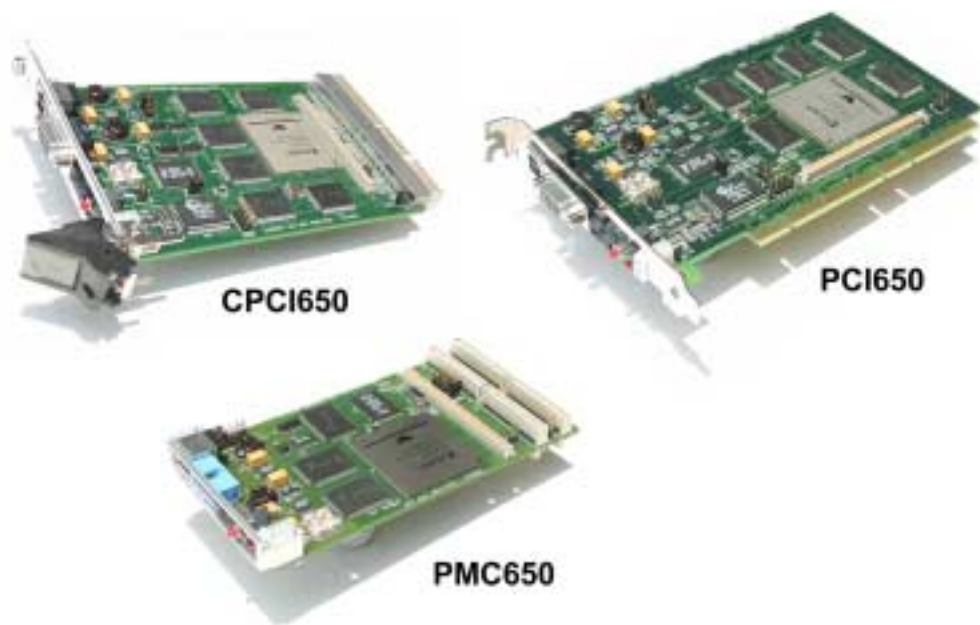


PCI650 Bus Analyzer
PMC650 Bus Analyzer
CPCI650 Bus Analyzer



U s e r ' s M a n u a l

Silicon Control
1020 Milwaukee Ave. Suite 305
Deerfield, IL 60015

October 2002
Revision 2.3

Silicon Control does not warrant the operation of the program will be uninterrupted or error free. In no event, will Silicon Control be liable for any damages including loss of data, lost profits, or cost of other incidental, consequential, or indirect damages arising from the use of this program or the accompanying documentation.

This Manual
© SILICON CONTROL INC.
ALL RIGHTS RESERVED.

No portion of this document may be copied without the written consent of Silicon Control, Inc. This program and documentation are subject to the copyright protection laws. The information in this document is subject to change without notice.

**Silicon Control Inc.
Software License Agreement**

**AnalyzeIt! Windows Graphical User Interface Program
for Silicon Control PCI650 System Analyzer**

Please read and be aware of the items listed in the following agreement. If you do not approve of the agreement, please return the complete package to the point of purchase for a complete refund.

1. USAGE RIGHTS

- a. Your rights with respect to the Program are non-exclusive.
- b. The Program can only be used by one user, on one computer at a time.
- c. The Program can be transferred to another computer as long as the requirements of item (b.) are satisfied.
- d. The Program and its documentation must not be distributed to others.
- e. Do not alter or modify the Program without prior consent of Silicon Control Inc.

2. BACKUP COPIES

- a. You may make as many backup copies of the Program as you like.
- b. The Silicon Control Inc. Software copyright notice must be included on each backup copy.

3. COPYRIGHTS

- a. The Program is copyrighted.
- b. The Program documentation is copyrighted.
- c. You may only copy the Program and program documentation for backup or to load the Program into your computer as part of program execution.

4. TERM OF LICENSE

- a. The Software License Agreement is effective until terminated.
- b. Terminate the Software License Agreement by destroying the Program, the documentation, and all backup copies.

5. LIMITED WARRANTY

- a. This Program is provided without any warranty of any kind.
- b. You bear the risk as to performance and suitability of the Program.
- c. Silicon Control Inc. does not warrant or guarantee the correctness, accuracy, completeness, or reliability of the Program.
- d. Silicon Control Inc. warrants the diskettes on which the Program is provided and any lock device provided to be free from manufacturers defects under normal use for a period of 90 days from the date of purchase.
- e. If the diskettes or lock device fail due to neglect, accident, or abuse, Silicon Control Inc. shall not be liable to replace the diskettes or lock device under this limited warranty.
- f. This limited warranty gives you specific legal rights. You may have additional rights depending on the state and/or country in which you live.
- g. Neither Silicon Control Inc. nor anyone involved in the development, manufacturing, or distribution of the Program or lock device shall be liable for any damages from the use, results of use, or inability to use the Program or lock device, even if Silicon Control Inc. has been notified of the probability of such damages or claims. Some states (countries) do not allow liability limitations for consequential or incidental damages or claims. This item may not be applicable.

6. LAWS

- a. This Software License Agreement shall be governed by the laws of the State of Illinois, United States of America.

7. ACCEPTANCE OF AGREEMENT

- a. You acknowledge that you have read, understand, and agree to abide by this Software License Agreement.
- b. You agree that this Software License Agreement is the complete agreement between Silicon Control Inc. and you.
- c. You agree that this Software License Agreement supersedes any and all prior agreements, written or verbal, between Silicon Control Inc. and you.

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION

- 1.1 FEATURES*
- 1.2 QUICK START*

CHAPTER 2. INSTALLATION

- 2.1. HARDWARE*
- 2.2. SOFTWARE*

CHAPTER 3. CAPTURING BUS ACTIVITY

- 3.1 STATE DISPLAY*
- 3.2 WAVEFORM DISPLAY*
- 3.3 SIGNAL PROPERTIES*
- 3.4 STATUS BAR*
- 3.5 SETUPS*
- 3.6 TIME MARKERS*
- 3.7 SEARCHING AND JUMPING*
- 3.8 SAVING SETUPS*
- 3.9 SAVING AND LOADING CAPTURED DATA*
- 3.10 PRINTING*

CHAPTER 4. PERFORMANCE ANALYSIS

- 4.1 PERFORMANCE DISPLAY*
- 4.2 SETUP*
- 4.3 UTILIZATION*
- 4.4 TRANSFER RATE*
- 4.5 LATENCY*
- 4.6 BURST DISTRIBUTION*
- 4.7 STATISTICS*
- 4.8 SAVING AND LOADING DATA*

CHAPTER 5. MASTER

- 5.1 DATA DISPLAY*
- 5.2 ADDRESS*
- 5.3 DATA*
- 5.4 COMMANDS*
- 5.5 SAVING AND LOADING DATA*
- 5.6 OPTIONS*

CHAPTER 6. STIMULUS

- 6.1 STIMULUS CONDITIONS*
- 6.2 STIMULUS CONTROL*
- 6.3 SAVING AND LOADING STIMULUS*
- 6.4 OPTIONS*

CHAPTER 7. CONFIGURATION SCANNING 

7.1 CONFIGURATION HEADER

7.2 CONFIGURATION ANALYSIS

7.3 FINDING DEVICES

7.4 SAVING AND LOADING CONFIGURATION INFORMATION

7.5 READING AND WRITING CONFIGURATION

7.6 OPTIONS

CHAPTER 8. PROTOCOL CHECKING 

8.1 ANOMALY CHECKLIST

8.2 TEST RESULTS

8.3 SAVING AND LOADING RESULTS

8.4 TRACE CONTROL

8.5 OPTIONS

CHAPTER 9. BACKPLANE TEST 

9.1 SHORT TEST

9.2 DRIVE TEST

CHAPTER 10. UTILITIES

10.1 SETUP

10.2 DOWNLOAD FIRMWARE

10.3 BOARD INFORMATION

CHAPTER 11. MISCELLANEOUS

11.1 ONLINE HELP

11.2 EXPANSION CONNECTOR

APPENDIX A – ANALYZER SPECIFICATIONS

APPENDIX B – API INTERFACE DOCUMENT

CHAPTER 1 INTRODUCTION

Silicon Control Inc. introduces the ultimate analyzer and exerciser for PCI, PMC and Compact PCI systems. The 650 family of analyzers represent our 3rd generation PCI analyzer combining high performance hardware with a sophisticated and intuitive software interface. The result is a powerful diagnostic tool for bus analysis - all on a single plug-in card!

1.1. FEATURES

The PCI650 analyzer provides a multitude of functions to help you analyze your system.

Analyzer

- ❑ 0 to 66 Mhz
- ❑ 32 and 64 bit
- ❑ 16, 32, 64, 128MB Trace
- ❑ Complex Triggering
- ❑ Trace Qualification

Exerciser

- ❑ Memory, I/O, Configuration
- ❑ 32 and 64 bit
- ❑ Read, Write, Test, Compare

Stimulus

- ❑ Hardware Simulation
- ❑ Pattern Generation

Protocol Violation Checker

- ❑ Detects >50 Protocol Violations

Timing Violation Checker

- ❑ Checks Unstable Signals
- ❑ Setup and Hold Verification
- ❑ Glitch Detection

Performance Analysis

- ❑ Bus Utilization
- ❑ Transfer Rate
- ❑ Latency
- ❑ Burst Distribution
- ❑ Statistics

Expansion Connector

- ❑ 500 Mhz Timing Module (optional)
- ❑ Board Test Module (optional)

1.2 QUICK START

This chapter provides a quick guide to installing and operating the analyzer with the AnalyzeIt! Windows software.

1.2.1. Hardware Installation

Insert the analyzer into an empty bus slot. Connect either the included RS232 or USB cable between the analyzer and a PC. If you are using the USB interface, the message “New Hardware Found” may appear after you connect the USB cable and apply power to the analyzer. If this message does not appear go to the Control Panel and Add New Hardware to install the drivers. The drivers are included on the CD ROM.

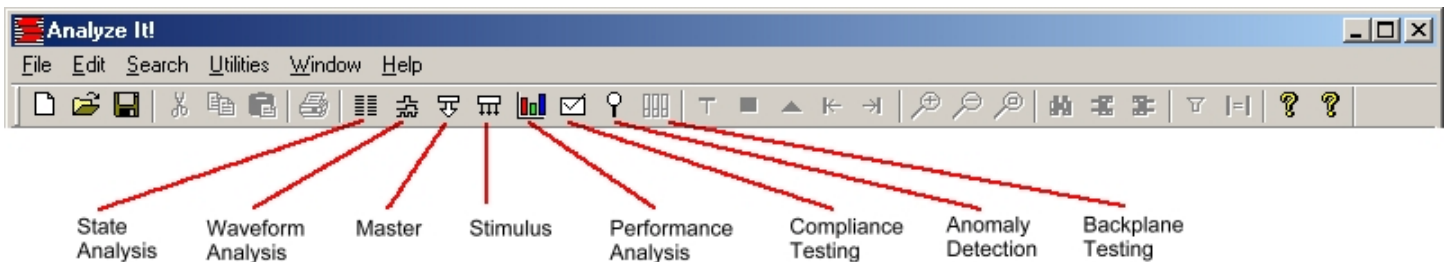
1.2.2. Software Installation

Install the AnalyzeIt! Software using the included CD or download the software from the Silicon Control web site at www.silicon-control.com.




