producers of measurement equipment, do everything to help you trouble-shoot your products. We do this for example with the practical 5 measurements described in this brochure. With these measurements, we try to give you an overview about the capabilities of our test cards. The input for these measurements comes from engineers with great experience in developing PCI and PCI-X. We tried to develop this brochure in an easy and understandable way.

Technical Data

The E2920 PCI/PCI-X series solutions consists of

- a test card available for most important PCI and PCI-X technologies (32/64 bit, 33/66 MHz) and form factors (standard PCI and CompactPCI) and for PCI-X (32/64 bit, 133 MHz),
- software tool providing sophisticated user interfaces for each phase of the design cycle

All tools are programmable via C-API or graphical user interface. The E2920 series provides standard solutions as required by each phase of the design cycle process and allows customized configurations for equipping large labs.

1 Check for PCI and PCI-X protocol rule violations

Protocol rule violations do not always crash a computer system. If protocol rule violations are caused by several devices, however, the system is likely to crash. To exclude this possibility, you should take steps to locate all protocol rule violations that occur on your system.

The Agilent Protocol Observer, in combination with the Agilent Analyzer, simplifies the task of detecting protocol rule violations; this is not easy- you have to be able to spot them in a data trace..

Aims of this Application Note

- To show you how to obtain a quick overview of all protocol rule violations.

- To provide a simple way of focusing on the protocol rule violations of interest.

Questions that can be answered with the help of the PCI/PCI-X Protocol Observer

- Have any protocol rule violations occurred?

- What is a trace that contains a rule violation telling you?

- Which of the devices on the bus are responsible for protocol rule violations?

Benefits of the PCI/PCI-X Protocol Observer

- At-a-glance information on protocol rule violations in your system.

- Triggering on protocol errors to enable you to figure out the causes of errors.

This application note is based around an example that shows you how the Agilent testcard can be used to detect a protocol violation that occurs during an access to the I/O space of a PCI graphics card.

Setting Up the Test

Introduction

The protocol observer hardware implemented on the Agilent testcard simultaneously checks 53 different protocol rules against the PCI/PCI-X Specification. If a protocol rule is violated, the protocol observer generates a signal (the berr signal for PCI and the proterr for PCI-X) that can be used to trigger the Analyzer, so that it can capture the bus traffic around the violation.

To check the protocol, you have to perform the following steps:

1. Run the Protocol Observer.

2. View the violated rules.

3. Disable all rules that you are not interested in.

4. Trigger on particular PCI and PCI-X protocol errors and run the Analyzer again.

5. Display the captured wave forms.

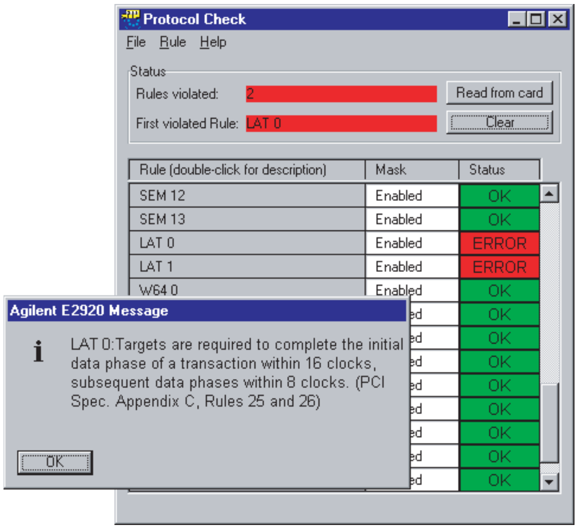
Running the Protocol Observer

When the testcard runs, the Protocol Observer continuously monitors the PCI/PCI-X bus. It operates in conjunction with the

Viewing Violated Rules

The Protocol Check window lets you view protocol rule violations that have been detected by the Protocol Observer.

- To open the Protocol Check window, select Protocol Check in the Analyzer menu.
  The Protocol Check window lists a total of 53 protocol rules. Any protocol rule violations are flagged up in the Status column as ERROR. You can scroll through the list to locate the errors.
  The total number of violated rules and the name of the first violated rule are shown in the Status field.
- To reset the list (clear the errors), click the Clear button.
- To update the list, click the Read from card button. This also updates the Status field.



Status

Rules violated: 2

First violated Rule: LAT 0

Rule (double-click for description)	Mask	Status
SEM 12	Enabled	OK
SEM 13	Enabled	OK
LAT 0	Enabled	ERROR
LAT 1	Enabled	ERROR
W64 0	Enabled	OK
	ed	OK
	ed	OK
	ed	OK
	ed	OK
	ed	OK
	ed	OK
	ed	OK
	ed	OK

Agilent E2920 Message

LAT 0:Targets are required to complete the initial data phase of a transaction within 16 clocks, subsequent data phases within 8 clocks. (PCI Spec. Appendix C, Rules 25 and 26)

Figure 1.

- Double-click any rule name field to get an explanation of what that rule checks. The message box in the above figure 1 shows the rule for LAT 0 (latency 0).

Focussing on Individual Errors

You can individually enable or disable each rule in the Protocol Check window. This prevents known errors from being flagged. The feature is very useful if
several protocol rule violations have been detected and you want to examine just one of them in detail. The Analyzer display modes (Waveform Viewer, Bus Cycle Lister and Transaction Lister) display the bus traffic pertaining to this violated rule.

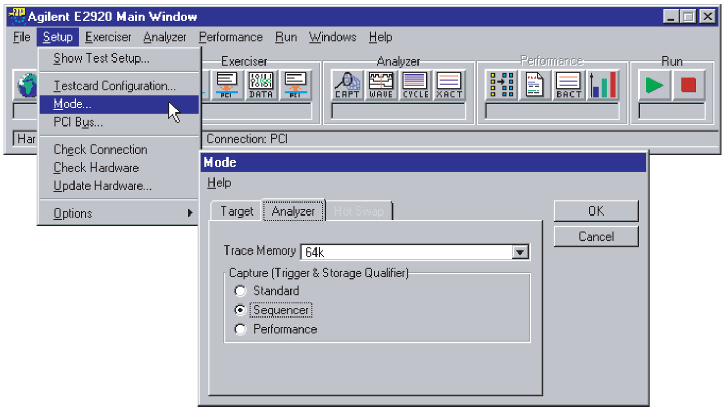
Programming the Protocol Observer to Trigger on Protocol Errors

In order to find out which device in your system caused a
violation, you have to program the Protocol Observer to trigger the Analyzer when a protocol error occurred. Setting up the trigger involves a procedure that varies for PCI systems and for PCI-X systems. These are described separately below.

Trigger Setup for the PCI Bus (valid only with Agilent E2920 PCI Series)

The PCI Protocol Observer asserts an output signal (called berr) when any of the enabled protocol rules have been violated. To use this signal to trigger the built-in logic Analyzer, you need to define it as a pattern term in the Analyzer's Capture window.

The trigger is set up as follows:
1. Ensure that Standard mode is selected.



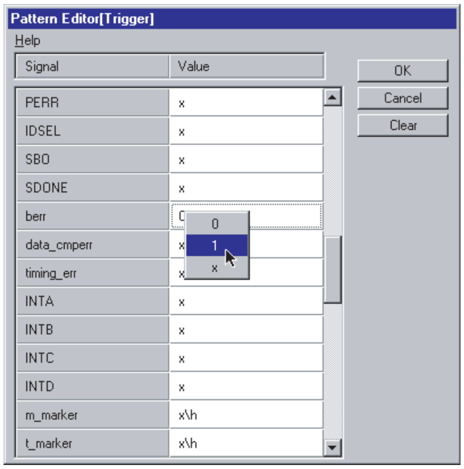
Connection: PCI

Trace Memory 64k

Capture (Trigger & Storage Qualifier)
○ Standard
● Sequencer
○ Performance

Figure 2

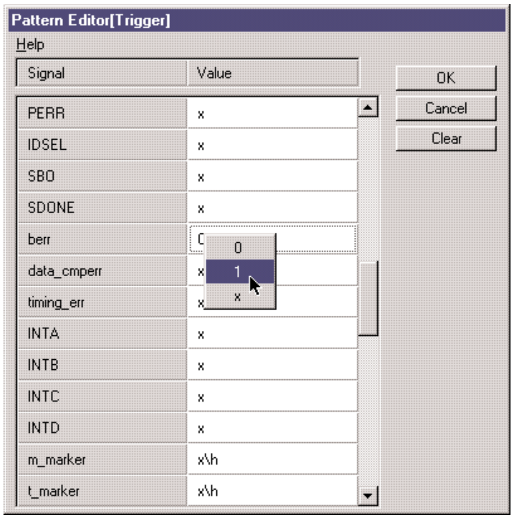
2. Open the Capture window by selecting Capture in the Analyzer menu.



Signal	Value
PERR	x
IDSEL	x
SBO	x
SDONE	x
berr	C
data_cmperr	x
timing_err	x
INTA	x
INTB	x
INTC	x
INTD	x
m_marker	x\h
t_marker	x\h

Figure 2a

3. Select Pattern, and press the Edit button next to the pat tern term.This opens the Pattern Editor window.



Signal	Value
PERR	x
IDSEL	x
SBO	x
SDONE	x
berr	C
data_cmperr	x
timing_err	x
INTA	x
INTB	x
INTC	x
INTD	x
m_marker	x\h
t_marker	x\h

Figure 3.

4. In the Pattern Editor window, press the Clear button to reset all pattern terms to "x" (= don't care), then set the berr field to 1.

5. Run the Analyzer by selecting Run in the Analyzer menu. The Analyzer now triggers only on the selected protocol errors.

Trigger Setup for the PCI-X Bus (valid only with Agilent E2920 PCI-X Series)

Set up the trigger as follows:

1. Open the Capture dialog box by selecting Capture from the Analyzer menu (or click the Capture button in the toolbar).

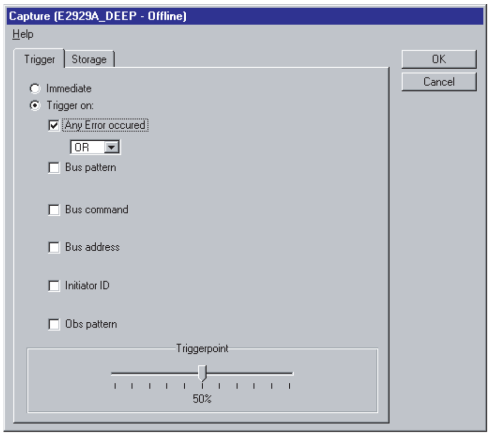
2. In the Capture dialog box, select the Trigger tab.

3. Choose the Trigger on: option.

4. Select the check box Any Error occurred.

5. Click OK.

The Capture window should now look like this.



Capture (E2929A_DEEP - Offline)

○ Immediate
● Trigger on:
☑ Any Error occured
OR
□ Bus pattern
□ Bus command
□ Bus address
□ Initiator ID
□ Obs pattern

Triggerpoint: 50%

Figure 4.

6. Press OK in the Capture window; the Analyzer is now ready to run.

7. Run the Analyzer by selecting Run in the Analyzer menu. The Analyzer now triggers on protocol errors.

Displaying the Captured Waveforms

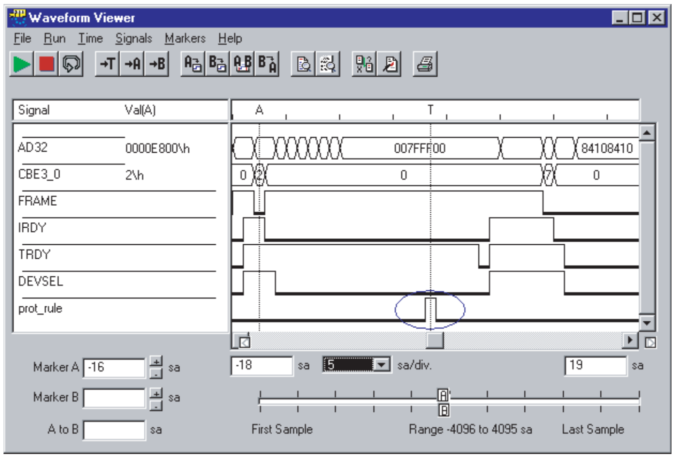
The display procedure is the same for PCI and PCI-X systems. Here, the waveforms produced when accessing a PCI device's I/O space are used as an example.

When the Analyzer runs and the device under test is accessed, the Analyzer triggers on the occurrence of the first violation. To view the results, proceed as follows:

1. Select Waveform Lister ... from the Analyzer menu to open the Waveform Viewer.
   Your display will of course look different to that shown below. Note that you can add and remove signals from the display or rearrange their display order using the Arrange item from the Signals menu.

Note: For PCI the berr signal is labelled prot_rule in the Waveform Viewer and is asserted at the trigger point. For PCI-X the signal is labelled proterr in the waveform viewer and is asserted at the trigger point.

2. Identify the Device that Causes the Protocol Rule Violation
The error in this particular case has been caused by the target with physical address 0000E800\h. This target did not respond to the I/O Read with the first word of data within 16 clocks, as specified for LAT 0.