1.2.3 Main Menu

Run the AnalyzeIt! Software. When using the RS232 port select Utilities then Setup to enter the COM port and baud rate. If the software cannot communicate with the analyzer it will enter a Demo mode.

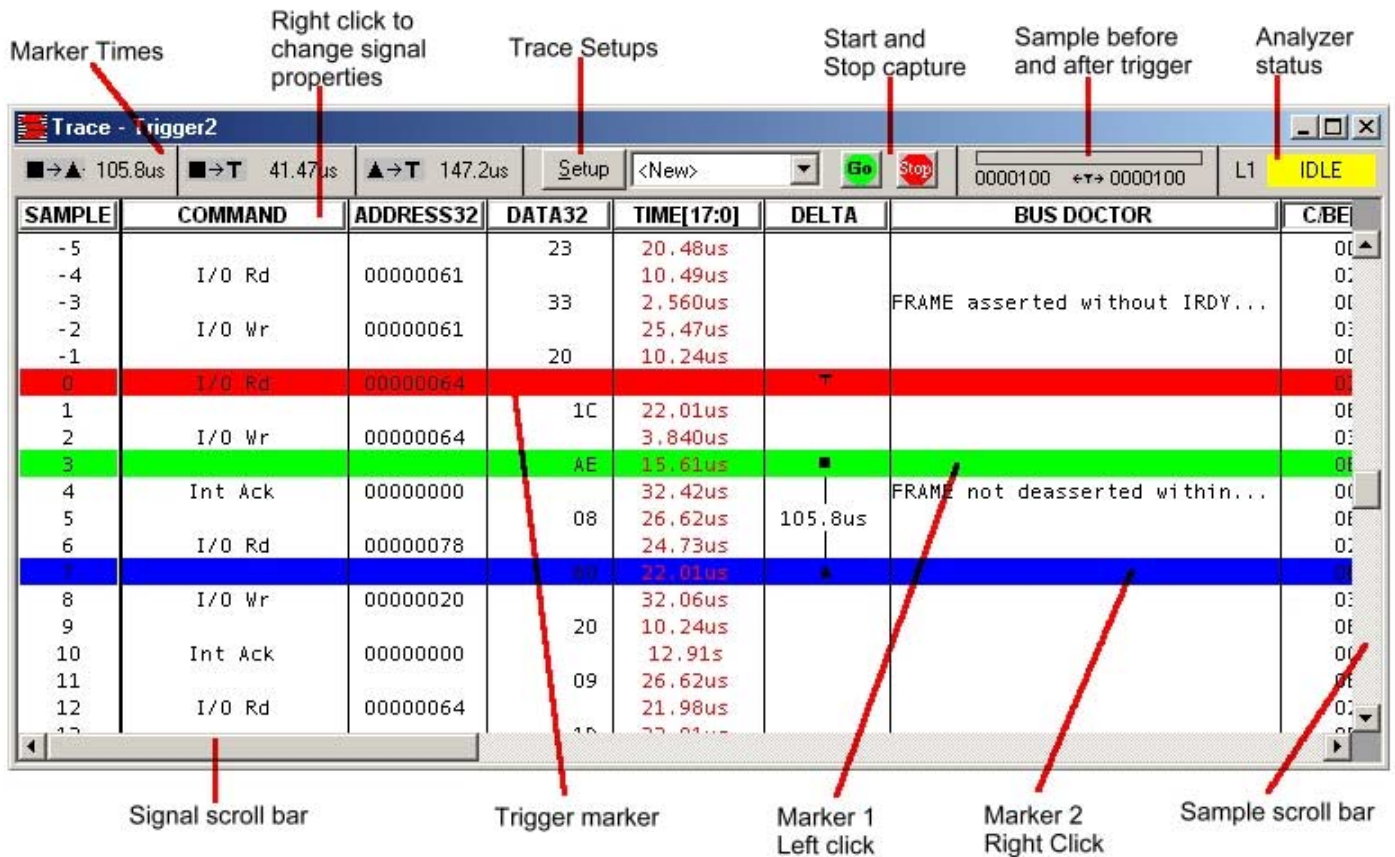
To start a new function, select **File** then **New** or click the function icons on the tool bar.



1.2.4. Capturing Bus Activity

To start capturing and viewing bus activity, select **File** in the main menu then **New**, **Capture**, and **State** or click the state display icon.  A blank state display appears. To start capturing activity, click the GO icon . A control bar indicates the buffer and trigger status and the samples before and after the trigger. Captured data fills the screen when the trace buffer fills or the STOP icon  is clicked.

1.2.5 State Display



1.2.6. Viewing Bus Activity


Use the sample scroll bar to view more data and the signal scroll bar to view more signals. The window may be resized to view more information. Add time markers by clicking directly in the data window. Left click for one marker, right click for another.

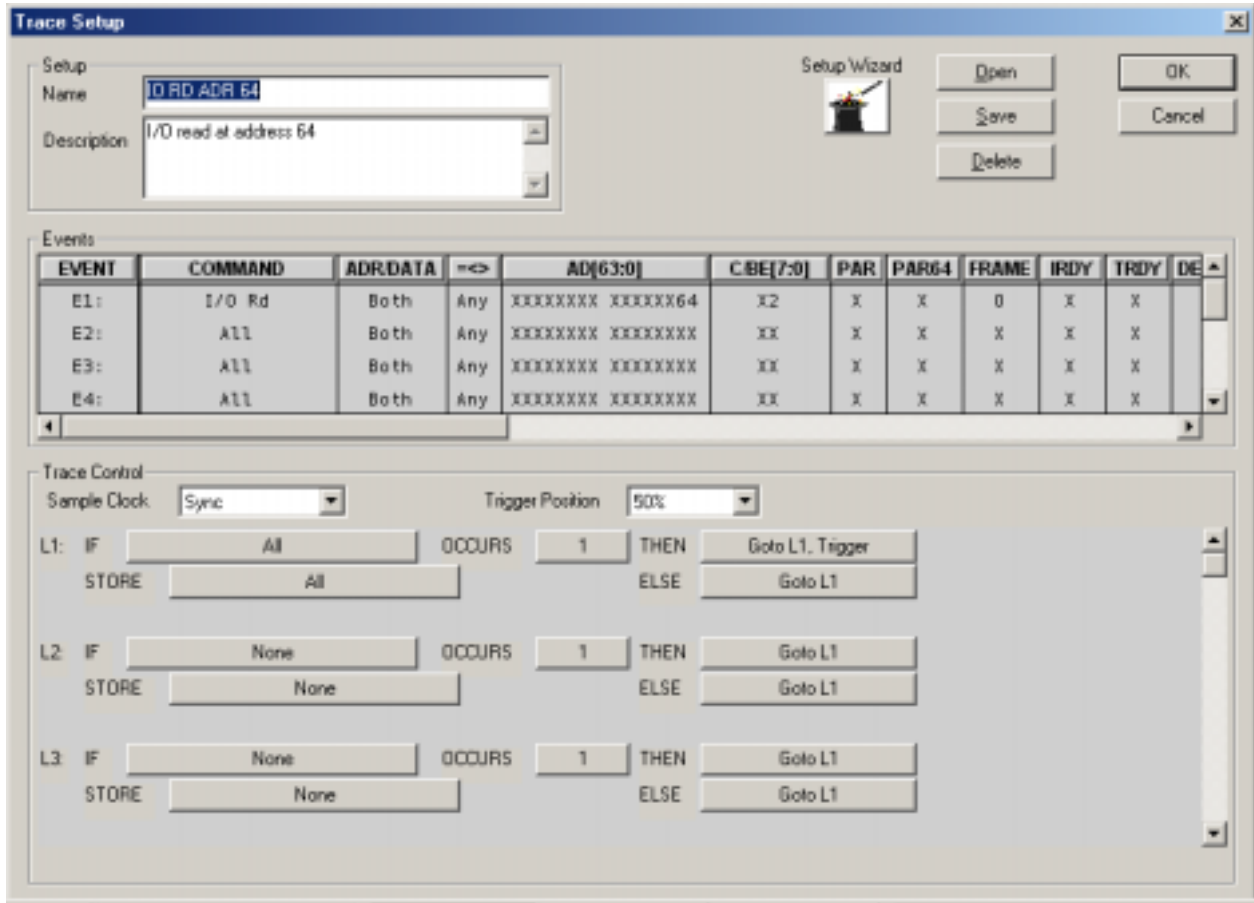
1.2.7. Changing the Display

Signals in the state display can be moved and resized. Click on the signal name and drag it to another position. Move the cursor over the edge of the signal box and drag the edge to resize it.

Right clicking on a signal name changes its properties. You can change the color and base, insert and delete signals, expand and collapse grouped signals and change the range of signals in a group.

1.2.8. Trace Setup

To control when and what bus activity to capture, click on the setup button . The setup window is divided into Event and Trace Control sections. Events specify what to look for on the bus and are used in the trace controls.



Events

To specify an event condition, click on the signal fields in the event section. An “X” (Don’t Care) means that the signal will not be used. Some fields such as COMMAND open a dialog box while others toggle through predefined states each time the field is clicked.


Trace Control

Trace controls specify how the analyzer captures bus activity. This multi-level structure provides a flexible way to setup simple or complex trace control.

Setups can be named, given a description, saved and used again. The Setup Wizard helps perform typical setups by asking a series of questions. Setup fields are filled in based on the answers to these questions.

1.2.9 Examples

Example – Trigger on a Memory Read at address 100 hex.

1. Click the Setup button  in a state or waveform window.

2. On the event labeled E1 set the fields as shown below:

EVENT	COMMAND	ADR/DATA	=<>	AD[63:0]
E1:	Mem Rd	Address	=	XXXXXXXX XXXXX100


3. On the trace control labeled L1 set the fields as shown below:

Sample Clock Trigger Position

L1: IF OCCURS THEN

STORE ELSE

4. Click the OK button  to send the setup to the analyzer.

5. Click the GO button  in the state or waveform window to start capturing data. Captured data will be displayed around the trigger sample.

1.2.7. Other Functions

To create other windows go to File and then New on the main menu or click on the function's icon on the main tool bar.

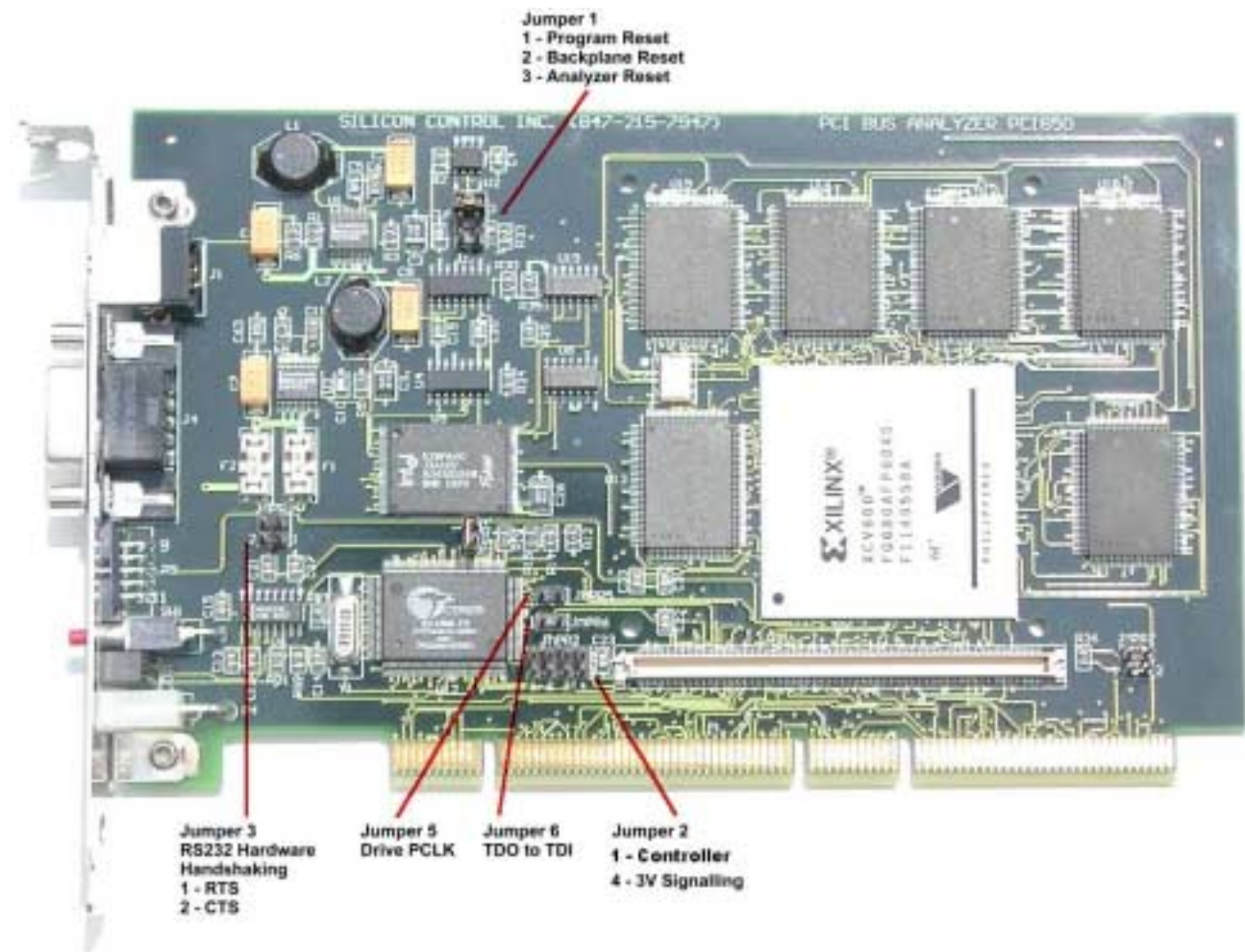
- ❑ The waveform display represents captured data in a graphical display and has many of the same features as the state display.
- ❑ Use the Master and Stimulus functions to transfer data onto the bus.
- ❑ Performance analysis includes 5 functions for measuring Bus Utilization, Transfer Rate, Latency, Burst Distribution and Statistics.
- ❑ Anomaly detection checks for protocol and timing violations and can be used to trigger and qualify captured trace data.
- ❑ Configuration scanning identifies and analyzes devices in a system.
- ❑ Compliance Testing is not supported on the 650 models.

**CHAPTER 2
INSTALLATION**

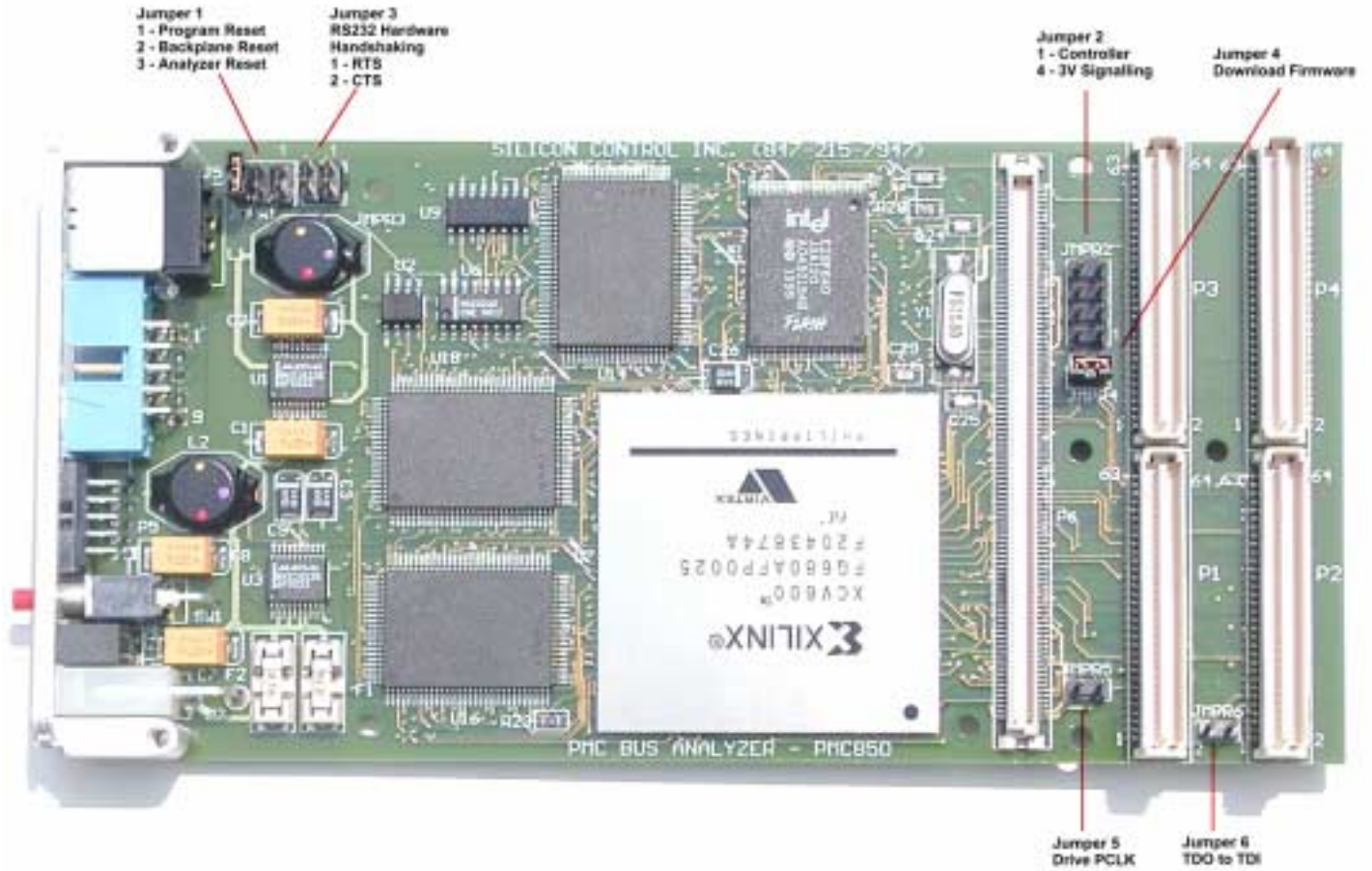
2.1 HARDWARE INSTALLATION

2.1.1 Jumpers

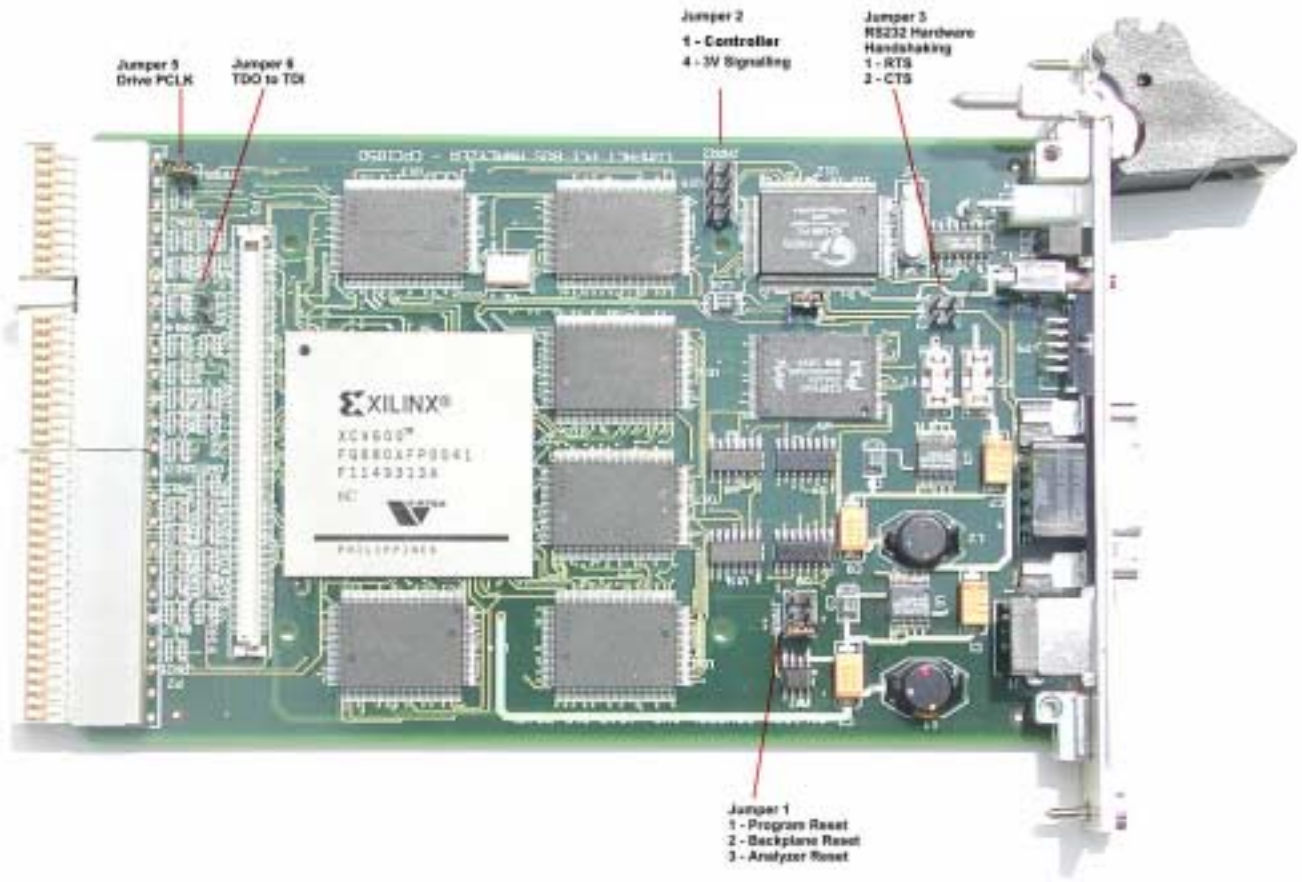
Jumpers on the analyzer provide configuration options for setting interfaces, hardware handshaking, reset control, driving the system clock and jumping the JTAG signals TDI to TDO. The location of these jumpers is shown below.