Marker A -16 sa

Marker B sa

A to B sa

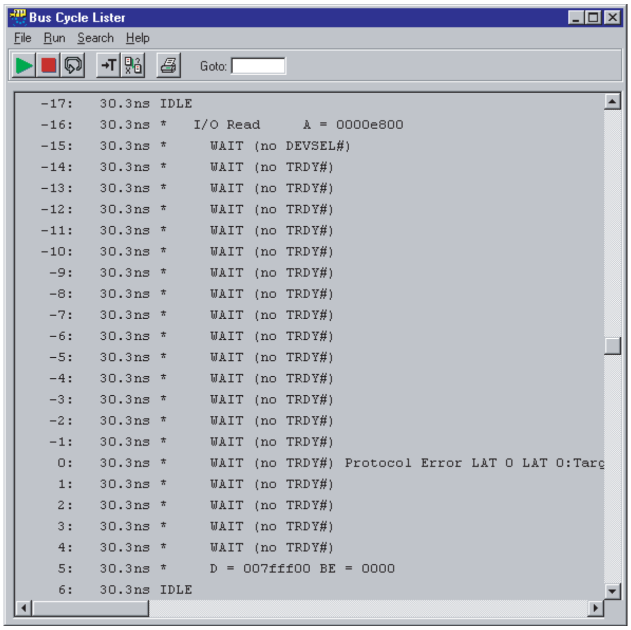
-18 sa 5 sa/div. 19 sa

Figure 5.

Note for identifying a master device that causes an error:

If a master device causes a protocol rule violation, for example, a LAT1 error, identification of this master is more complicated.
The testcard can only identify masters that are correctly connected to it. The GNT# and REQ# lines of the masters must be connected to the testcard's external trigger input pins-if necessary, by soldering wires.
To identify the master that causes the error, in the Waveform Viewer, check the levels of the external trigger input pins connected to the masters (trigger0, trigger1, ...) around the trigger point.

3. Select Bus Cycle Lister in the Analyzer menu. The lister shows the associated transaction, along with the protocol error message.



```
 -17:   30.3ns IDLE
 -16:   30.3ns *   I/O Read     A = 0000e800
 -15:   30.3ns *     WAIT (no DEVSEL#)
 -14:   30.3ns *     WAIT (no TRDY#)
 -13:   30.3ns *     WAIT (no TRDY#)
 -12:   30.3ns *     WAIT (no TRDY#)
 -11:   30.3ns *     WAIT (no TRDY#)
 -10:   30.3ns *     WAIT (no TRDY#)
  -9:   30.3ns *     WAIT (no TRDY#)
  -8:   30.3ns *     WAIT (no TRDY#)
  -7:   30.3ns *     WAIT (no TRDY#)
  -6:   30.3ns *     WAIT (no TRDY#)
  -5:   30.3ns *     WAIT (no TRDY#)
  -4:   30.3ns *     WAIT (no TRDY#)
  -3:   30.3ns *     WAIT (no TRDY#)
  -2:   30.3ns *     WAIT (no TRDY#)
  -1:   30.3ns *     WAIT (no TRDY#)
   0:   30.3ns *     WAIT (no TRDY#) Protocol Error LAT 0 LAT 0:Targ
   1:   30.3ns *     WAIT (no TRDY#)
   2:   30.3ns *     WAIT (no TRDY#)
   3:   30.3ns *     WAIT (no TRDY#)
   4:   30.3ns *     WAIT (no TRDY#)
   5:   30.3ns *     D = 007fff00 BE = 0000
   6:   30.3ns IDLE
```

Figure 6

2 Verify if a Device can stand all kinds of protocol variations

Validating servers and workstations that contain various I/O systems, various peripherals and high and low-speed devices, has become a sophisticated task. If you are a validation engineer, you have to ensure that server products can stand all possible protocol variations, that corner cases are covered and that the testing stays within reasonable limits.
Agilent Techologies' solution is a combination of testcards and software running on the system under test, the System Validation Package (SVP).

Aims of this Application Note

- To show how to set up and control several Agilent test cards to simulate PCI or PCI-X application-realistic system traffic.
- To show how to analyze occurred protocol rule violations or data-compare errors.

Questions that can be answered with the help of the SVP

- How can protocol variations be set?
- Can my system stand the protocol variations I set?
- Can my system transfer data in the correct way?
- How can I analyze occurred protocol violation or data-compare errors?

Benefits of the Agilent Product

- Provides the easiest way of defining corner cases when designing systems, bridges and add-on cards.
- Uses the Protocol Permutator and Randomizer (PPR) technology, which ensures permutation of PCI or PCI-X protocol parameters and data traffic in a pseudo-random way.
- Provides measurable test coverage and test repeatability.
- Provides short testing time.

Setting Up the Test

Introduction

To make the test setup easier, the SVP provides several ready-to-run tests to put particular data paths in I/O systems under stress in a controlled and predictable way.
In this application note, the operating principles of the SVP are introduced by checking the availability/readability of the system memory.
To perform this, the ready-to-run System Memory Read test is used. The System Memory Read test requires one PCI/PCI-X test-card plugged into the PCI/PCI-X bus connected to the host bridge.

Task

A host bridge is to be tested by reading from the system memory. The memory read should be performed with different burstlengths (2, 3, 8 and 11).

Errors (protocol violations and data compare errors) that occur during the test should be by using the Analyzer functionality of the testcards.

Steps required to fulfill this Task

The test for stressing a bridge is performed as follows:

1. Set up the test configuration with the help of the SVP GUI. That means, select the ready-to-run test, select the test cards for this test and set the test parameters (for example, address space and test duration).
2. Set protocol variations (different burstlengths).
3. Run the test.
4. Analyze the test results with the help of the Analyzer functionality of the testcard.

These steps are described in the following sections.

Set Up the Test Configuration

1. Plug the testcard in the bus that is connected to the host bridge, as shown in figure 1 below.

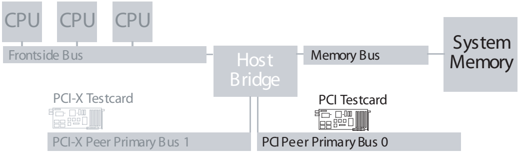
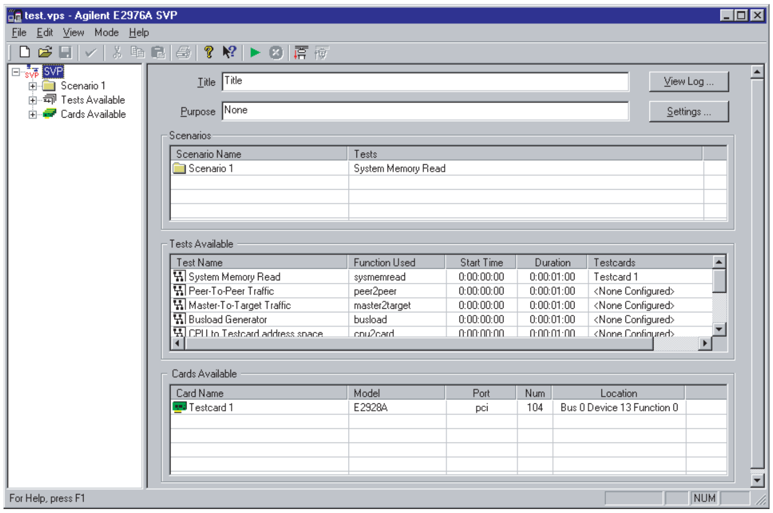


Figure 1

2. Run the SVP software.
The software automatically scans all connected PCI and PCI-X busses and other control interfaces (RS-232, USB, Fast Host Interface) for Agilent test cards and initializes them.
The main window of the SVP GUI now looks as follows (fig 2):



Title: Title

Purpose: None

Scenarios

Scenario Name	Tests
Scenario 1	System Memory Read

Tests Available

Test Name	Function Used	Start Time	Duration	Testcards
System Memory Read	sysmemread	0:00:00:00	0:00:01:00	Testcard 1
Peer-To-Peer Traffic	peer2peer	0:00:00:00	0:00:01:00	<None Configured>
Master-To-Target Traffic	master2target	0:00:00:00	0:00:01:00	<None Configured>
Busload Generator	busload	0:00:00:00	0:00:01:00	<None Configured>
CPU to Testcard address space	cpu2card	0:00:00:00	0:00:01:00	<None Configured>

Cards Available

Card Name	Model	Port	Num	Location
Testcard 1	E2928A	pci	104	Bus 0 Device 13 Function 0

Figure 2

The SVP software provides three set-up levels; scenarios, tests and testcards:

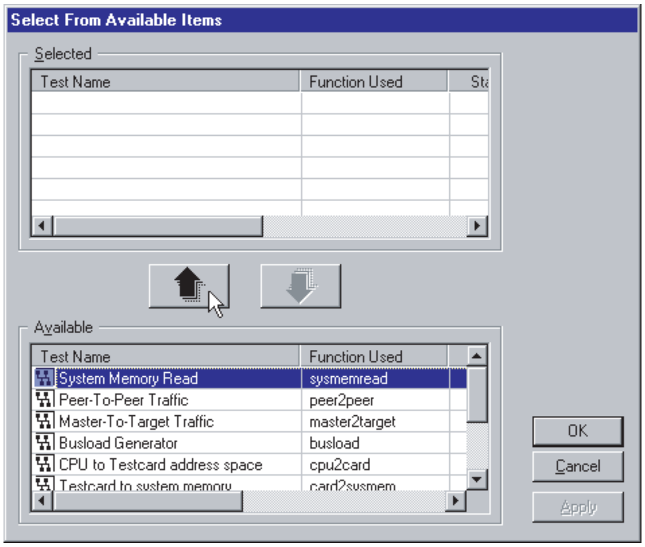
- One **scenario** can combine several ready-to-run tests to run in parallel. Several scenarios are executed successively.
- The ready-to-run **tests** are specified by the testcards they use (must be selected by the user), the address space, test duration, bytes to transfer and the bandwidth with which the test tries to occupy the bus.
- The **testcards** are specified by their location in the bus system, their features that can be enabled or disabled and their protocol variation settings.

How to set up the scenario, the test and the testcards necessary for this task is described in the following:

To select the System Memory Read test:

3. Select Scenario 1 in the SVP navigator and click the Select Test(s) button in the Scenario Details window.
This opens the Select from Available Items dialog box.

4. Move the System Memory Read test from the Available list to the Selected list and click OK.



Selected

Test Name	Function Used	Sta

Available

Test Name	Function Used
System Memory Read	sysmemread
Peer-To-Peer Traffic	peer2peer
Master-To-Target Traffic	master2target
Busload Generator	busload
CPU to Testcard address space	cpu2card
Testcard to system memory	card2sysmem

Figure 3.

To select the testcard for the System Memory Read test:

5. Select the System Memory Read item in the navigator and click the Select Card(s) button in the Test Setup window.
This opens the Select from Available Items dialog box.

6. Move the respective testcard into the Selected list.

In the Test Setup window, the default values for the test parameters are used.

Setting Testcard Properties

The Agilent SVP software specifies testcard properties for all available testcards by default. The settings are displayed in the Testcard Set Up window.

To carry out the task mentioned above, the testcards must be set up as follows:

- The Analyzer feature of each testcard must be enabled to allow analyzing of errors occurred.
- The master of each testcard transfers data with burst lengths of 2, 3, 8 and 11.

To enable the Analyzer feature:

1. Open the Testcard Set Up window by clicking the respective testcard in the SVP navigator.

2. Check the Use Analyzer option for each testcard.
   Now, the Analyzer triggers as soon as an error (data-compare error or protocol violation) occurs.

The captured data is stored in the testcard's trace memory and can later be uploaded for analyzing.
See section *Analyzing the Results*.

The testcard properties shown in the Testcard Set Up window include the master and target settings for the testcard.
These settings (displayed when you click the Details buttons next to the Use Master and Use Target options) are the properties permutated by the PPR technology.

By default, all memory accesses are varied through all possible combinations of their attributes. This applies for varying block sizes and the use of the different memory commands, such as write, read, write invalidate, read line, and read multiple. Furthermore, permutations are made in terms of the alignments and byte enables. This means that all variations of byte, word, and dword accesses are used.

In this test example, you have to modify these settings to ensure the use of various burst lengths.

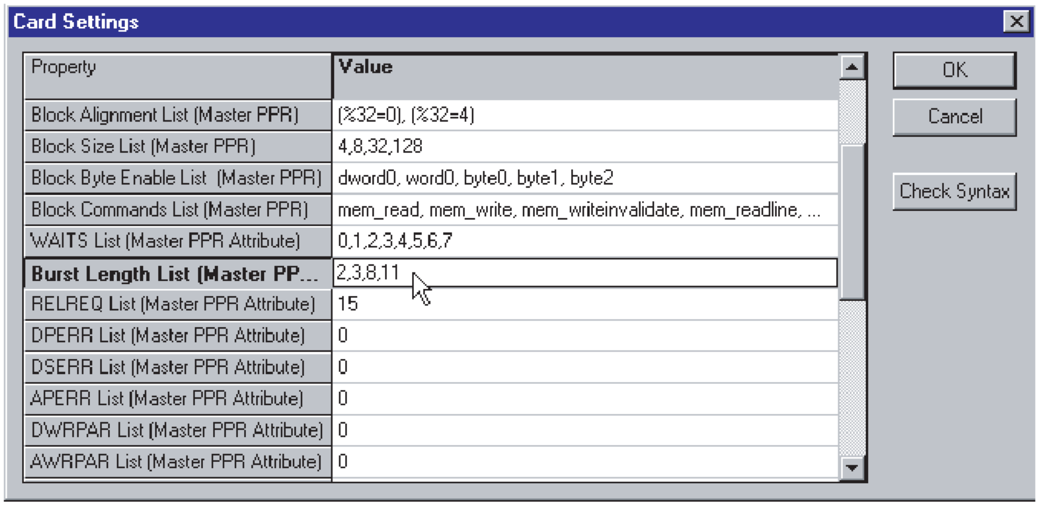
To set burst length variations:

1. In the Testcard Set Up window for each testcard, click the Details button next to the Use Master option.
   This opens the Card Settings dialog box.

2. Enter the values 2, 3, 8 and 11 for the property Burst Length List (Master PPR attribute) and click OK, Note:

The Check Syntax button is used to check if the testcard is able to store the defined variations.

Now the test is ready to run.



Card Settings

Property	Value
Block Alignment List (Master PPR)	(%32=0), (%32=4)
Block Size List (Master PPR)	4,8,32,128
Block Byte Enable List (Master PPR)	dword0, word0, byte0, byte1, byte2
Block Commands List (Master PPR)	mem_read, mem_write, mem_writeinvalidate, mem_readline, ...
WAITS List (Master PPR Attribute)	0,1,2,3,4,5,6,7
Burst Length List (Master PP...	2,3,8,11
RELREQ List (Master PPR Attribute)	15
DPERR List (Master PPR Attribute)	0
DSERR List (Master PPR Attribute)	0
APERR List (Master PPR Attribute)	0
DWRPAR List (Master PPR Attribute)	0
AWRPAR List (Master PPR Attribute)	0

Figure 4.

Running the Test

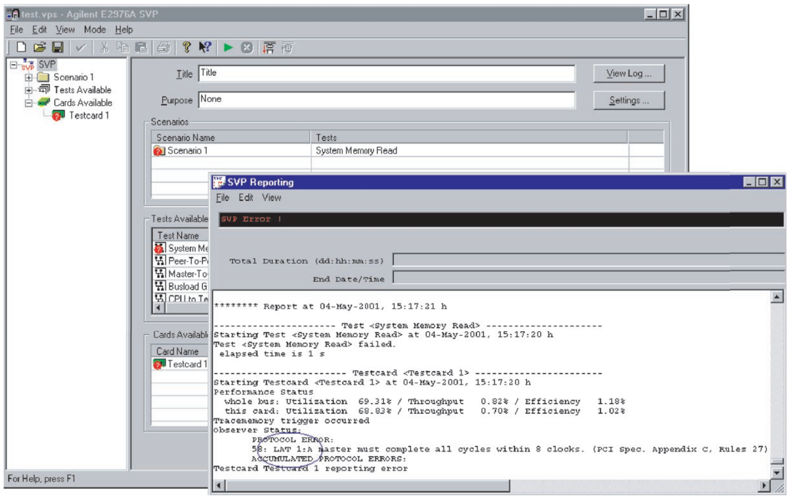
To run the test:

1. Ensure that you are in online mode. If not, click Go Online in the Mode menu.

2. Click the run icon ▶ in the tool bar.

The software runs the test and the results are shown in the SVP Reporting dialog box, which is opened automatically.

The test report informs, among other things, about the testcard status. That means, it shows all errors occurred.



Title: Title
Purpose: None

Scenario Name	Tests
Scenario 1	System Memory Read

```
SVP Error !

Total Duration (dd:hh:mm:ss)
      End Date/Time

******** Report at 04-May-2001, 15:17:21 h

--------------------- Test <System Memory Read> ---------------------
Starting Test <System Memory Read> at 04-May-2001, 15:17:20 h
Test <System Memory Read> failed.
 elapsed time is 1 s

----------------------- Testcard <Testcard 1> -----------------------
Starting Testcard <Testcard 1> at 04-May-2001, 15:17:20 h
Performance Status
  whole bus: Utilization  69.31% / Throughput   0.82% / Efficiency   1.18%
  this card: Utilization  68.83% / Throughput   0.70% / Efficiency   1.02%
Tracememory trigger occurred
Observer Status:
       PROTOCOL ERROR:
       58: LAT 1:A master must complete all cycles within 8 clocks. (PCI Spec. Appendix C, Rules 27)
       ACCUMULATED PROTOCOL ERRORS:
Testcard Testcard 1 reporting error
```

Figure 5.

In this test example, the report is interrupted as soon as an error (data-compare error or protocol violation) occurs.
To define this:

1. Click the SVP item in the navigator and select the Settings... button.
This opens the SVP Test Settings dialog box.

2. In the SVP Test Settings dialog box, select Stop On Error.

The error is displayed in the SVP test report and can now be analyzed with the Analyzer software of the testcard.

Analyzing the Results

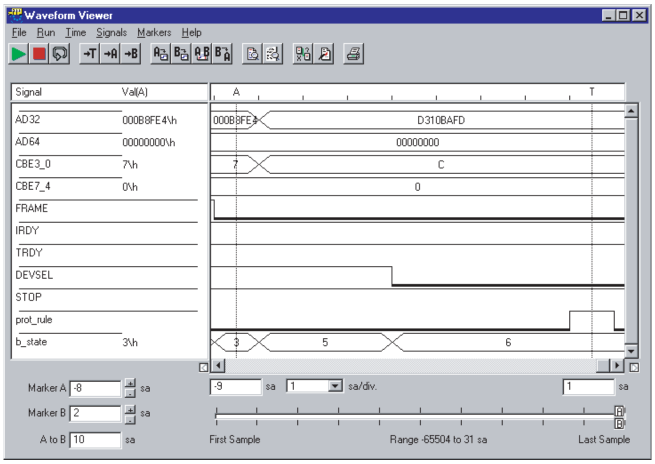
As previously defined, the testcard's Analyzer triggers as soon as an error occurs.
The captured data stored in the testcard's trace memory can now be uploaded to the Analyzer GUI for analysis.
To help you analyze the captured data, the Analyzer GUI provides several display modes. In this test example, the Waveform Viewer is used to analyze the data.

To analyze the errors with the Analyzer software, some preparations are necessary:

1. Switch the SVP software to offline mode by selecting Go Offline in the Mode menu.

2. Start the E2920 PCI Analyzer software or PCI-X Analyzer software respectively.

3. Open the Waveform Viewer by clicking the WAVE in the Analyzer toolbar.



Signal	Val(A)
AD32	000B8FE4\h
AD64	00000000\h
CBE3_0	7\h
CBE7_4	0\h
FRAME	
IRDY	
TRDY	
DEVSEL	
STOP	
prot_rule	
b_state	3\h

Marker A -8 sa
Marker B 2 sa
A to B 10 sa
-9 sa 1 sa/div. 1 sa

Figure 6.

Click the go to Trigger →T button to display data around the trigger point. You can now analyze the transaction that causes the protocol error.

You can also use the Bus Cycle Lister, which provides a more compact view of the captured PCI transactions.

3

Analyze how efficiently a device uses PCI/PCI-X resources

The Agilent PCI/PCI-X Performance Optimizer, a combination of testcard and software, provides help regarding performance analysis of extensive systems or single devices.
If you perform benchmark tests, the Agilent PCI/PCI-X Performance Optimizer helps you to quickly evaluate the performance of your system.
Predefined measurements such as efficiency, throughput and utilization lead to quick results.
If you are a device designer, the Agilents PCI/PCI-X Performance Optimizer helps you not only to evaluate the performance of a particular device, but also shows you where you can improve it.

Aims of this Application Note

- To test the overall system performance.
- To analyze and optimize the performance of a particular device pair.

Questions that can be answered with the help of the SVP

- Real-time performance measurements for benchmarking:
  - How efficiently is the system operating?

- Post-processed analysis:
  - How efficiently is the device operating?
  - How can I optimize performance?

Benefits of the Agilent Product

- Real-time performance:
  - Pre-defined measurements are available.
- Post-processed performance measurements:
  - Detailed measurements are provided.
  - Parallel measurements are provided.
  - Performance analysis is possible.

Viewing the Overall System Performance for PCI

The Agilent E2920 PCI software can calculate and display two performance measures simultaneously. Each measure can either show one of the predefined measurements (efficiency, throughput, utilization and retries), or can be programmed manually for advanced performance measurements.

Note:
The software provides eight performance measurements-the GUI provides direct access to two of them. The others can be accessed through the C-API.

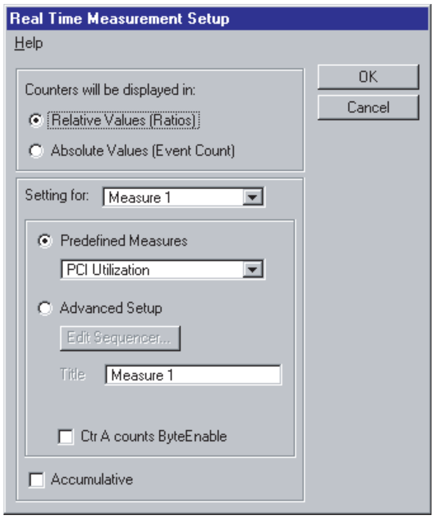
Setting Up a Predefined Performance Measurement for PCI

Task

Set up the software so that the performance measurements PCI Utilization and PCI Efficiency are displayed.

Procedure

1. Open the Real Time Measurement Setup dialog box by selecting Real Time Counter Setup... in the Performance menu.



Real Time Measurement Setup

Help

Counters will be displayed in:

● Relative Values (Ratios)

○ Absolute Values (Event Count)

Setting for: Measure 1

● Predefined Measures

PCI Utilization

○ Advanced Setup

Title Measure 1

□ Ctr A counts ByteEnable

□ Accumulative

Figure 1.

2. Use the default settings for Measure 1 to measure the PCI Utilization.

3. For Measure 2, from Predefined Measures, select PCI Efficiency.

Viewing the Results

1. Open the Real Time Counter Result window by selecting Real Time Counter Results... in the Performance menu.

2. Click Start.
   The results are displayed in Momentary view, which shows the results of the last measurement time interval.
   To view the result history over time, select the Time History tab.

Advanced Performance Measurement for PCI

For each advanced performance measurement, two programmable counters (A and B) can be set up to count particular events on the bus, such as idle states, retries, and so forth.
To specify the events, pattern terms are to be defined. The counters A and B are controlled by sequencers; you have to define these in the sequencer description table.
Additionally, there is a reference counter for counting bus cycles.

For details about programming the counters, please refer to the Agilent PCI Analyzer User's Guide.

Viewing the Overall System Performance for PCI-X

The Agilent E2920 PCI-X Series software provides two different user interfaces for real-time performance and post-processed measurements.
To access the user interface for real-time performance analysis, select:
Start > Programs > Agilent E2920 PCI-X > PCI-X Real Time Performance GUI

The PCI-X Real-Time Performance software can calculate and display two performance measurements simultaneously. The software provides standardized predefined measurements to be run either for the card the system, or both. Each measurement can either show the predefined measurements efficiency, throughput and utilization, or the retry rate or the split rate.

Setting Up a Real-Time Performance Measurement for PCI-X

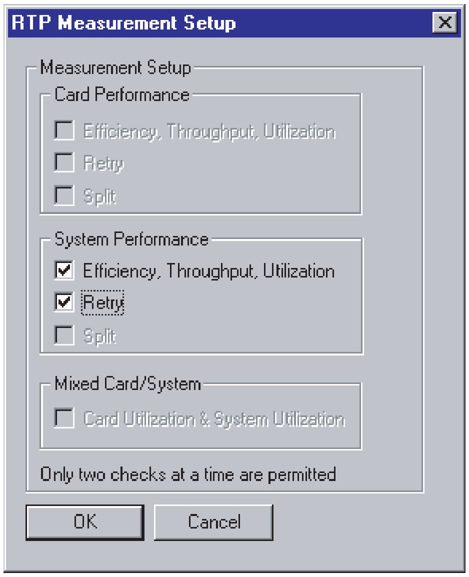
Task

Set up the performance measurement so that utilization, efficiency, throughput and split rate for the whole system are displayed.

Procedure

1. Open the RTP Measurement Setup window by selecting Setup Measurement... in the Setup menu.
2. For System Performance, select only the following:
- Efficiency, Throughput and Utilization
- Retry
Note that you may have to clear any other selected measurements first

The RTP Measurement Setup window looks as follows:


RTP Measurement Setup

Measurement Setup

Card Performance
☐ Efficiency, Throughput, Utilization
☐ Retry
☐ Split

System Performance
☑ Efficiency, Throughput, Utilization
☑ Retry
☐ Split

Mixed Card/System
☐ Card Utilization & System Utilization

Only two checks at a time are permitted

Figure 2.

Viewing the Results

Click Run ▶ in the PCI-X Real Time Performance window. The results are displayed by default as Bar Graphs, which show the results of the last measurement time interval.

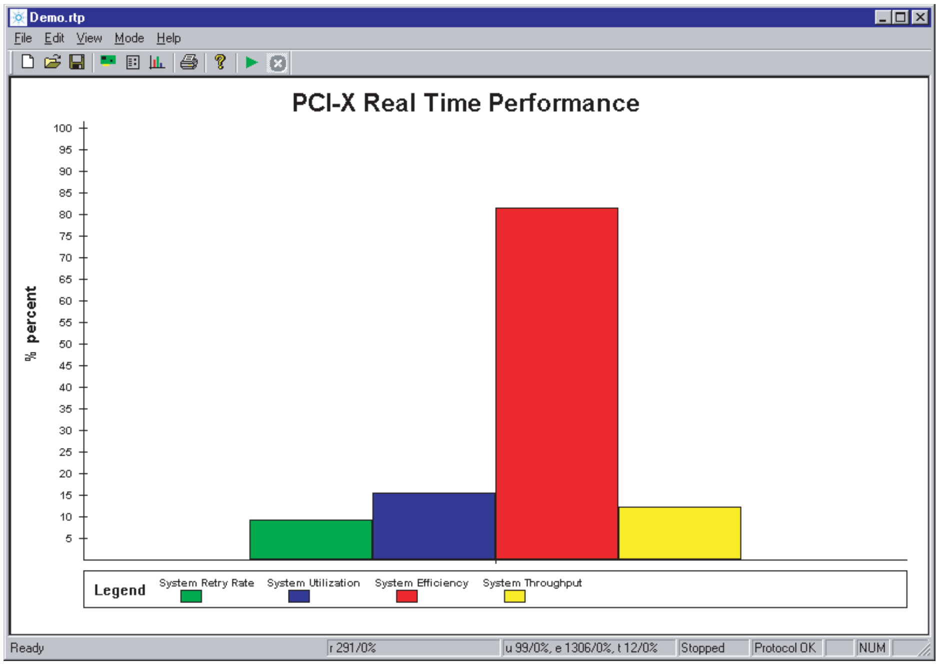


Figure 3.