PCI650 Analyzer Jumpers



PMC650 Analyzer Jumpers



CPCI650 Analyzer Jumpers

Jumper 1 – Reset Jumper

<u>Position</u>	<u>Description</u>
1	User Controlled Backplane Reset –Generates a system reset through the AnalyzeIt! Software.
2	Power Up and Pushbutton System Reset – Generates a system reset at power up or when the pushbutton is pressed.
3 (default)	Power Up and Pushbutton Analyzer Reset – Generates an analyzer reset at power up or when the pushbutton is pressed.

Jumper 2 – Baud Rate and 3.3V Signaling

<u>Position</u>	<u>Description</u>
1	Controller – Insert this jumper when the analyzer is a controller. The GNT and IDSEL signals are driven. Also use Jumper 5 to drive the system clock.
4	3.3V Signaling – Insert this jumper when the analyzer is being installed in a 3.3V system.

Jumper 3 – RS232 Hardware Handshaking

<u>Position</u>	<u>Description</u>
1	RTS control
2	CTS control

Jumper 5 – Drive PCLK

Installing this jumper enables the on board analyzer clock generator to drive the system clock. The frequency of the clock generator is specified by the AnalyzeIt! Software.

Jumper 6 – TDI to TDO

Installing this jumper connects the JTAG signals TDI to TDO to connect this daisy-chained signal.

2.1.2. Front Panel



USB	A standard Type B connector operating at 12Mb/sec.
RS232	A female DB9 connector operating at up to 57.6K baud with RTS and CTS hardware handshaking.
Trigger IN/OUT	A 10 pin shrouded male IDC connector on 2mm centers with a keyed slot. The signals include 1 ground, 1 trigger output and 8 trigger inputs.
Pushbutton Reset	Resets the analyzer / system based on Jumper 1 settings.
LEDs	The green LED illuminates after a successful power up and self-test diagnostics. The red LED illuminates for approximately 1 second at power up and indicates that hardware is being configured. If the red LED stays on an error occurred during configuration.
External Power	A 2-pin power connector provides external power to the analyzer. The power supply voltage range must be between 4.5V and 5.5V and capable of supplying a minimum current of 2.8 Amps.

Important: When supplying external power the F1 fuse must be removed and the F2 fuse installed. Carefully remove a fuse from the fuse holder with a pliers or similar device.

2.1.3. Cables

USB – An 8 foot USB cable is included with a Type A connector on the PC side and a Type B connector on the analyzer side.

RS232 – An 8 foot RS232 cable is included with a DB9 female connector on the PC side and a DB9 male connector on the analyzer side.

Trigger – A 10 pin trigger input/output ribbon cable is included with a 10-pin IDC connector on the analyzer side and individual test clips on the other side. The test clips are color coded as follows:

<i>Signal</i>	<i>Color</i>
Ground	Black
Trigger Out	Yellow
Trigger In (8)	Red

Power – This cable provides external power to the analyzer and consists of a 2 wire molded power connector with un-terminated red and black heavy gauge wires. The red wire must be connected to a voltage source between 4.5V and 5.5V with a minimum current capability of 2.8A. The black wire connects to ground.

Important: When supplying external power the F1 fuse must be removed and the F2 fuse installed. Carefully remove a fuse from the fuse holder with a pliers or similar device.

2.1. SOFTWARE INSTALLATION

2.2.1. PC Requirements

The system requirements for a PC running the AnalyzeIt! Software is as follows:

<u>Minimum</u>	<u>Recommended</u>
Pentium 3 Processor	Pentium 3 Processor at 600Mhz
2GB Hard Drive	5GB Hard Drive
64MB RAM	128K RAM
CD ROM Drive	CD ROM Drive
USB or Serial port	USB or Serial port
15" Monitor	17" Monitor
Mouse	Mouse

2.2.2 Software Installation

2.2.2.1 AnalyzeIt! Software

Install the AnalyzeIt! Software using the included CD or download the software from the Silicon Control web site at www.silicon-control.com.

After inserting the CD into the drive, browse for the file AnalyzeIt! The file is a self-extracting executable file. To install the software double click the file. Follow the installation instructions on the screen. You will have to restart your computer after installation.

2.2.2.2 USB Drivers

If you are using the USB interface, the message "New Hardware Found" may appear after you connect the USB cable and apply power to the analyzer. If this message does not appear go to the Control Panel and Add New Hardware to install the drivers. The drivers are included on the CD ROM.

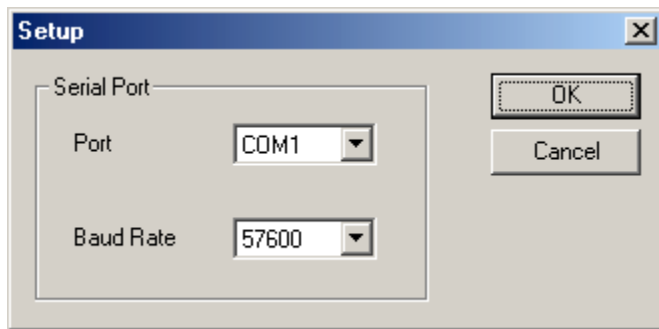
2.2.2.3 Software Updates

The latest up to date software is available on Silicon Control's web site at www.silicon-control.com. Download the AnalyzeIt! Software into a folder on your computer. The downloaded file is a self-extracting executable file. To install the software double click the file.

2.2.3. Initial Setup

Run the software by double clicking the AnalyzeIt! Icon on the desktop or go to Start – Programs – AnalyzeIt! The analyzer has both RS232 and USB interfaces. You must select which interface before using the analyzer. The AnalyzeIt software will come up in a demo mode if it cannot communicate with the analyzer. You can then set the interface you are using and restart the software. The new interface setting will be used the next time you start the software.

To configure the RS232 or USB port, click on **Utilities** in the Main Menu and select **Setup**.



For the RS232 interface, enter the COM port number and the baud rate you are using. The board rate of the analyzer will be automatically adjusted to match the software setting. Select USB if using the USB port.

Note: If you are having communication problems using the RS232 interface try lowering the baud rate. The speed of this interface may be influenced by the PC speed and cable type and length.

**CHAPTER 3
CAPTURING BUS ACTIVITY**

3.1 State Display

The state display presents captured bus activity in a column form and is best used when viewing data transfers. Signals can be moved, inserted, deleted and color and radix defined. Grouped signals such as address can be collapsed and expanded and their range specified.

To open a state display, select File - New or Open - Capture - State. You can also click on the State display icon in the main toolbar.



Annotations in the screenshot:

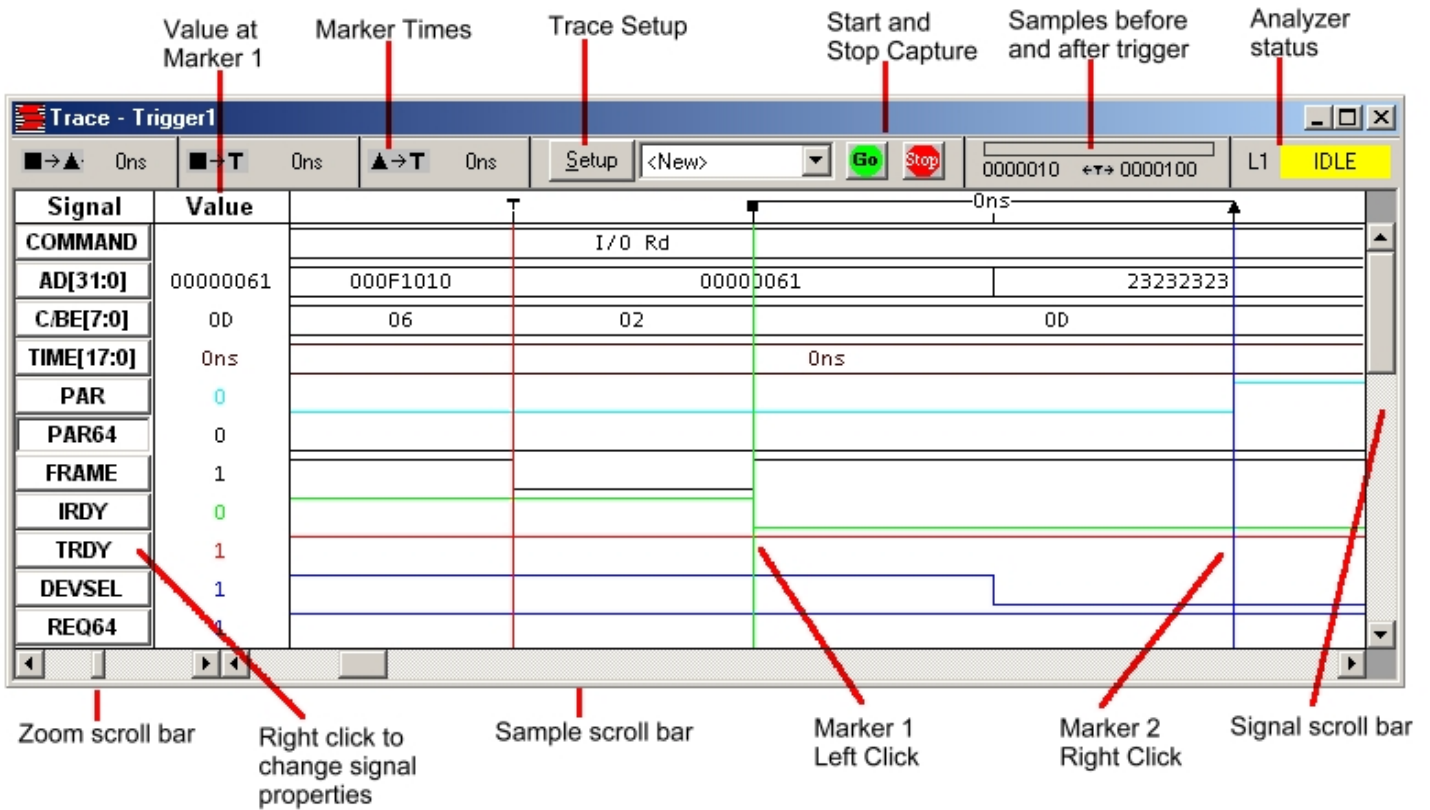
- Marker Times: 105.8us, 41.47us, 147.2us
- Right click to change signal properties
- Trace Setups
- Start and Stop capture
- Sample before and after trigger
- Analyzer status: L1 IDLE
- Signal scroll bar
- Trigger marker
- Marker 1 Left click
- Marker 2 Right Click
- Sample scroll bar

SAMPLE	COMMAND	ADDRESS32	DATA32	TIME[17:0]	DELTA	BUS DOCTOR	C/BE
-5			23	20.48us			0C
-4	I/O Rd	00000061		10.49us			0:
-3			33	2.560us		FRAME asserted without IRDY...	0:
-2	I/O Wr	00000061		25.47us			0:
-1			20	10.24us			0:
0	I/O Rd	00000064					0:
1			1C	22.01us			0E
2	I/O Wr	00000064		3.840us			0:
3			AE	15.61us			0:
4	Int Ack	00000000		32.42us		FRAME not deasserted within...	0:
5			08	26.62us	105.8us		0E
6	I/O Rd	00000078		24.73us			0:
7			80	22.01us			0:
8	I/O Wr	00000020		32.06us			0:
9			20	10.24us			0E
10	Int Ack	00000000		12.91s			0:
11			09	26.62us			0:
12	I/O Rd	00000064		21.98us			0:

3.2 Waveform Display

The timing display represents trace data as waveforms for timing analysis. A numeric value is provided on top of each waveform. As in the state display, signals can be moved, inserted, deleted and color and radix defined. Grouped signals such as address can be collapsed and expanded and their range specified. A zoom is provided to get a better look at waveform relationships.