You can change the display and measurement options in the RTP Options dialog box, which you open by selecting Options in the View menu. This dialog box allows you to modify the measurement update interval, to create a report file and to configure the display of the results.

Post-processed Performance Analysis Test

You can use the post-processed performance feature to analyze the performance of a particular PCI/PCI-X device. The results helps you to optimize your device.

The following is required to set up a post-processed performance analysis test:

1. Enable post-processed performance analysis (valid only with the Agilent E2920 PCI software).
   The software has to run in performance capture mode.

2. Identify the device(s) under test.
   To enable the Agilent PCI/PCI-X Performance Optimizer to identify particular devices communicating on the PCI/PCI-X bus, you need to specify them in the setup. This differs for the device under test, as described below:

Master Identification

The testcard can only identify masters that are correctly connected to it. The GNT# of the master must be connected to one of the testcard's external trigger input pins-if necessary,(PCI only) by soldering a wire. If you also want to run latency tests, the REQ# lines of the respective masters need to be connected to the testcard's external trigger input pins as well.

Target/Completer Identification

Target and completer devices are specified by their address space.

Requester Identification

To identify the requester, you need the bus, device, and function number of this device. To determine these numbers, run the performance measurement and view the performance report (available from the Performance menu). The software does a configuration scan on the whole PCI-X bus system. Subsection 1.1 in this report lists all requesters found on the bus with their corresponding bus, device and function numbers.

3. Set up the data capture.
   A trigger has to be set up for a particular event on the bus. For example, the PCI/PCI-X Analyzer can be set up to trigger when the device under test is accessed. The software then sets up the Analyzer so that only data important for performance calculations is captured in the trace memory. Thus, the trace memory is used very efficiently.

4. Select the device pair.
   The Performance Optimizer allows you to focus the performance test on a single device or a single master/target pair, or requester/completer pair respectively.

5. Start the measurement, and view and analyze the results.

Note:

Because the test setups for PCI and PCI-X devices are very similar, the differences are described in each section.

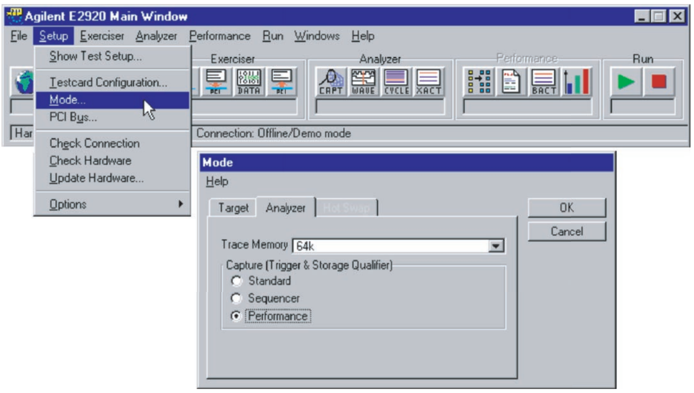
The setup of a post-performance analysis test is now described by means of an example.

Task

The performance of a graphic card is to be analyzed. Only write accesses to the card are to be captured. The measurements are to be restricted to memory commands. The accesses of all available master devices to this target are to be captured and analyzed.

Enabling Post-Processed Performance Analysis

To enable post-processed performance analysis, set the mode as shown in the following figure 4.



Connection: Offline/Demo mode

Trace Memory: 64k

Capture (Trigger & Storage Qualifier)

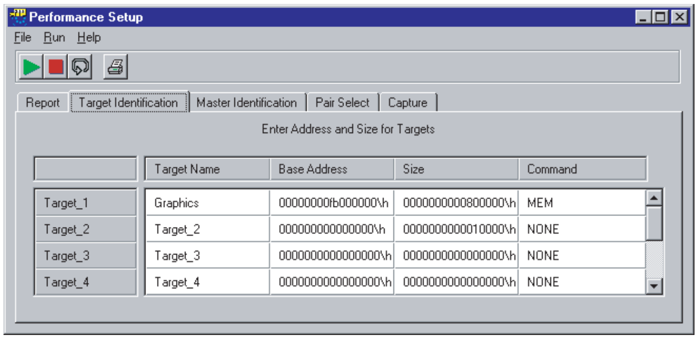
○ Standard
○ Sequencer
● Performance

Figure 4.

Setting Up the Device Identification

To identify the target:

1. Open the Performance Setup window by selecting Performance Setup... in the Performance menu.
2. Select the Target Identification tab.
3. Enter the target's base address and memory size in the Target Identification window.
4. Restrict your measurement to traffic using memory commands by selecting MEM from the Command list.



Enter Address and Size for Targets

	Target Name	Base Address	Size	Command
Target_1	Graphics	00000000fb000000\h	0000000000800000\h	MEM
Target_2	Target_2	000000000000000\h	0000000000010000\h	NONE
Target_3	Target_3	0000000000000000\h	0000000000000000\h	NONE
Target_4	Target_4	0000000000000000\h	0000000000000000\h	NONE

Figure 5.

Note:
To identify a PCI-X completer device, use the Completer Identification page and specify it by its address space just as for a PCI target.

Identifying particular masters is not necessary in this example, because all masters are analyzed.

Note for Identifying Masters and Requesters:

Identifying Masters (valid only with E2920 PCI software)
To identify particular masters, select the Master Identification tab.
After you have connected the GNT# and REQ# lines of the masters to the external trigger input pins of the testcard (if necessary, by soldering wires to them):

1. Enter the trigger port numbers that these masters' GNT# lines are connected to into the respective GNT# Pin fields.

2. Enter the trigger port number that these masters' REQ# lines are connected to into the respective REQ# Pin fields. Entering zero (0) means that GNT# and REQ# are not connected.

Identifying Requesters (valid only with E2920 PCI-X software)
To identify particular requesters, select the Requester Identification tab and enter the corresponding bus, device and function numbers.

Specifiying the Data Capture

To set up the trigger for data capture in the trace memory:

1. In the Performance Setup window, select the Capture tab.
2. Select Edit... in the Trigger on group.
   This opens the Capture window.

For a **PCI device**, proceed as follows:

- In the Capture window, select trigger on Pattern and click Edit ....
  This opens the Pattern Editor.

- In the Pattern Editor:

– For signal AD32, enter the target's address to trigger when this target memory space is accessed.

– For signal CBE3_0, enter xxx1\b to trigger on write commands.

– For signal b_state, enter Addr to trigger on address phases.

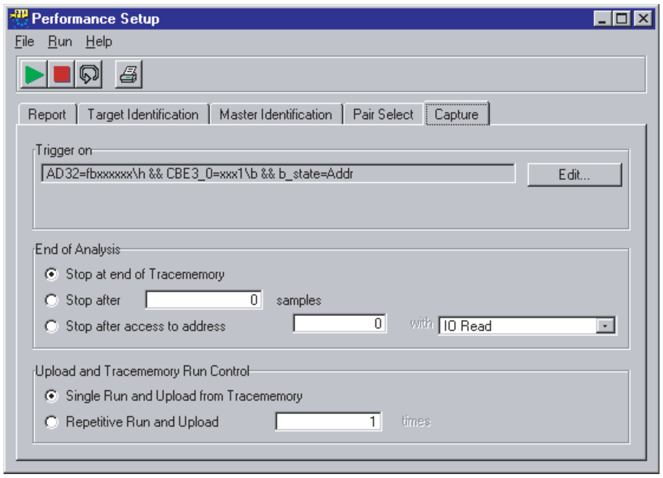
For a **PCI-X device,** proceed as follows:

- In the Capture window, select trigger on Bus pattern, click Edit ... and enter the line:
  AD32=fbxxxxxx\h &&
  CBE3_0=xxx1\b &&
  b_state=Addr

Select the Device Pair to be tested

In this performance test, the accesses from all masters (or requesters) to the device under test (graphic card) are to be analyzed:

In the Performance Setup window, click the Pair Select tab and select the pair All Masters (or All Requesters) and Graphics.



Trigger on: AD32=fbxxxxxx\h && CBE3_0=xxx1\b && b_state=Addr

End of Analysis
● Stop at end of Tracememory
○ Stop after 0 samples
○ Stop after access to address 0 with IO Read

Upload and Tracememory Run Control
● Single Run and Upload from Tracememory
○ Repetitive Run and Upload 1 times

Figure 6.

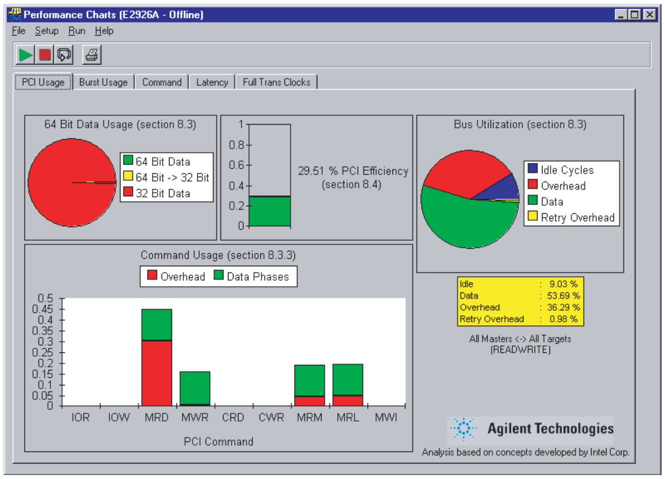
Viewing and Analyzing the Results

For viewing the results, proceed as follows:

1. Start the Performance Optimizer by clicking Run in the Performance menu.

2. Open the Performance Charts window from the Performance menu.

As an example, the PCI Usage chart looks as follows, in figure 7, (valid only with Agilent E2920 PCI software):



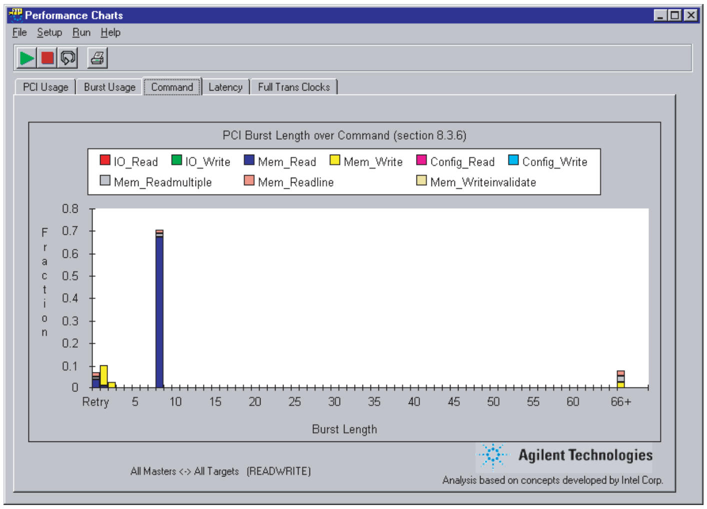
```
Idle            :  9.03 %
Data            : 53.69 %
Overhead        : 36.29 %
Retry Overhead  :  0.98 %
```

All Masters <-> All Targets
(READWRITE)

Figure 7.

Note:
Move the mouse over the pie charts to view the values of the various pie slices.

In the Command Usage chart, you can see that memory read command (MRD) causes the most overhead. It shows that using this command is not as efficient as on the extended commands memory read multiple (MRM) and memory read line (MRL).
The Command tab provides further information about the command usage.



All Masters <-> All Targets (READWRITE)

Figure 8.

This chart shows that the bus was mainly occupied with memory reads. Most of the memory read accesses were terminated by the target after 8 Dwords. When using MRM and MRL, transfers with burst lengths of over 66 Dwords are possible. Most of the data transfers were performed by using these extended commands.

Conclusion

Analyzing this performance charts helps you optimize the performance of the device under test. In this case, if more extended commands used, PCI Efficiency and PCI Utilization would increase.
The PCI Efficiency would increase, because more bytes could be transferred in less time.
The PCI Utilization (the relation between busy and idle bus time) would increase, because when using read multiple accesses, the host bridge realizes that large data transfers are initiated. As a result, the bridge performs prefetching and needs fewer wait states, retries and idle states.

4 Read and write register values of your device before a driver is available

When designing a device, you may need to read values from (peek) and write values into (poke) its registers or memory before a driver is available. The Agilent PCI/PCI-X Exerciser, a combination of testcard and software, simplifies this task. The testcard makes it possible to peek and poke the registers of the device under test either via an external host or via PCI within the control PC. The | testcard can be easily programmed by entering specific commands in the Command Line Interface of the Agilent's Exerciser GUI.

Aims of this Application Note

To show how to access registers of a device before a driver is available by using the Exerciser's Command Line Interface (CLI).

Questions that can be answered with the help of the PCI/PCI-X Exerciser

How can I peek and poke:
- Register values of a PCI device
- Memory spaces of a PCI device
- Register values of a PCI-X device before a driver is available?

Benefits of the Agilent PCI/PCI-X Exerciser

- Allows interactive work, which allows you to immediately see the effects of your changes.

Setting Up the Test

Plug the testcard into the system under test, and start the Agilent E2920 test software either on an external host or on the control PC.

If the software is running on an external host, connect the control PC via parallel or serial port to the testcard. (The parallel port is used in combination with the Fast Host Interface card plugged into the host PC, see figure below.)

If the software is running on the system under test, the testcard accesses the device under test via the PCI bus (see figure below).

To establish the connection to the testcard:

1. Open the Testcard Configuration dialog box by clicking this item in the Setup menu of the main window.

2. Select the port the testcard is connected to.

To communicate with the testcard, you need to send commands to it that you enter in the CLI. To open the CLI, select Command Line Interface in the Windows menu of the Main Window.

The following figures, 1+2 show the possible test setups.

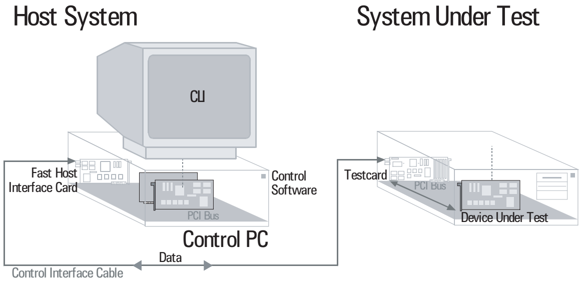


Figure 1.

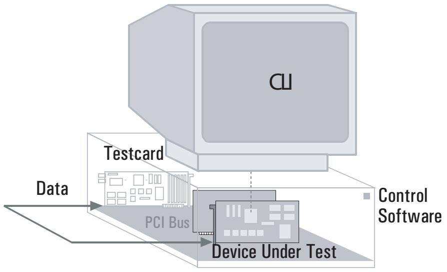


Figure 2

All commands allow access to the memory, I/O and configuration space of the device under test.

Note
All available commands used to access registers of PCI and PCI-X devices are introduced in the following examples. Note that not all the available parameter settings can be shown here. For further information on these, refer to the Summary of Commands Used at the end of this document.

Peeking and Poking Register Values of a PCI Device

This section shows how to peek and poke register values of a PCI device by means of the following examples:

- Peeking register values from the video frame buffer memory of a VGA graphics adapter
- Poking register values into the video frame buffer memory of a VGA graphics adapter

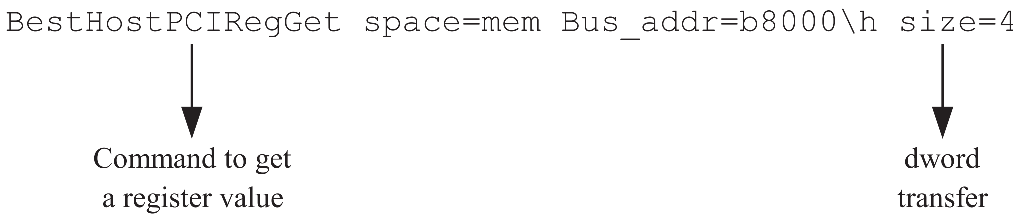
Peeking Register Values

Task

To peek register values from the video frame buffer memory of a VGA graphics adapter.

Procedure

1. Open the Command Line Interface.
2. Read a dword from the physical memory address 0xb8000 by entering the following command line (fig 3):



```
BestHostPCIRegGet space=mem Bus_addr=b8000\h size=4
```

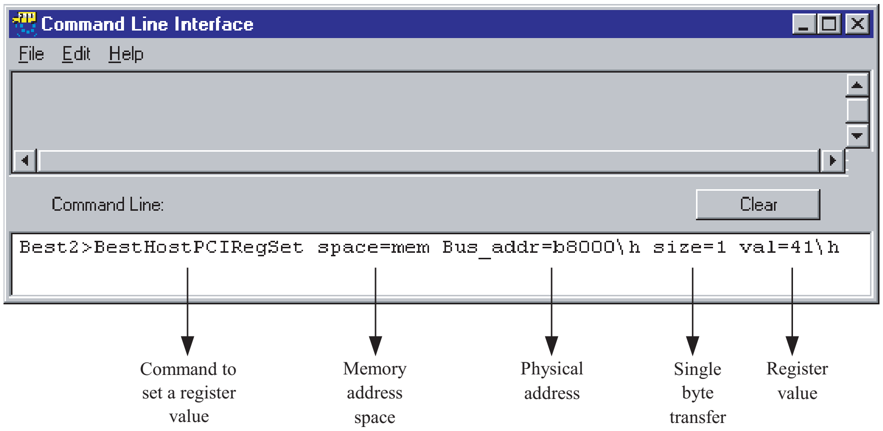
Figure 3.

Poking Register Values Task

To poke a character into the top left corner of the VGA text screen (visible in DOS mode only).

Procedure

1. Open the Command Line Interface (CLI).
2. Write a single byte to the physical memory address 0xb8000 by entering the following command line fig 4:



```
Best2>BestHostPCIRegSet space=mem Bus_addr=b8000\h size=1 val=41\h
```

Figure 4.

You can also poke the I/O or configuration space of a device by setting space=io or space=config.
Each successful completion of a command is indicated by Ready displayed in the CLI.

Results

The character 'A' will be displayed in the top left-hand corner of the DOS screen.

Peeking and Poking Memory Spaces of a PCI Device

The PCI/PCI-X Exerciser operates as follows when accessing memory spaces:

Data Transfer to the Device Under Test (Poking)

The Exerciser transfers data from the control PC to the device under test in two steps:

1. It transfers the data into a memory buffer in the testcard.

2. It writes the data from the testcard into the device under test.

Data Transfer from the Device Under Test (Peeking)

The Exerciser transfers data from the device under test to the control PC in two steps:

1. It reads the data from the device under test into a memory buffer in the testcard.

2. It transfers the data from the testcard to the control PC.

Because all data transfers require the use of the testcard's data memory, you need to allocate a buffer in this memory before you start the transfer.

The following examples show you how to use Exerciser commands to make data transfers:

- Peeking a data block from the video frame buffer memory of a VGA graphics adapter

- Poking a data block into the video frame buffer memory of a VGA graphics adapter

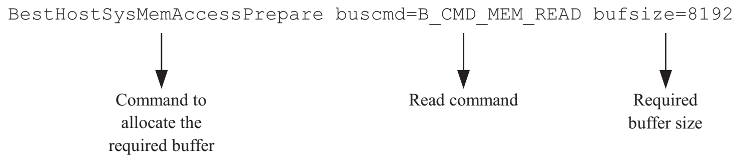
Peeking Memory Spaces

Task

To read 32 Kbytes from a VGA graphics adapter (PCI address 0xB8000000) to the memory of the control PC.

Procedure

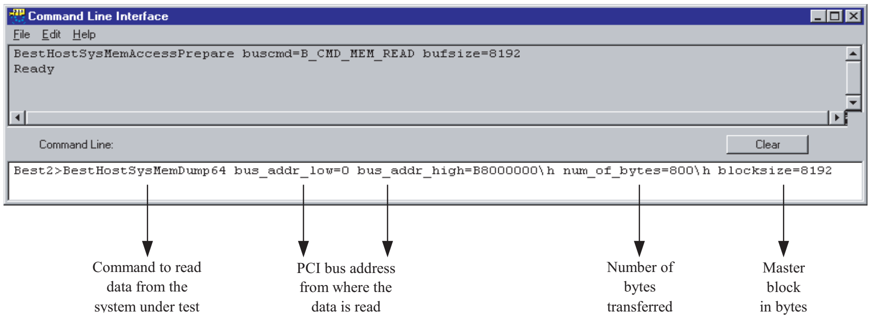
1. Open the CLI.
2. Allocate the required buffer in the testcard's data memory by entering the following command line, fig 5:



```
BestHostSysMemAccessPrepare buscmd=B_CMD_MEM_READ bufsize=8192
```

Figure 5.

3. Perform the data transfer by entering the following command line:



Command Line Interface
File Edit Help
Command Line:
Clear

```
BestHostSysMemAccessPrepare buscmd=B_CMD_MEM_READ bufsize=8192
Ready
```

```
Best2>BestHostSysMemDump64 bus_addr_low=0 bus_addr_high=B8000000\h num_of_bytes=800\h blocksize=8192
```

Command to read data from the system under test
PCI bus address from where the data is read
Number of bytes transferred
Master block in bytes

Figure 6

Results

The CLI displays the value read from the accessed memory space.

Poking Memory Spaces

Task

To write the data (1\h, 2\h, 3\h, 4\h, 5\h, 6\h, 7\h, 8\h) to a VGA graphic adapter (PCI address 0xB8000000).

Procedure

1. Open the CLI.
2. Allocate the required buffer in the testcard's data memory by entering the following command line fig 7:

```
BestHostSysMemAccessPrepare buscmd=B_CMD_MEM_WRITE bufsize=8192
```

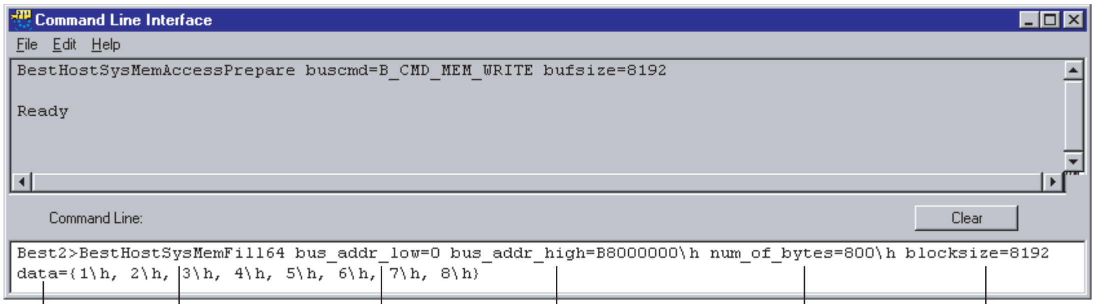
Command to allocate the required buffer

Write command

Required buffer size

Figure 7.

3. Perform the data transfer by entering the following command line fig 7.b:



```
BestHostSysMemAccessPrepare buscmd=B_CMD_MEM_WRITE bufsize=8192

Ready
```

```
Best2>BestHostSysMemFill64 bus_addr_low=0 bus_addr_high=B8000000\h num_of_bytes=800\h blocksize=8192
data={1\h, 2\h, 3\h, 4\h, 5\h, 6\h, 7\h, 8\h}
```

Transfer data

Command to write data to the system under test

PCI bus address to which the data is written

Number of bytes transferred

Master block size in bytes

Figure 7.b

Results

The values are written to the PCI device memory

Peeking and Poking Register Values of a PCI-X Device (valid only with Agilent E2920 PCI-X Card)

The principles used to program data transfers between the Exerciser and a PCI-X device are the same as for PCI devices. The commands provided by the PCI-X Exerciser software to access the registers of a PCI-X device are introduced in the following examples.

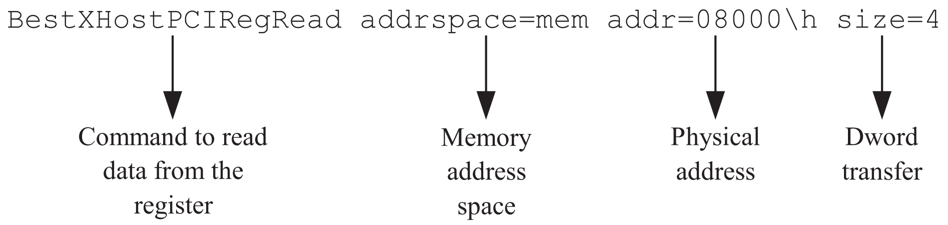
Peeking Register Values

Task

To transfer a dword from a register in a PCI-X device at the physical memory address 0x8000 to the control PC.

Procedure

1. Open the Command Line Interface.
2. Read the register data by entering the following command line fig 8:



```
BestXHostPCIRegRead addrspace=mem addr=08000\h size=4
```

Figure 8.

Results

The register value is displayed in the Command Line Interface.

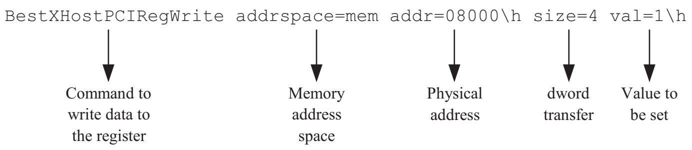
Poking Register Values

Task

To transfer a dword from the control PC to a register at the physical memory address 0x8000 of a PCI-X device.

Procedure

1. Open the CLI.
2. Write the value to the register by entering the following command line fig 7 c:



```
BestXHostPCIRegWrite addrspace=mem addr=08000\h size=4 val=1\h
```

Figure 7c.

Results

The value is written to the PCI-X device register.

Summary of Commands Used

Commands for PCI

BestHostPCIRegGet / BestHostPCIRegSet

Description Reads/writes the value from/to a specific PCI device register in a 32-bit address space—the type address space determines the configuration, memory or I/O read.