To open a waveform display, select File - New or Open - Capture - Waveform. You can also click on the Waveform display icon in the main toolbar.



3.3 Signal Properties

Signals in the state and waveform display have many properties and functions.

Moving – Click on a signal and drag it to another position

Sizing – Drag the edge of a signal box to size the signal field

Color – Right click on the signal name and select color to change the color of the data displayed

Base - Right click on the signal name and select base (Hex, Octal, Binary) to change the radix

Change - Right click on the signal name and select Change Signal to change to another signal

Expand - Right click on the signal name and select Expand to expand a bussed signal into individual signals.

Collapse – Select a number of signals with the same bus name (i.e. AD[0], AD[1], AD[2]). Two methods to select multiple signals are:

1. Hold the Ctrl key with the mouse to select more than one signal
2. To select a range of signals, click on one signal then hold the Shift key and select another signal

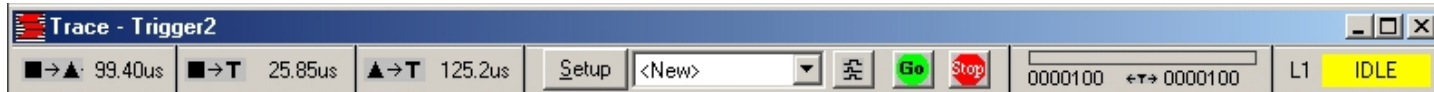
Right click on one of the selected signal names and select Collapse to collapse them into a single bussed signal.

Insert - Right click on the signal name and select Insert to insert a new signal

Delete - Right click on the signal name and select Delete to remove the signal

Insert Blank - Right click on the signal name and select Insert Blank to insert a blank field. This is useful for separating groups of signals for better clarity.

3.4 Control Bar



The control bar provides information and control while capturing and displaying trace data.

Go Button – Starts the capture of bus activity.

Stop Button – Stops the capture of bus activity and displays captured trace data. A Stop occurs automatically if the trace buffer fills.

Number of Samples Before and After the Trigger – When capturing bus activity these values are constantly updated to reflect the amount of data stored in the trace buffer. When reading trace data this display indicates your position within the trace buffer.

Analyzer Status – Indicates the current state of the analyzer.

IDLE – not capturing or reading trace data

ACTIVE – capturing trace data

READING – reading trace data

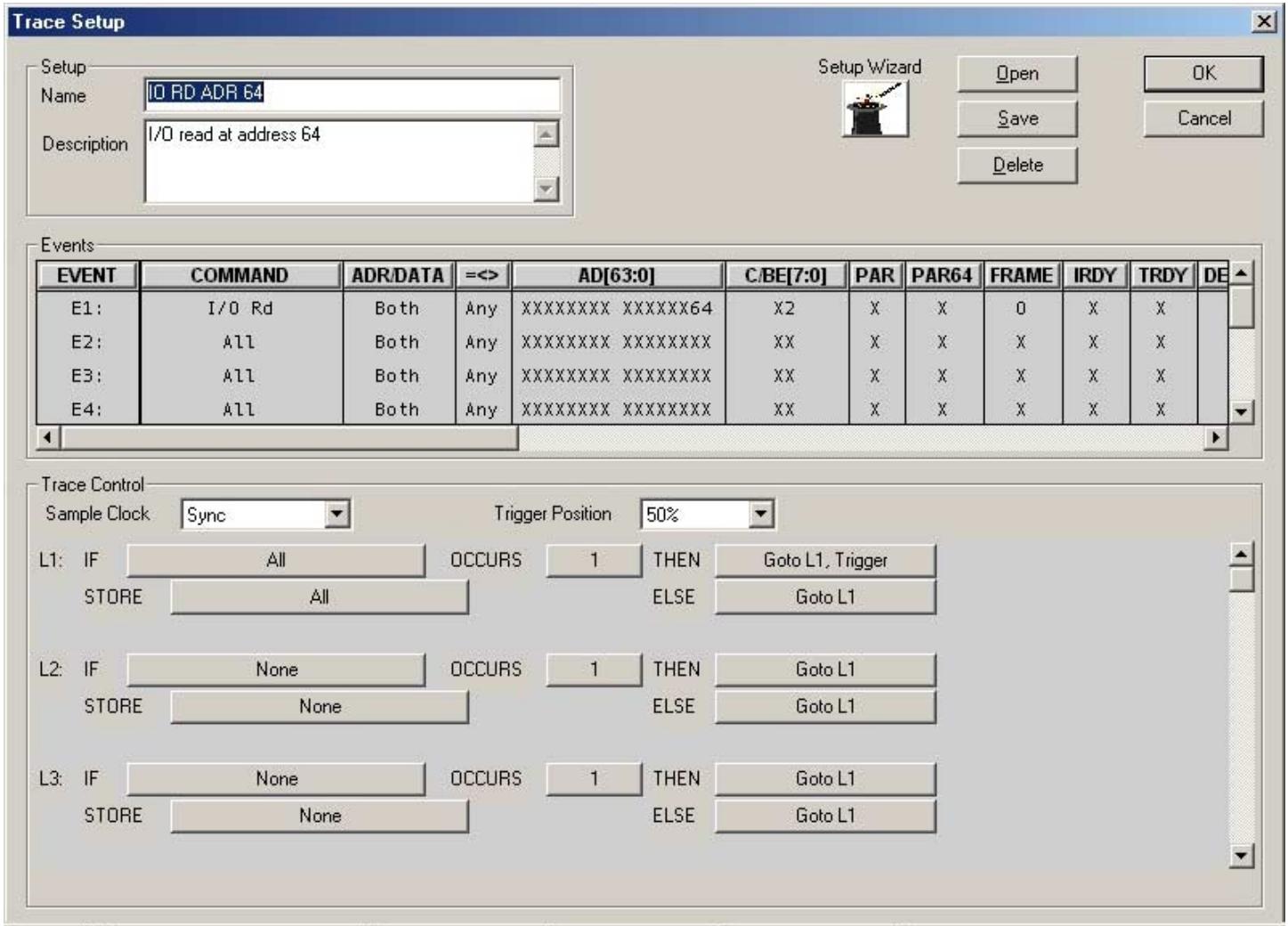
Setup Button and Drop Down List – Click the Setup button to modify a new or saved setup. The drop down list is a quick way to select previously saved setups.

State/Waveform Button – This button changes the window between state and waveform displays. It also allows you to open another state or waveform display, which contains the same data as the first display and whose position is linked together.

Marker Time Display – The time from the Trigger to Marker1, Trigger to Marker 2 and Marker1 to Marker 2 is displayed.

3.5 Setups

The type of information captured in the trace buffer is controlled by setup information. Clicking the Setup button in the control bar allows you to specify this information.



The setup window is divided into 3 main sections: Setup Utilities, Events, and Trace Control.

Setup Utilities – These are general setup functions:

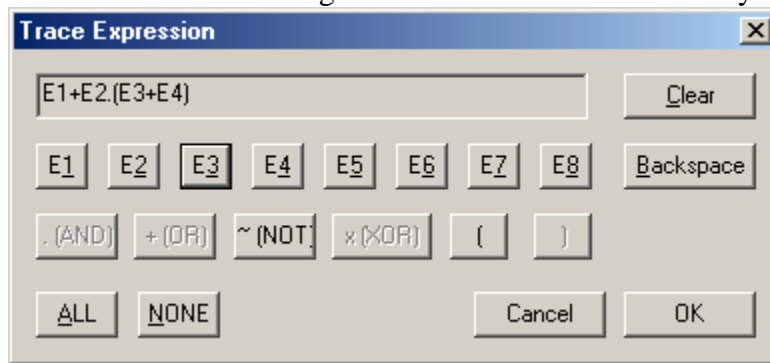
1. Specify a setup Name
2. Give a setup Description
3. Open, Save and Delete setups
4. Use the Setup Wizard to assist in defining setups

Events – Events are bus conditions that are of interest to you. They are used in the trace controls for triggering, trace qualification and decision-making. Each event specifies a complete set of bus signals and their state. Click on a field in the event section to change a signal. Grouped signals such as AD[63:0] open a dialog box while individual signals rotate through preset selections when clicked. The signal fields can be moved, sized and changed to customize the event display. See Signal Properties in section 3.3 for a complete list of signal properties.

Trace Control – Trace controls specify how the analyzer captures bus activity. This multi-level structure provides a flexible way to setup simple or complex trace control.

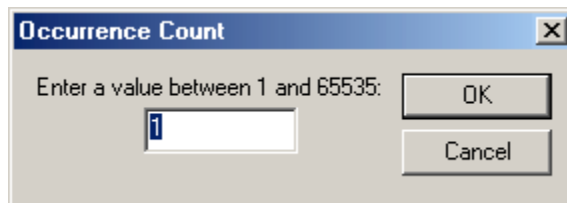
Each level has the following fields:

IF: This field defines an event or logical combination of events that you want to look for.

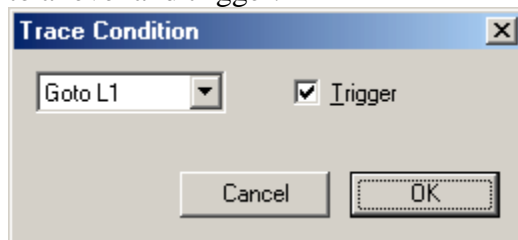


When you select the IF field an expression generator helps you construct an event condition.

OCCURS: The number of times the event occurs.



THEN: If the event in the IF field occurs the number of times in the OCCURS field you can jump to a level and trigger.

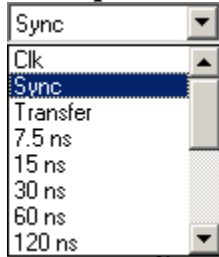


ELSE: If the event in the IF field DOES NOT occur you can jump to a level and trigger.

STORE: An event or logical combination of events specifies what to store in the trace buffer.

The trace control section also sets the sample clock and trigger position.

Sample Clock



The analyzer samples the bus using either the bus clock or a high precision analyzer clock.

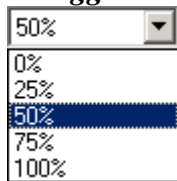
Clk samples on every positive edge of the system clock.

Sync samples on the system clock when address and data are valid.

Transfer samples on the system clock during a data transfer

The remaining selections are periodic intervals based on an analyzer clock.

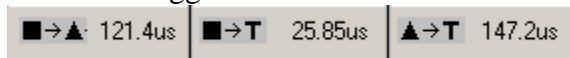
Trigger Position



The trigger position defines the amount of information stored in the trace buffer before and after a defined event called the trigger. Once a trigger is encountered a trigger position of 0% stores all events starting at the beginning of the trace buffer.