Syntax
BestHostPCIRegGet space=<addrspace> bus_addr=<bus_addr> size=<size>
BestHostPCIRegSet space=<addrspace> bus_addr=<bus_addr> size=<size> val=<reg_value>

Parameters

addrspace:	mem	(Memory Space)
	io	(IO Space)
	config	(Type 0 access to Config Space)
	config_type	(Type 1 access to Config Space)
bus_addr:	PCI bus address	
size:	1	(byte)
	2	(word)
	4	(Dword)
reg_val:	Register Value	

BestHostSysMemAccessPrepare

Description Prepares the internal address, the command in the master block properties, and a memo buffer for a transfer through the data memory of the testcard. Data verification can be activated.

Syntax BestHostSysMemAccessPrepare buscmd=<buscmd> bufsize=<bufsize>

Parameters

buscmd:	B_CMD_MEM_WRITE	(memory write)
	B_CMD_MEM_READ	(memory read)
bufsize:	internal memory buffer size in dwords (minimum: 2 dwords)	

BestHostSysMemDump64/BestHostSysMemFill64

Description Transfers data from a PCI device to the host system memory./
Transfers data from the host system memory to a PCI device.

Syntax

BestHostSysMemDump64	bus_addr_low=<bus_addr_low>
	bus_addr_high=<bus_addr_high>
	num_of_bytes=<num_of_bytes> blocksize=<blocksize>
	[data>"file path"]

BestHostSysMemFill64	bus_addr_low=<bus_addr_low>
	bus_addr_high=<bus_addr_high>
	num_of_bytes=<num_of_bytes>
	blocksize=<blocksize>
	(data=<{data_list}>) | (data<"file path")

Parameters

bus_addr_low/ bus_addr_high:	PCI bus address
num_of_bytes:	Number of bytes to be transferred (maximum of 128 Kbytes).
blocksize:	Size of the master block transfers in bytes.
data_list:	List of data to be transferred.
data<"file path" (optional):	File to which the data can be exported.

Commands for PCI-X

BestXHostPCIRegRead / BestXHostPCIRegWrite

Description Reads/writes the value from/to a specific PCI-X device register in a 32-bit address space—the type of address space determines the Configuration, Memory or I/O Read.

Syntax	BestXHostPCIRegRead	space=<addrspace> bus_addr=<bus_addr> size=<size>
	BestXHostPCIRegWrite	space=<addrspace> bus_addr=<bus_addr> size=<size> val=<reg_value>

Parameters	addrspace:	mem	(Memory Space)
		io	(IO Space)
		config	(Access to Config Space)
	bus_addr:	PCI bus address	
	size:	(byte)	1
		(word)	2
		(Dword)	4
	reg_val:	Register Value	Register Value

5 Monitor How Your Device Acts on the PCI/PCI-X Bus

Monitoring traffic on the PCI/PCI-X bus is done easily, but if you are designing PCI/PCI-X devices, you need to monitor the traffic relating to a particular device. This application allows you to debug your system quickly and effectively.

The Agilent PCI/PCI-X Analyzer, a combination of testcard and software, simplifies this task. By specifying trigger points, you can focus on a particular device on the bus.

Aims of this Application Note

- To show a simple way of capturing traces of a particular device on the PCI/PCI-X bus in order to analyze functional problems.
- To introduce the trigger sequencer, which lets you perform more advanced measurements.

Questions that can be answered with the help of the PCI/PCI-X Analyzer

- How does data transfer to or from a particular device take place?
  Here the Analyzer can help you resolve performance problems. You can see whether the transfer consists mainly of byte transfers or single transfers, or if too many retries are taking place.

- During which bus transaction is data being incorrectly transferred?

- How do changes to the design of your device affect its behavior on the bus?

Benefits of the Agilent PCI/PCI-X Analyzer

- Provides three different display modes to view and evaluate the captured PCI/PCI-X traffic.
  You can set the modes so that the resulting views correlate with one another. This considerably eases the task of analyzing your device.

- Includes a programmable trigger sequencer.
  This sequencer implements a state machine that allows you to define individual trigger and storage qualifier conditions for each state.

Setting Up the Test

Introduction

This application note is based around an example that shows you how the Agilent testcard can be used to capture traffic on a PCI or PCI-X bus. Differences between setups for PCI and PCI-X bus systems are explicitly stated.

The example shows you how to set up the testcard to trigger on a particular address range and capture bus traffic that occurs around this trigger point (both prior to and following it). The software offers two methods to let you set the trigger point:

- Single-Level Triggering (for simple measurements)
- The Trigger Sequencer (for more advanced measurements)

Single-Level Triggering

Here the aim is to set up the trigger and the storage qualifier for the Analyzer.
The trigger defines when the Analyzer starts capturing data. The trigger allows you to start capturing data when the programmed trigger event has occurred.
The storage qualification allows you to selectively capture only certain types of data.

Setting Up the Trigger

The following steps explain how to set up the trigger:

1. Open the Capture window. Use the Capture button in the icon bar of the main window, or choose Capture from the Analyzer menu.

2. Open the Pattern Editor dialog box:
- Select the Trigger tab.
- Choose trigger on Pattern.
- Click the Edit button next to the pattern term.

3. Set up the testcard to start capturing data when a single address phase occurs on the bus in the address range of 0xfb000000 to 0xfbffffff. The b_state signal gives an indication of the bus state. It is internally generated by the testcard and is used as a trigger source.
- Set the AD32 field in the Pattern Editor dialog box to FBxxxxxx\h.
- Click on the text field to the right of the signal b_state to open a Selection List dialog box.
- Highlight Addr and click the right-arrow button to place it in the right (Selected) box. Press OK.
- Use the left-arrow button to remove the "DON'T CARE" entry from the Selected list.

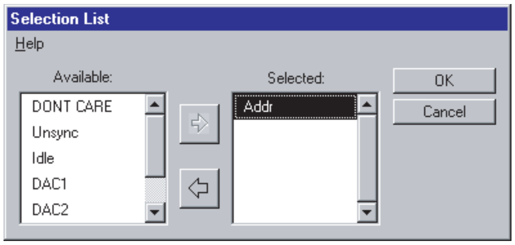
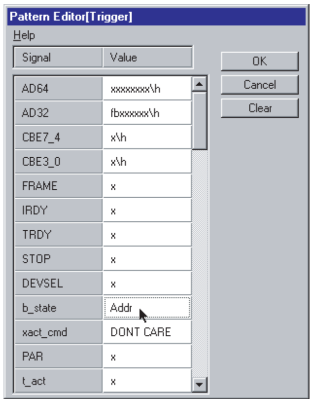


Figure 1.

4. In the Capture window (see below), you can select whether the Analyzer should trigger on the occurrence of the specified pattern on the bus, or on the absence of the pattern within a certain number of clocks.

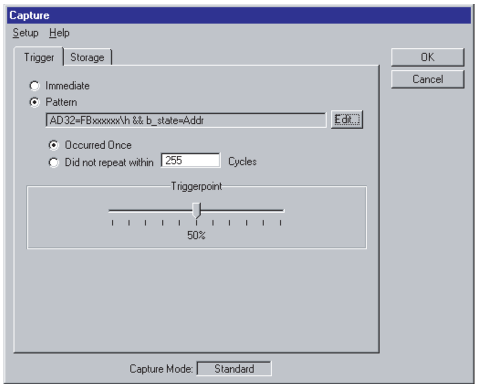
In this example, you want the Analyzer to trigger immediately after the specified pattern has occurred on the bus. Click the Occurred Once radio button.
The Pattern editor window should now contain the following values:



Signal	Value
AD64	xxxxxxxx\h
AD32	fbxxxxxx\h
CBE7_4	x\h
CBE3_0	x\h
FRAME	x
IRDY	x
TRDY	x
STOP	x
DEVSEL	x
b_state	Addr
xact_cmd	DONT CARE
PAR	x
t_act	x

Figure 2.

5. Click OK.
The Capture window should now look like this. The default position for the trigger point slider is at 50%. That means that 50% pre-trigger history and 50% post-trigger history is stored.



○ Immediate
● Pattern
AD32=FBxxxxxx\h && b_state=Addr
● Occurred Once
○ Did not repeat within 255 Cycles
Triggerpoint: 50%
Capture Mode: Standard

Figure 3.

Setting Up the Storage Qualifier

The Storage page of the Capture window allows you to define the storage qualifier.
The storage qualifier instructs the Analyzer to capture:

- Unconditionally one sample per clock.

- Only certain types of data.

Unconditionally Capturing One Sample per Clock

Use the default setting All.

Capturing Only Certain Types of Data

Choose Selected Transactions/States to suppress idle cycles between transactions and/or wait states during a transaction.
You can further restrict what is captured in the trace memory by storing only particular transaction types (for example, only memory writes).
Click the Edit... button and select the required storage restrictions from the Pattern Editor list.

Using the Trigger Sequencer

The following example highlights the advantages of using the trigger sequencer when setting up sophisticated trigger conditions. It also shows how the captured data can be viewed with the three display modes to analyze PCI behavior of the tested device.

The example shows how to set up the trigger to test a target device and a master device. The device under test (DUT) is a PCI graphics card.

Setting Up the Trigger Sequencer

The aim is to monitor only the traffic from or to a particular device. You therefore need to program the Analyzer to begin capturing data when access to this device starts and to stop capturing data as soon as access terminates.

Setting up the trigger condition is simple: You have to provide the analyzer with the base address of your DUT and specify the associated sequence that is to be detected.

Defining a Sequence

The Analyzer has to act as follows:

1. It waits for the address phase of an access to a card for example the graphic card.

2. When the address phase is detected, it triggers and stores all the transfers.
3. It stops storing if an idle cycle occurs (it ignores the idle cycle).
4. It waits for the next access to the graphic card.

The trigger sequencer is a programmable state-machine. The above sequence can be illustrated by a state diagram as shown below. The trigger conditions are determined by patterns pt0 and pt1.

Setting The Trigger Conditions

For this example, the Analyzer is set up to trigger on a PCI address phase with an address in the range between 0xfb000000 and 0xfbxxxxxx.

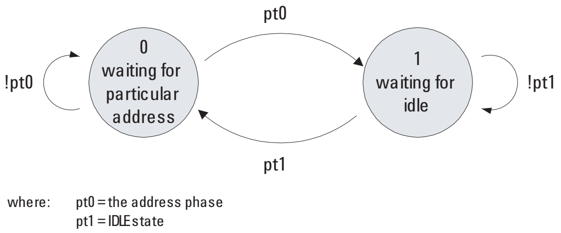


Figure 4.

If you do not know the address of your device, use the Agilent testcard to perform a configuration scan of the PCI bus:

1. Plug the testcard into the primary PCI bus.

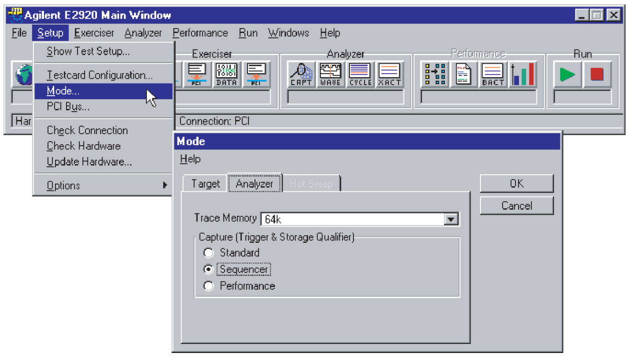
2. Open the Command Line Interface from the window menu.

3. Enter the commands cscan and then cscanprnt (PCI only), each followed by the return key.

The CLI lists all devices found on the PCI bus including their address spaces.

Programming the Trigger Sequencer and the Trigger Conditions for a Target PCI Device

To use the trigger sequencer, the Analyzer must operate in the Sequencer Capture mode. Set the mode as shown in the following figure 5:



Trace Memory: 64k

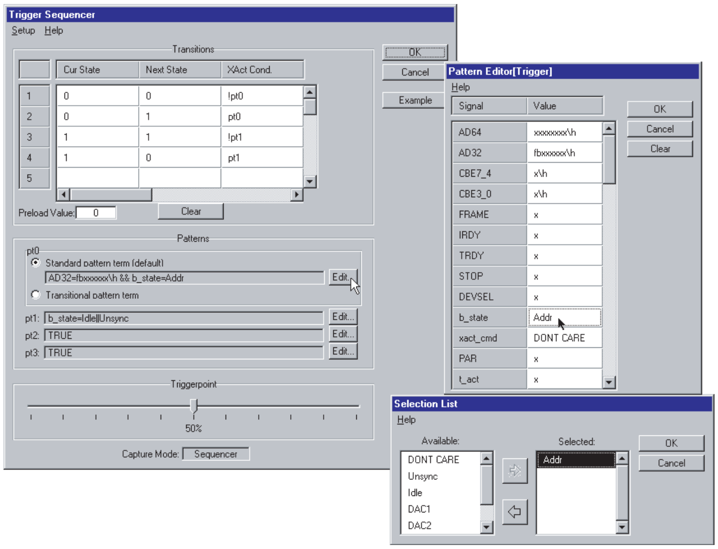
Capture (Trigger & Storage Qualifier)

○ Standard
● Sequencer
○ Performance

Figure 5.

After you have set the mode, you can set the sequence and the trigger conditions in the Trigger Sequencer dialog box. Open the dialog box by clicking the Capture button CAPT in the main window or by selecting Capture ... in the Analyzer menu.