3.6 Time Markers

Time measurements on the state and waveform displays are performed using markers. Left click on data in the display window to place one marker. Right click to place another marker. The time between the markers is displayed on the data display. The control bar in the state and waveform window also shows the time from the trigger to each marker and from marker to marker.



3.7 Searching and Jumping


Searching – To find information in the trace buffer use the search features on the main tool bar. After defining search criteria you can search forward or backward through the trace buffer. The search starts at the 1st marker position.



Jumping – Use the jump features on the main tool bar to quickly move to specific samples in the trace buffer. Jump to the start, end, trigger, marker 1 and marker 2.



3.8 Saving Captured Data

To save trace data, select File and Save or Save As or click the Save icon  in the main tool bar. Trace data is saved in a text format that can be opened later or read in a word processor, database or spread sheet. The file name is given a .tra file extension.

3.9 Printing

To print state and waveform data select File and Print or click the print icon  in the main tool bar.

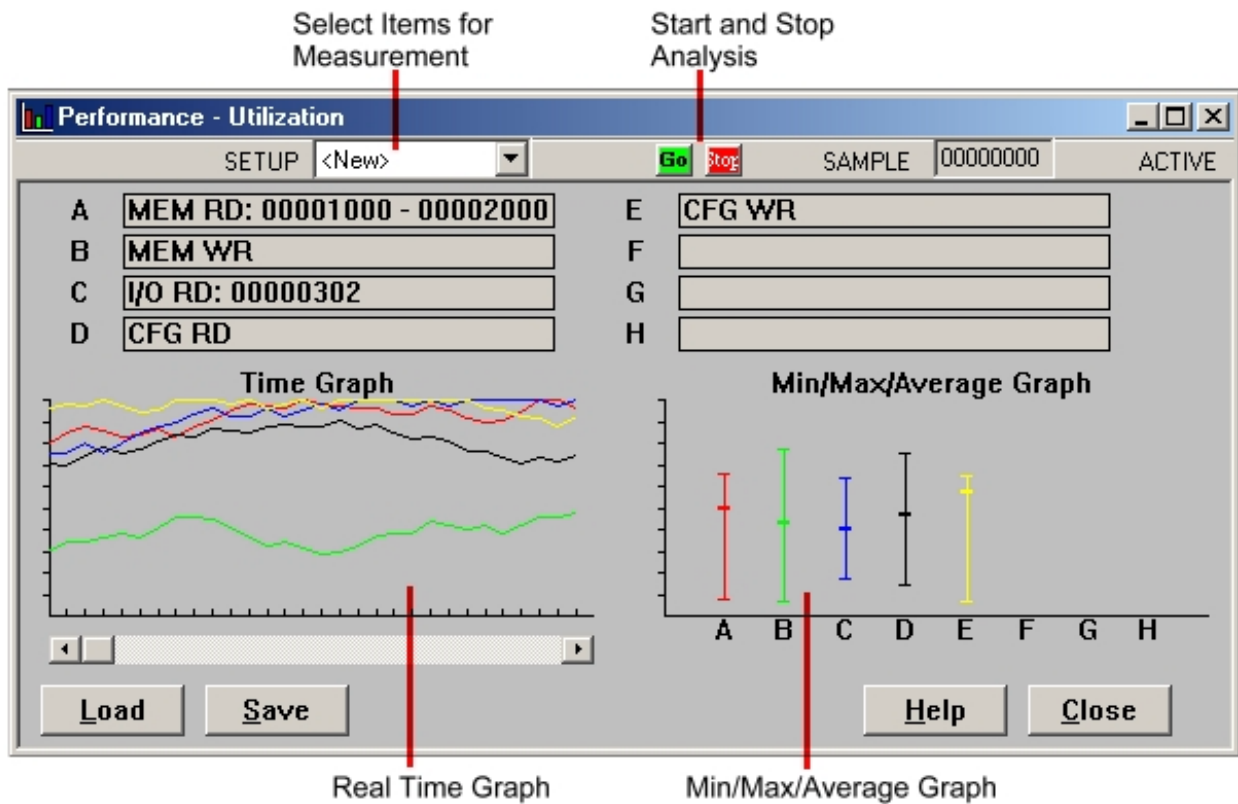
CHAPTER 4 PERFORMANCE ANALYSIS

Time graphs and histograms give an overview of system performance, bottlenecks and problem areas.

There are 5 performance analysis measurements:

1. Utilization - Computes the percentage of time a signal is active.
2. Transfer Rate - Computes the speed of signals in MB/sec.
3. Latency - Computes the delay of signals in microseconds.
4. Burst Distribution - Computes the amount of data transferred in a burst transfer.
5. Statistics - Counts the number of occurrences of selected signals.

Dedicated hardware counters accumulate occurrences of selected items. The counter data is plotted on a real time graph. The minimum, maximum and average values are computed from this data and displayed on a separate bar chart.



4.1 Performance Display

The performance display consists of a moving time graph and an average/min/max bar or pie chart. The time graph shows the current measurement and is updated at fixed time intervals specified by the user. The bar chart averages this moving data and tracks the min and max values. A scroll bar under the time graph lets you review the previous data.

4.2 Setup

Click the setup button to select signals for performance analysis. These can include bus signals, group signals and address ranges. To start the performance measurements click on GO icon.

4.3 Utilization

Utilization measurements compute the percentage of time a signal is active. This measurement identifies the resources that are using the bus. These can include address usage, command type and overall idle and busy times.

4.4 Transfer Rate

The transfer rate measurement measures in MB/s the speed of data transfers. These measurements can be specified for specific address ranges and cycles.

4.5 Latency

The efficiency of data transfers in a system depend upon the latencies involved. Measurements are taken of master, target, address, data and arbitration to provide a look at latencies that slow a bus down.

4.6 Burst Distribution

The amount of data transferred during each cycle is measured and displayed in 8 ranges. The ranges include single byte transfers and up.

4.7 Statistics

Statistics can be obtained for any bus signal or group of signals. For example the number of memory reads, memory writes, writes to an address or address range can be counted.

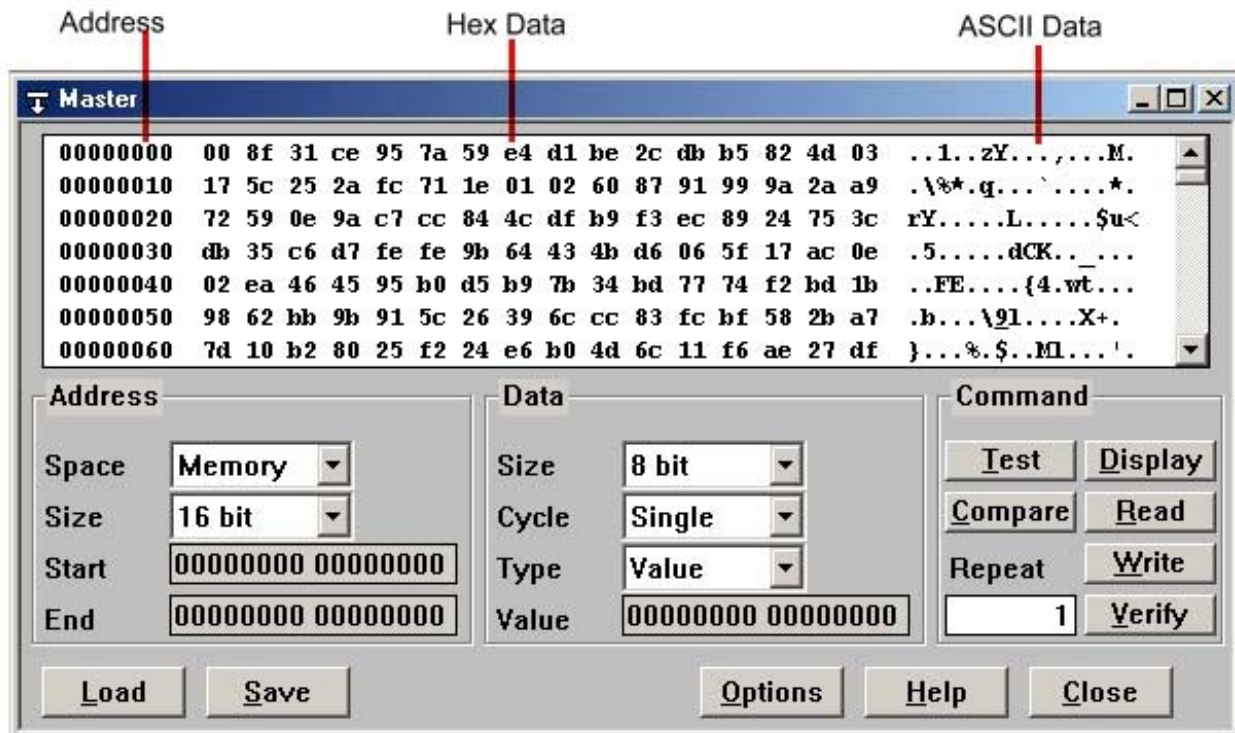
4.8 Saving and Loading Performance Data

Performance data can be saved to a file by clicking the Save button. To look at previously saved data click the Load button. Saved performance files have a .per extension.

**CHAPTER 5
MASTER**

The Master functions perform memory, I/O and configuration data transfers across the bus. The user has complete control over many aspects of the transfer including:

- Address: space, size and range
- Data: size, cycle, type and value
- Command: read, write, test, compare, verify



5.1 Data Display

The data display is divided into 3 sections:

- Address** – the starting address of the data line in hexadecimal.
- Hex Data** – the data read from the bus in hexadecimal, 16 bytes per line.
- ASCII Data** – the ASCII equivalent of the hex data, a period is a non-ASCII character.

5.2 Address Setup

The address setup section specifies the address used for read and writes. The fields are as follows:

- Space** – Memory, I/O, Configuration
- Size** – 8, 16, 32, 64 bit
- Start** – Starting address in hexadecimal.
- End** – Ending address in hexadecimal

5.3 Data Setup

The data setup section specifies the data used for reads and writes. The fields are as follows:

Size – 8, 16, 32, 64 bit

Cycle – single, burst

Single cycle transfers perform one read or write at a time through the address range. Arbitration takes place for each data transfer of the specified size.

Type – value, file, target

During a bus write command the data used is specified in the Type field.

Value - uses the data in the Value field.

File - uses data stored in a data file that may have been previously saved with the Save button in the Master window

Target – uses data in the target memory on the analyzer

Value – data value in hexadecimal

5.4 Commands

The commands used in the master functions are:

Read – reads data from a target on the bus

Write – writes data to a target on the bus

Verify – writes data to a target and reads it back for verification

Test – writes out test patterns and reads them back for verification

Compare – reads data and compares it with the data specified in the data setup

5.5 Saving and Loading Data

Read data and setup information can be saved to a file by clicking the Save button. To look at previously saved data click the Load button. Saved master files have a .mst extension.