1. Enter the sequence which you have specified previously in Defining a Sequence. In the following figure, the sequence settings have been entered in the Trigger Sequencer dialog box:



	Cur State	Next State	XAct Cond.
1	0	0	!pt0
2	0	1	pt0
3	1	1	!pt1
4	1	0	pt1
5			

Preload Value: 0

pt0

● Standard pattern term (default)

AD32=fbxxxxxx\h && b_state=Addr

○ Transitional pattern term

pt1: b_state=Idle||Unsync

pt2: TRUE

pt3: TRUE

Capture Mode: Sequencer

Signal	Value
AD64	xxxxxxxx\h
AD32	fbxxxxxx\h
CBE7_4	x\h
CBE3_0	x\h
FRAME	x
IRDY	x
TRDY	x
STOP	x
DEVSEL	x
b_state	Addr
xact_cmd	DONT CARE
PAR	x
t_act	x

Available:

DONT CARE
Unsync
Idle
DAC1
DAC2

Selected:

Addr

Figure 6.

2. Click the Edit ... button to open the Pattern Editor. Enter the trigger condition values.

3. Finally, assign the value Addr to the signal b_state by selecting it in the Selection List.

Setting the Trigger Condition to Monitor a Master PCI Device

In order to measure and evaluate PCI traffic initiated by particular master devices, you need to know when the master is active. For this purpose, the masters' GNT# (grant) lines must be made accessible to the Agilent testcard. These lines must be connected to the external trigger input pins of the testcard- if necessary, by soldering wires to them.

Defining a Sequence

The Analyzer has to act as follows:

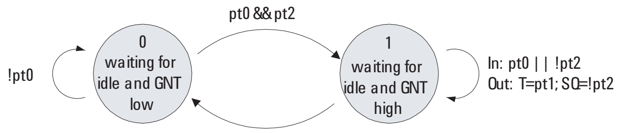
1. It waits until the master becomes active (GNT# line is low) while the bus is idle (trigger0=0 && b_state=idle).

2. When these conditions are met, it triggers on the address phase (T=b_state=addr) and stores the transfers, except idle states (SQ=!(bstate=idle)).

3. It stops storing if an idle cycle occurs and the master's GNT# line is high. (trigger0=1 && b_state=idle).

4. It waits for the next transfer started by the master.

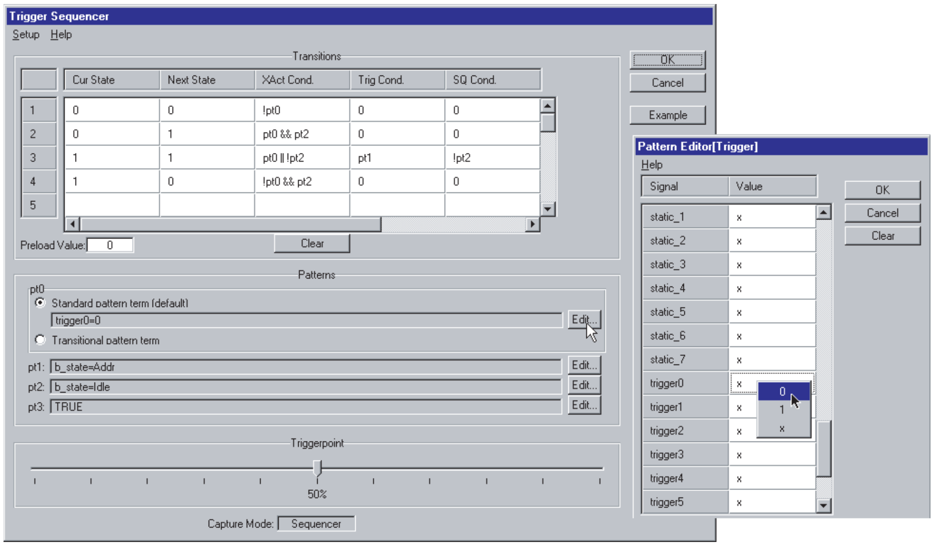
The following figure shows the



where:
pt0 = GNT low (trigger0 = 0)
pt1 = address phase
pt2 = IDLE state
T = Trigger Condition
SQ = Storage Qualifier

Figure 7.

In this example, the Master's GNT# line is connected to the external trigger input pin[0]. Enter the trigger conditions in the Trigger Sequencer dialog box as follows:



Transitions

	Cur State	Next State	XAct Cond.	Trig Cond.	SQ Cond.
1	0	0	!pt0	0	0
2	0	1	pt0 && pt2	0	0
3	1	1	pt0 || !pt2	pt1	!pt2
4	1	0	!pt0 && pt2	0	0
5					

Preload Value: 0

Patterns

pt0
● Standard pattern term (default)
trigger0=0
○ Transitional pattern term

pt1: b_state=Addr
pt2: b_state=Idle
pt3: TRUE

Triggerpoint 50%

Capture Mode: Sequencer

Signal	Value
static_1	x
static_2	x
static_3	x
static_4	x
static_5	x
static_6	x
static_7	x
trigger0	x
trigger1	x
trigger2	x
trigger3	x
trigger4	x
trigger5	x

0
1
x

Figure 8.

Using the Trigger Sequencer for PCI-X Devices (valid only with Agilent E2920 PCI-X Series)

The following example shows how to set up the trigger to test a PCI-X graphics card.

Defining a Sequence

The PCI-X Analyzer has to act as follows:

1. It waits for the address phase of an access to the graphic card for example.

2. When the address phase is detected, it triggers and stores all the transfers.

3. It ignores any idle cycles.

4. It waits for the next access to the graphic card for example.

Setting the Trigger Conditions

With the PCI-X software, pattern terms are devided in bus and observer pattern terms. For this sequence, the following patterns need to be detected:

- bus0: addr_phase==1 && AD32==fbxxxxxx\h
  Bus pattern term bus0 detects an address phase that addresses the range fb000000 ... fbffffff.
- obs0: xact_cmd==Mem_ReadDW && ri_act==1
  Observer pattern term obs0 detects Memory Read DWord transfers executed by the requester-initiator.

- obs1: bstate==Idle
  Observer pattern term obs1 detects idle cycles.

The following figure shows the state diagram of this sequence:

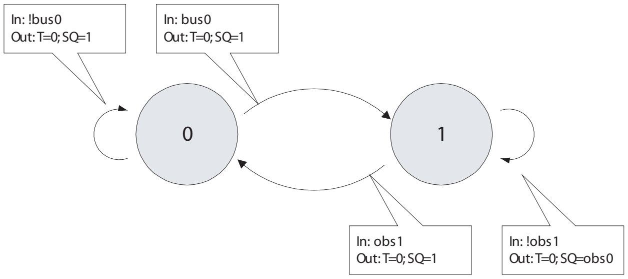


Figure 9.

Setting Up the Trigger Sequencer

Setting up the trigger sequencer for PCI-X devices is very similar to PCI devices:

1. To use the trigger sequencer, the PCI-X Analyzer must operate in the Sequencer Capture mode. To set this mode, check Trigger Sequencer under Options in the Setup menu:

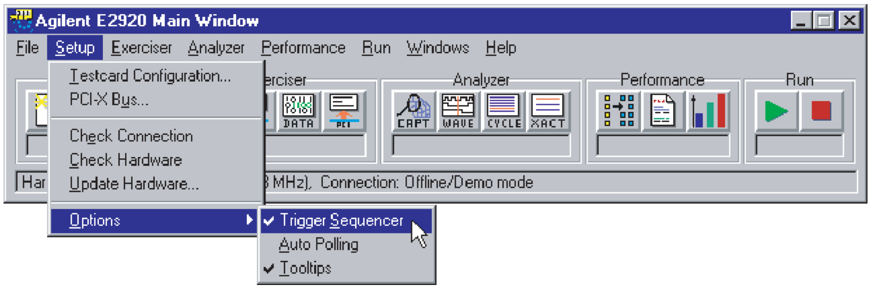
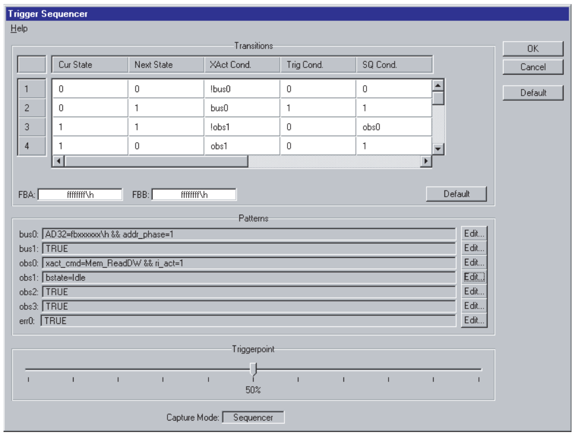


Figure 10.

2. After you have set the mode, you can set the sequence and the trigger conditions in the Trigger Sequencer dialog box. Open the dialog box by clicking the Capture button CAPT in the main window or by selecting Capture ... in the Analyzer menu.

3. Enter the sequence and the bus and observer pattern terms that you have previously specified. This can be done in the same way as for PCI devices.

In the following figure, the sequence settings and pattern terms have been entered in the Trigger Sequencer dialog box:



Trigger Sequencer

Help

Transitions

	Cur State	Next State	XAct Cond.	Trig Cond.	SQ Cond.
1	0	0	!bus0	0	0
2	0	1	bus0	1	1
3	1	1	!obs1	0	obs0
4	1	0	obs1	0	1

FBA: ffffffff\h FBB: ffffffff\h

Patterns

bus0: AD32=fbxxxxxx\h && addr_phase=1
bus1: TRUE
obs0: xact_cmd=Mem_ReadDW && ri_act=1
obs1: bstate=Idle
obs2: TRUE
obs3: TRUE
err0: TRUE

Triggerpoint

50%

Capture Mode: Sequencer

Figure 11.

Running the Test

To run the test, press the Run button ▶ in the main window or select Run in the Analyzer menu.

Viewing and Analyzing the Results

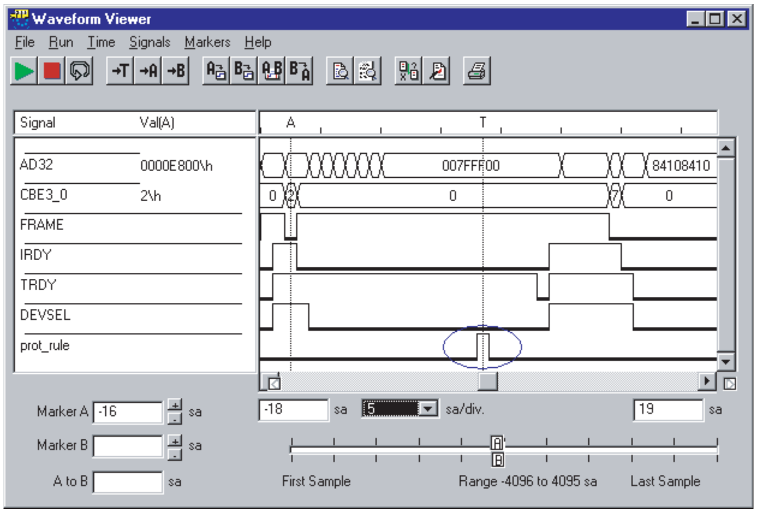
To help you analyze the captured data, three different display modes are available:

- Waveform Viewer to analyze captured waveforms
- Bus Cycle Lister to analyze bus cycles
- Transaction Lister to analyze transactions

Analyzing Captured Waveforms

The Waveform Viewer provides a detailed view of the signals on the bus. It is the appropriate display mode when analyzing single transactions or when checking the state of individual control signals.

Open the Waveform Viewer by clicking the Waveform Viewer button WAVE in the main window (or select the Waveform Lister item in the Analyzer menu).



Signal	Val(A)
AD32	0000E800\h
CBE3_0	2\h
FRAME	
IRDY	
TRDY	
DEVSEL	
prot_rule	

Marker A -16 sa

Marker B sa

A to B sa

-18 sa 5 sa/div. 19 sa

Range -4096 to 4095 sa

Figure 12.

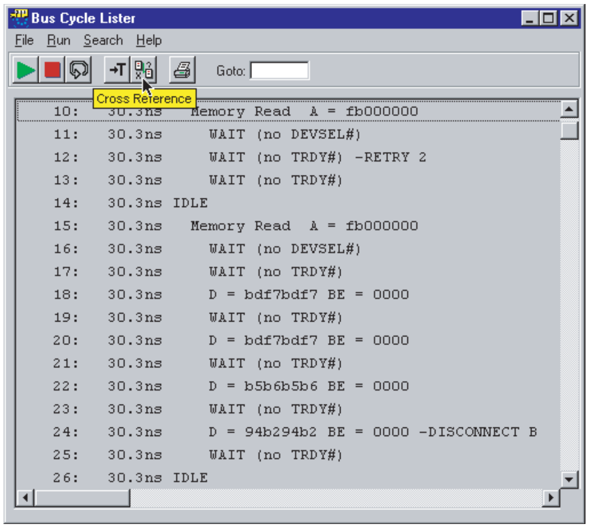
Click the Trigger button →T to display data around the trigger point.
To view different signals, select them in the Arrange Signals dialog box. This is available in the Signals menu.

Analyzing Bus Cycles

The Cycle Lister provides a more compact view of the captured PCI transactions.