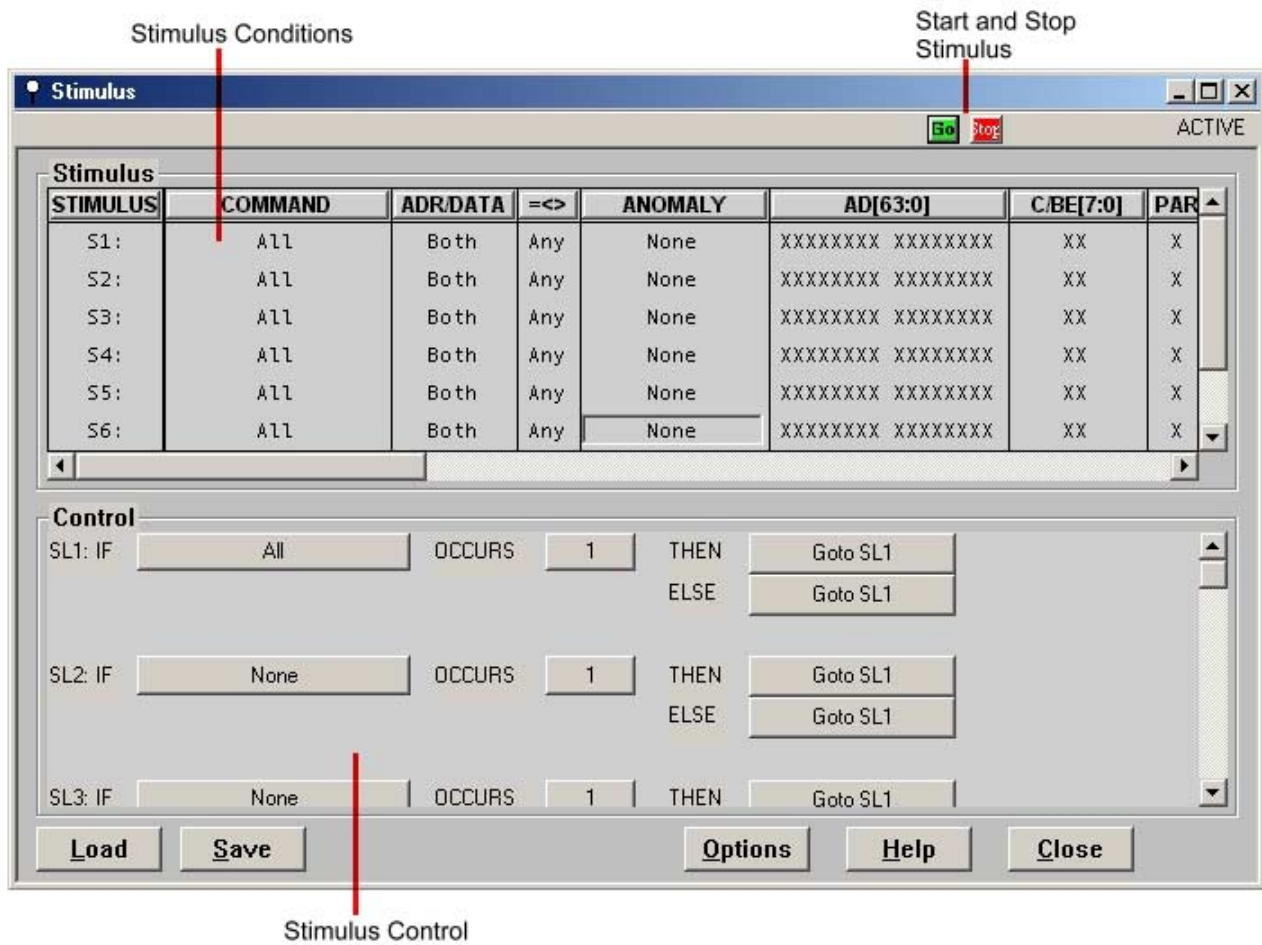
5.6 Options

No options are available.

**CHAPTER 6
STIMULUS**

The analyzer has the ability to generate signal patterns on the bus. These features provide hardware simulation and test system response to events.

Stimulus is driven onto the bus under control of a 16 level sequencer. At each level any combination of bus signals can be specified along with an activation time and wait time synchronized to the system clock. These levels can then be linked together to form complex stimuli. Initiation of this stimulus can be controlled manually or in response to a bus event.



6.1 Stimulus Conditions

Each stimulus condition specifies which signals drive the bus and their drive level. The drive states are:

- X – do not drive the bus
- 0 – drive the bus to a low state
- 1 – drive the bus to a high state

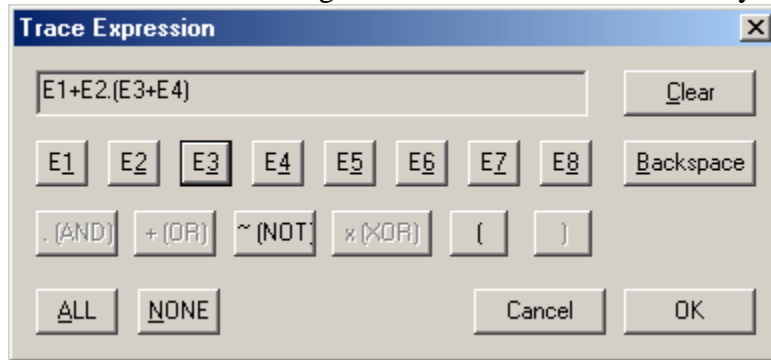
The stimulus signal fields can be moved, sized and changed to customize the stimulus display. See Signal Properties in section 3.3 for a complete list of signal properties.

6.2 Stimulus Control

Stimulus controls specify how the analyzer stimulates bus activity. This multi-level structure provides a flexible way to setup simple or complex stimulus control.

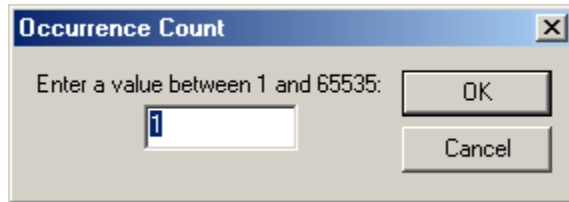
Each level has the following fields:

IF: This field defines an event or logical combination of events that you want to look for.

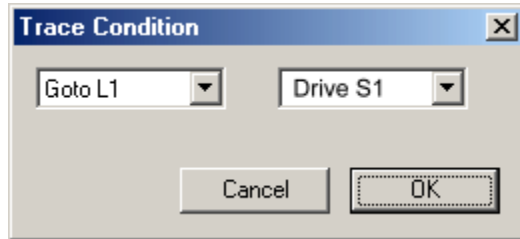


When you select the IF field an expression generator helps you construct an event condition.

OCCURS: The number of times the event occurs.



THEN: If the event in the IF field occurs the number of times in the OCCURS field you can jump to a level and drive a stimulus condition.



ELSE: If the event in the IF field DOES NOT occur you can jump to a level and drive a stimulus condition.

6.3 Saving and Loading Stimulus

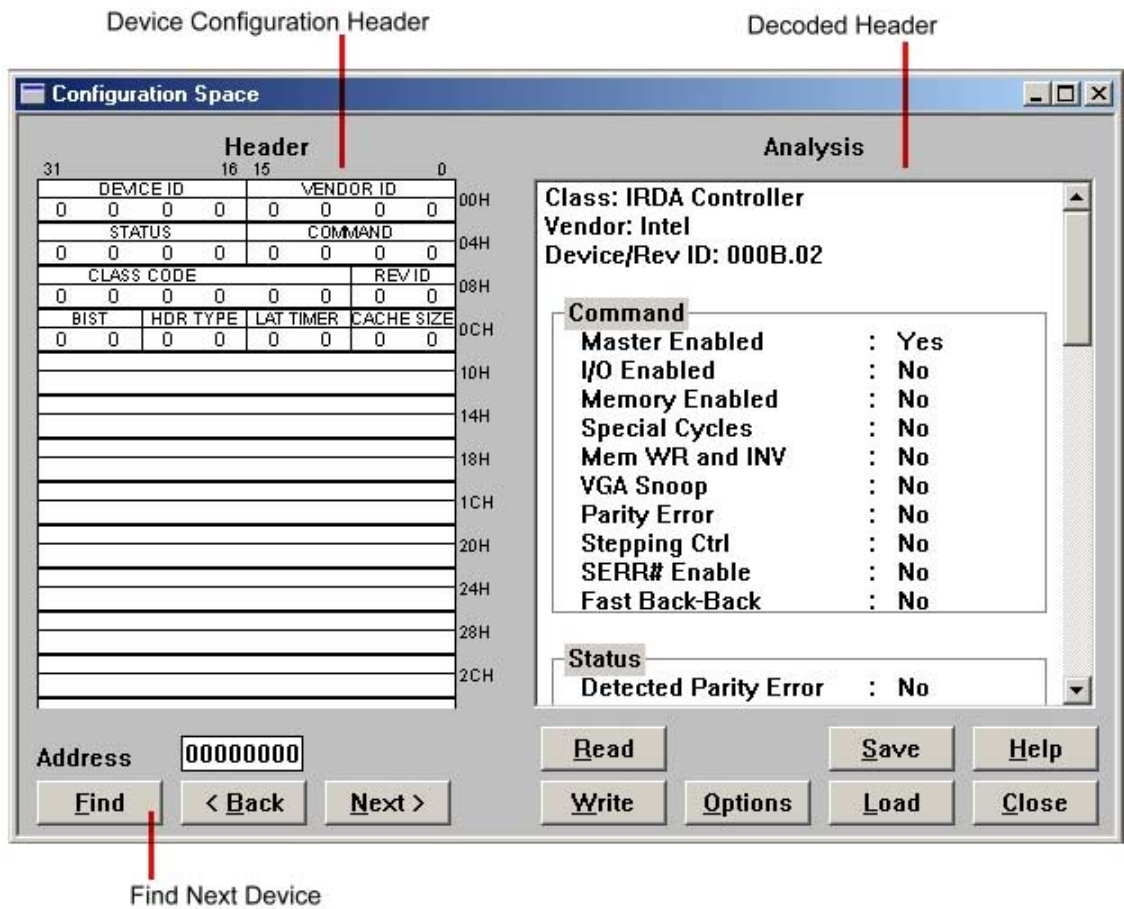
Stimulus conditions and controls can be saved to a file by clicking the Save button. To look at previously saved stimulus click the Load button. Saved stimulus files have a .stm extension.

6.4 Options

No options are available.

CHAPTER 7 CONFIGURATION SCANNING

An automatic scan of configuration space identifies the boards in a system and displays the configuration information. A search is performed of all configuration registers and the results are displayed and decoded for easy identification. The values of the configuration registers are displayed in one window and the decoded meaning in another window. Vendor names, type of board, version numbers, etc. are displayed for easy interpretation.



7.1 Configuration Header

The configuration header window displays binary configuration data. The first 16 bytes are the same for every device. The Header Type field, in the upper 16 bytes, define the format of the data that follows. The display automatically adjusts the format based on the Header Type field.

7.2 Configuration Analysis

The raw information in the configuration header is analyzed and the results are displayed in an analysis window. The device class and vendor ID is looked up and the other fields are decoded.

7.3 Finding Devices

Several buttons search for devices in the configuration space. The FIND button scans the configuration space to locate the next device. The NEXT and BACK buttons also locate devices before or after the current device.