Open the Bus Cycle Lister by clicking the Bus Cycle Lister button

in the main window (or select the Bus Cycle Lister item in the Analyzer menu).



```
   10:   30.3ns   Memory Read  A = fb000000
   11:   30.3ns     WAIT (no DEVSEL#)
   12:   30.3ns     WAIT (no TRDY#) -RETRY 2
   13:   30.3ns     WAIT (no TRDY#)
   14:   30.3ns IDLE
   15:   30.3ns   Memory Read  A = fb000000
   16:   30.3ns     WAIT (no DEVSEL#)
   17:   30.3ns     WAIT (no TRDY#)
   18:   30.3ns     D = bdf7bdf7 BE = 0000
   19:   30.3ns     WAIT (no TRDY#)
   20:   30.3ns     D = bdf7bdf7 BE = 0000
   21:   30.3ns     WAIT (no TRDY#)
   22:   30.3ns     D = b5b6b5b6 BE = 0000
   23:   30.3ns     WAIT (no TRDY#)
   24:   30.3ns     D = 94b294b2 BE = 0000 -DISCONNECT B
   25:   30.3ns     WAIT (no TRDY#)
   26:   30.3ns IDLE
```

Figure 13

To view data around the trigger point, click the Trigger button

To examine the displayed cycles in more detail, click the Cross Reference button

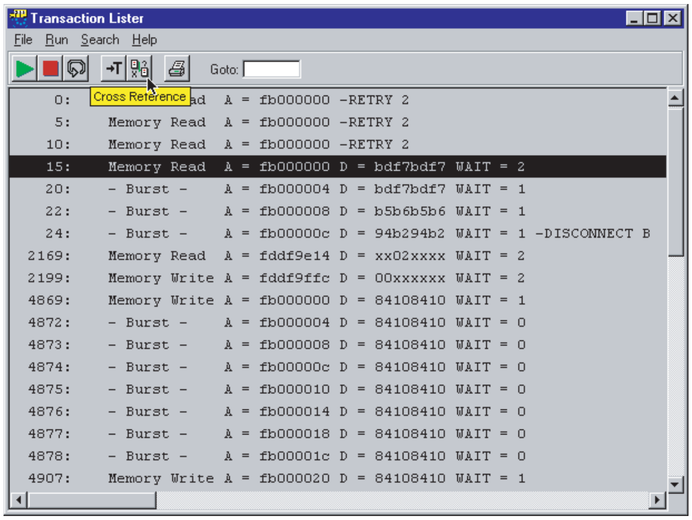
(highlighted in the above figure). The Waveform Viewer displays the data. (The Waveform Viewer must be active and not iconized.)

Analyzing Captured Transactions

The Transaction Lister provides a compressed overview of the transactions that occurred on the bus. It removes idles from the display and summarizes the number of waits for each data phase, displaying only useful information such as address and data phases. Address reconstruction is also done during bursts.

Open the Transaction Lister by clicking the Transaction Lister button

in the main window (or use the Transaction Lister item in the Analyzer menu).



```
    0:                ad  A = fb000000 -RETRY 2
    5:    Memory Read  A = fb000000 -RETRY 2
   10:    Memory Read  A = fb000000 -RETRY 2
   15:    Memory Read  A = fb000000 D = bdf7bdf7 WAIT = 2
   20:    - Burst -    A = fb000004 D = bdf7bdf7 WAIT = 1
   22:    - Burst -    A = fb000008 D = b5b6b5b6 WAIT = 1
   24:    - Burst -    A = fb00000c D = 94b294b2 WAIT = 1 -DISCONNECT B
 2169:    Memory Read  A = fddf9e14 D = xx02xxxx WAIT = 2
 2199:    Memory Write A = fddf9ffc D = 00xxxxxx WAIT = 2
 4869:    Memory Write A = fb000000 D = 84108410 WAIT = 1
 4872:    - Burst -    A = fb000004 D = 84108410 WAIT = 0
 4873:    - Burst -    A = fb000008 D = 84108410 WAIT = 0
 4874:    - Burst -    A = fb00000c D = 84108410 WAIT = 0
 4875:    - Burst -    A = fb000010 D = 84108410 WAIT = 0
 4876:    - Burst -    A = fb000014 D = 84108410 WAIT = 0
 4877:    - Burst -    A = fb000018 D = 84108410 WAIT = 0
 4878:    - Burst -    A = fb00001c D = 84108410 WAIT = 0
 4907:    Memory Write A = fb000020 D = 84108410 WAIT = 1
```

Figure 14

Glossary

External Trigger

The testcard provides input lines to be used as trigger input and output.

As input, they can be used in pattern terms just like PCI/PCI-X signals.

As output, they can be used to trigger other devices.

Pattern Term

The pattern terms are programmable for recognizing bus events (signal patterns on the bus). They are part of the testcard's Analyzer.

The output of pattern terms (always 1 or 0) can be used in Analyzer functions, for example, to trigger trace memory or to count bus events.

PCI Analyzer

The testcard's Analyzer queries for information on the PCI bus, the state of the Exerciser, or the external trigger inputs, so that it can check timing and protocol rules, capture traffic and measure performance.

PCI-X Analyzer

The testcard's Analyzer queries for information on the PCI-X bus, the state of the Exerciser, or the external trigger inputs, so that it can check capture traffic.

PCI Protocol Observer The PCI Protocol Observer is part of the PCI Analyzer. It is hardware implemented on the Agilent testcard, and simultaneously checks 53 different protocol rules against the PCI Specification.

PCI-X Analyzer

The PCI-X Protocol Observer is hardware implemented on the Agilent test card, and simultaneously checks 53 different protocol rules against the PCI-X Specification.

Sequencer

The sequencers of the testcard recognize sequences of patterns on the PCI bus. They are used by the testcard's Analyzer.

A sequencer can be programmed to change between internal states with the occurrence of certain patterns. On each state change, it can issue certain output signals. To recognize these patterns, pattern terms are used.

Storage Qualifier

Storage qualifiers are used by the Analyzer when storing samples in the trace memory. The trace memory stores samples of bus states for post-processed analysis. To exclude unnecessary samples from being stored, the Analyzer provides storage qualifiers.

Trace Memory

The trace memory is part of the testcard's Analyzer. It stores all PCI/PCI-X signals along with extensive bus state and Exerciser state information. All this information is aligned.

Trace Memory Trigger

The trace memory trigger is part of the Analyzer. The trace memory stores samples of bus states for post-processed analysis. To control the start of the sampling, the Analyzer provides the trace memory trigger.

PPR Technology

PPR permutates the PCI or PCI-X protocol parameters and data traffic in a pseudo-random way.

Protocol Checker

The testcard's protocol checker continuously monitors the bus and checks for violations of predefined protocol rules, which are partly defined by the PCI and the PCI-X specification and partly by Agilent.

Each individual rule can be masked out and will then neither trigger the trace memory nor appear in any report. To mask rules, click the details button next to the Use Protocol Checker (Rule Masking) check box.

Completer

A PCI-X device addressed by a transaction (other than a Split Completion) on the PCI-X bus.

Performance Measure

The performance measures are circuits consisting of sequencer-controlled counters. They count different bus events and calculate specified ratios. For example, busy cycles of transfers can be counted and divided by all cycles, thereby calculating the bus utilization.

Requester

A PCI-X device that first introduces a transaction. If the completer or a bridge terminates the transaction with Split Response, the requester becomes the target of the subsequent Split Completion.

Control PC

This PC runs the software that controls the testcard.

Data Memory

The data memory holds received test data, and data to be transferred by the test card. It is shared between the master and the target and can be set up or read out with host access functions. It also provides a data compare unit to compare incom ing data with previously stored reference data.

Exerciser (PCI)

The PCI Exerciser functions enable the testcard to emulate a PCI master or target.

Exerciser (PCI-X)

The PCI-X Exerciser functions enable the testcard to emulate a PCI-X requester-initiator, completer-target, completer-initiator or requester-target:

Host

Same as Control PC.

Peeking

Reading values from device registers or memory spaces.

Poking

Writing values into specific device registers or memory spaces.

System Under Test

The system under test is the PCI/PCI-X system into which the testcard is plugged

PCI Protocol Observer

The PCI Protocol Observer is part of the PCI Analyzer. It is hardware implemented on the Agilent testcard, and simultaneously checks 53 different protocol rules against the PCI Specification.

PCI-X Protocol Observer

The PCI-X Protocol Observer is hardware implemented on the Agilent test card, and simultaneously checks 53 different protocol rules against the PCI-X Specification.

Overview PCI/PCI-X E2920 Series

	PCI Analyzer -protocol checker -64K state PCI logic analyzer -4MB fast host interface -timing checker -real-time performance measures -GUI -RS-232 interface	**E2940A** compact PCI 32/64 bit 66MHz	**E2925B** PCI 32 bit 33 MHz	**E2928A** PCI 32/64 bit 66 Mhz	**E2929B** PCI-X 32/64 bit 133 MHz -protocol checker -RS-232/USB interface - GUI
Opt.100					**PCI-X Analyzer** - 2M state PCI logic analyzer -4MB fast host interface -real-time performance measures - GUI
Opt..200	**PCI Performance Optimizer**	4M trace memory recommended please order separately		32/64 bit 66 MHz	**PCI-X Performance Optimizer** -post processed and real-time performance analyzer -performance report - GUI
Opt.300	**PCI Exerciser** - master and target -GUI - CLI -512 KB on-board memory	32/64 bit 33 MHz	32 bit 33 MHz	32/64 bit 66 MHz	PCI-X Exerciser - master - target - GUI - 1MB onboard data memory
Opt. 310	**System validation package**	- peer-to-peer test -system memory test - system load test - protocol load test - protocol check - GUI			
Opt.320	**C-API/PPR**	- C-programming interface library -protocol permutation and randomization library			

PCI-PCI-X Bundle:

With the E2997A Agilent also offers a great price on the purchase of the E2928A PCI Card and the E2929B PCI-X card.

Master Target Test Card

The E2922B PCI-X Master target Test card provides validation engineers in the semiconductor industry a fast and predictable way to set up PCI-X protocl compliance of first silicon.

Accessories Agilent Products	**E2940A**	**E2925B**	**E2928A**	**E2929B**
E2991A External power supply		•	•	•
E2993A External Agilent Logic Analyzer Adapter		•	•	
E2994A External general purpose Logic Analyzer Adapter		•	•	
E2995A 155 x 4M trace memory		•	•	
E2996A 155 x 4M trace memory	•	•		
System test library	•	•	•	•

Agilent Technologies' Test and Measurement Support, Services, and Assistance

Agilent Thnologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Support is available for at least five years beyond the production life of the product. Two concepts underlay Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise

Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you use Agilent equipment, we can verify that it works properly, help with product operation, and provide basic measurement assistance for the use of specified capabilities, at no extra cost upon request. Many self-help tools are available.

Your Advantage

Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and on-site education and training, as well as design, system integration, project management, and other professional services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

Related Agilent Literature

- Agilent E2925B 32bit, 33 MHz, Agilent E2926B 32/64bit, 33 MHz PCI Exerciser & Analyzer, technical specifications, p/n 5968-3501E
- Agilent E2928A 32/64bit, 66 MHz, PCI Exerciser & Analyzer, technical specifications, p/n 5968-3506E
- Agilent E2929A PCI Exerciser & Analyzer, technical specifications, P/n 5968-8984E
- Agilent E2922A PCI-X Master Target Card, technical overview, p/n 5968-9577E
- Agilent E2940A CompactPCI Exerciser & Analyzer, technical overview, P/n 5968-1915E
- Agilent E2976A System Validation Pack, Agilent E2977A System Test Library, technical overview, p/n 5968-3500E
- Agilent E2920 Computer Verification Tools, PCI Series, brochure, p/n 5968-9694E
- Intel discusses basic concepts of PCI performance and efficient use of PCI with the Agilent E2920 series, case study, p/n 5988-0448ENDE
- HP NSD stabilizes server designs quickly and completely with the Agilent E2920 PCI Series, case study, p/n 5968-6948E
- HP HSTC speeds high-end server testing and reduces engineering costs with the Agilent E2920 PCI Series, case study, p/n 5968-6949E
- Agilent E2920 Verification Tools, PCI Series gives Altera Corporation competitive Advantage, case study, p/n 5968-4191E

You can find the current literature and software at:
www.agilent.com/find/pci_products
For more product information, please visit us at:
www.agilent.com/find/pci_overview

Agilent Email Updates

www.agilent.com/find/emailupdates

Get Free Email Updates

Keep up to date with Agilent's free Email Updates. As a subscriber, you will receive regular, customized email updates on the topics you select. Updates cover support, products and services, applications, promotions, events, and other areas. It is easy to unsubscribe or change your preferences. Subscribe today: http://www.agilent.com/find/emailupdates.

Agilent is committed to respecting and protecting your privacy. Our Privacy Statement at http://www.agilent.com/go/privacy describes our commitment to you. Please direct any questions about Agilent's privacy program to privacy_advocate@agilent.com.

For more information, please visit us at:

www.agilent.com

By internet, phone, or fax, get assistance with all your test & measurement needs

Online assistance:
www.agilent.com/find/assist

Phone or Fax
United States:
(tel) 1 800 452 4844

Canada:
(tel) 1 877 894 4414
(fax) (905) 206 4120

Europe:
(tel) (31 20) 547 2000

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Latin America:
(tel) (305) 267 4245
(fax) (305) 267 4286

Australia:
(tel) 1 800 629 485
(fax) (61 3) 9272 0749

New Zealand:
(tel) 0 800 738 378
(fax) 64 4 495 8950

Asia Pacific:
(tel) (852) 3197 7777
(fax) (852) 2506 9284

Product specifications and descriptions in this document subject to change without notice.

Printed in Germany May 22nd 2002
5988-3395EN

Agilent Technologies