7.4 Saving and Loading Configuration Information

Configuration information can be saved to a file by clicking the Save button. To look at previously saved configurations click the Load button. Saved configuration files have a .cfg extension.

7.5 Reading and Writing Configuration

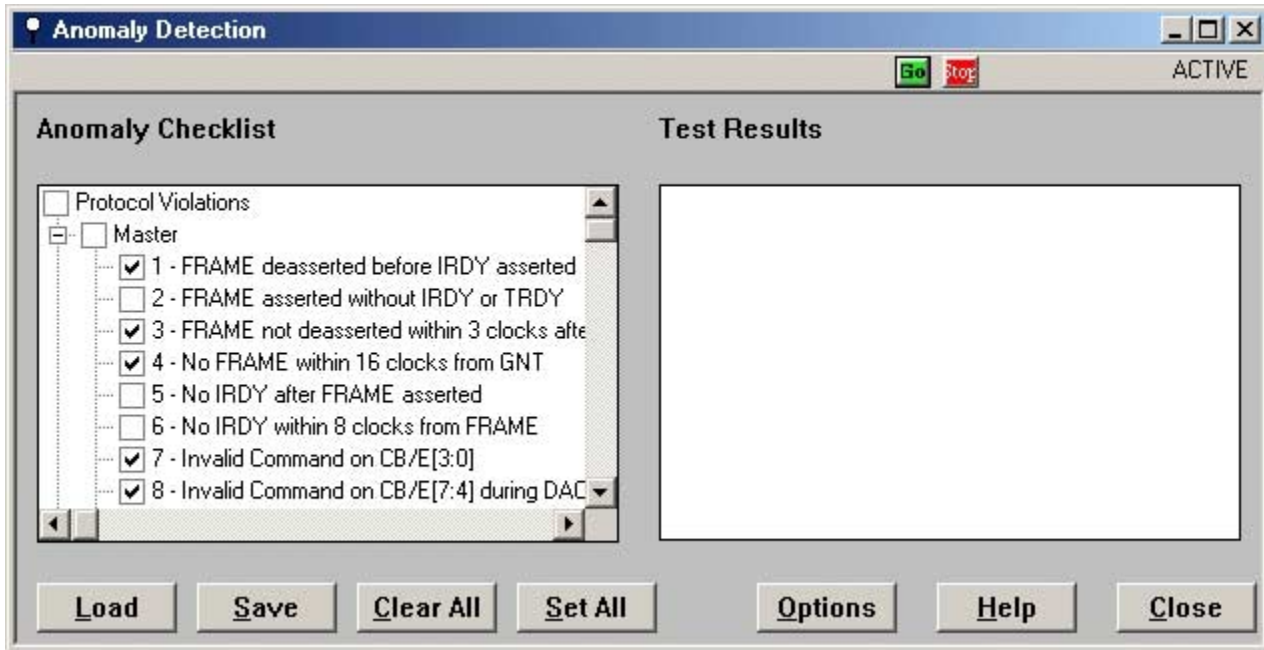
The READ and WRITE buttons provide a convenient way of modifying and displaying configuration space.

7.6 Options

No options are available

**CHAPTER 8
PROTOCOL CHECKING**

The analyzer continuously monitors and detects over 100 protocol and timing violations. Dedicated hardware checks setup and hold times, unstable signals and glitches on lines. Protocol checking screens for illegal signal assertions referenced to the PCI specification.



8.1 Anomaly Checklist

The anomaly checklist allows the selection of specific tests. A checklist tree structure groups anomalies into categories. Clicking the “+” sign next to a category expands the checklist into individual anomalies. Individual anomalies can be checked. Clicking the “-” sign next to a category compresses the checklist back into a single category. Checking a category selects all the anomalies under that selection. The Clear All button removes all the checkmarks while the Set All button marks every anomaly.

8.2 Test Results

Results of the anomaly checking are displayed in the test results window as the tests are being performed. A scroll bar allows the viewing of previous test results.

8.3 Saving Results

Anomaly test information can be saved to a file by clicking the Save button. To look at previously saved anomaly data click the Load button. Saved anomaly test files have an .adt extension.

8.4 Trace Controls

The trace controls of the state and waveform display incorporate the anomaly checklist. Anomalies can be used as in the trigger and trace qualification when capturing trace activity. An anomaly event field specifies the use of the anomalies selected in the checklist.

8.5 Options

No options are available.

CHAPTER 9 BACKPLANE TEST

The backplane test checks for shorts and the ability to drive signals high and low in an open backplane. This test will not run correctly if other boards are plugged into the backplane. The name of the signal is displayed as it is being tested. If a short or drive problem is encountered it is listed in a test results window.

9.1 Short Test

The short test checks for shorts between signals by pulsing each bus signal and monitoring every other signal for that pulse. If a pulse is detected on a signal not being driven it is considered a short and displayed in the test results window.

9.2 Drive Test

The drive test checks for a signals ability to be driven low and high. A low to high pulse and high to low pulse is driven onto each bus signal while it is monitored. If the monitored pulse does not match the pulse driving the signal a drive test error is flagged and displayed in the test results window.

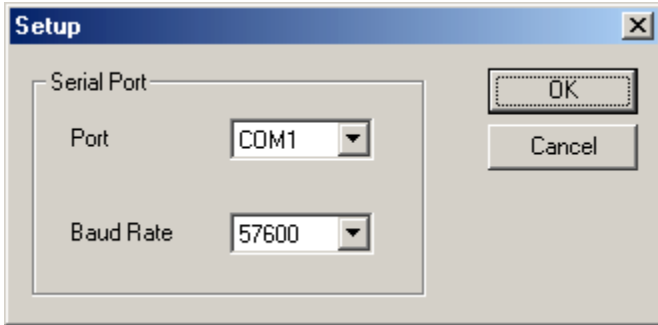
Important: This test requires the system clock. This can be driven from the bus or generated by the analyzer by inserting jumper B5.

**CHAPTER 10
UTILITIES**

The Utilities menu provides for general maintenance, setup and information functions.

10.1 Setup

To configure the RS232 or USB port, click on **Utilities** in the Main Menu and select **Setup**.



For the RS232 interface, enter the COM port number and the baud rate you are using. The board rate of the analyzer will be automatically adjusted to match the software setting. Select USB if using the USB port.

Note: If you are having communication problems using the RS232 interface try lowering the baud rate. The speed of this interface may be influenced by the PC speed and cable type and length.

10.2 Download Firmware

The analyzer firmware resides in a Flash memory that can be reprogrammed. Updates are available on Silicon Control’s web site at www.silicon-control.com. Downloading firmware into the analyzer can take up to 20 minutes.



10.3 Board Information

This function reads basic information from the analyzer. The model number, trace memory size, hardware version and firmware version is displayed.

CHAPTER 11 MISCELLANEOUS

11.1 Online Help

Help on the operation of the analyzer and software is available in the help menu. A contents page presents help in a book form or use index to find information from keywords.

11.2 Expansion Connector

A 200 pin expansion connector provides for add on boards. All bus signals are available on the connector as well as analyzer control and data signals. A 500 Mhz timing module and board test module are currently available.

**APPENDIX A
ANALYZER SPECIFICATIONS**

General Specifications

PCI Compliance: PCI 2.2
 Bus Size: 64 or 32 bit
 Bus Signal Levels: 5V or 3.3V

Trace Specifications

Trace Memory:
 PCI650-1 512KB (32K by 144 bits)
 PCI650-2 1MB (64K by 144 bits)
 PCI650-3 2MB (128K by 144 bits)
 PCI650-4 4MB (256K by 144 bits)

Sampling Rate: 66 Mhz
 High speed module (optional) 500 Mhz

Sampling Modes: System Clock
 System Clock w/ Address/Data
 System Clock w/ Transfers
 On board precision Oscillator
 (7.5ns to 15us)

Sampled Signals: AD[63:0], C/BE[7:0], FRAME, DEVSEL, TRDY, IRDY, PAR, REQ, GNT, RST, LOCK, CLK, INTA, INTB, INTC, INTD, PAR64, PERR, SERR, REQ64, ACK64, TDO, TDI, TCK, TMS, TRST, SDONE, SBO, EXT[7:0]

External Inputs: 8 Front Panel Trace/Trigger

External Outputs: 1 Programmable Trigger Output

Triggers: 8 Trigger Conditions each specifying 100 PCI signals, 8 external triggers and anomaly errors

Trigger Types: Single Condition
 Logical Combination
 16 Level Sequencer

Trigger Positions: 0%, 25%, 50%, 75%, 100%

Occurrence Counters: 16 hardware counters 20 bits

Event Counters: 16 hardware counters 20 bits

Time Tag: 7.5 ns to 60 sec.

Exerciser Specifications

Initiator Bandwidth: 528 MB/s rate

Initiator Bus Width: 64 or 32 bit

Initiator Transfers: Memory, I/O, Configuration

Protocol Violations Checked**Master**

- 1 FRAME deasserted before IRDY asserted
- 2 FRAME asserted without IRDY or TRDY
- 3 FRAME not deasserted within 3 clocks after STOP
- 4 No FRAME within 16 clocks from GNT
- 5 No IRDY within 8 clocks from FRAME
- 6 No TRDY or STOP within 16 clocks from initial data phase
- 7 No TRDY within 8 clocks after initial data phase
- 8 No IRDY after FRAME asserted
- 9 DEVSEL deasserted before transaction complete
- 10 Invalid Command on CB/E[3:0]
- 11 Invalid Command on CB/E[7:4] during DAC cycle
- 12 IRDY asserted during turnaround cycle
- 13 Address PAR error
- 14 Data PAR error
- 15 Address PAR64 error
- 16 Data PAR64 error
- 17 No PERR after parity error
- 18 Improper Master Abort - IRDY deasserted before TRDY/STOP asserted
- 19 FRAME asserted before GNT
- 20 STOP not deasserted after FRAME deasserted
- 21 FRAME and IRDY high when STOP is asserted
- 22 STOP not asserted when FRAME asserted

Target

- 23 No target turnaround
- 24 Lock not released
- 25 STOP deasserted with FRAME deasserted
- 26 PERR asserted during special cycle
- 27 PERR asserted during address cycle
- 28 TRDY asserted during target abort
- 29 LOCK asserted after target abort
- 30 IRDY asserted when FRAME deasserted
- 31 TRDY asserted before DEVSEL
- 32 IRDY asserted before DEVSEL
- 33 STOP asserted before DEVSEL
- 34 LOCK asserted during address phase

General

- 35 Maximum write completion time exceeded
- 36 REQ to GNT time > 15 clocks
- 37 REQ to GNT time > 30 clocks
- 38 REQ to GNT time > 45 clocks
- 39 REQ to GNT time > 60 clocks
- 40 REQ to GNT time > 90 clocks
- 41 Improper fast back-back cycle
- 42 AD[1:0] not 0 during MWI cycle
- 43 REQ64 not asserted with FRAME
- 44 ACK64 not asserted with DEVSEL
- 45 Master Abort during DEVSEL
- 46 FRAME deasserted after DAC cycle
- 47 Target responding to invalid command
- 48 Target response before DEVSEL
- 49 Address does not match Byte Enables

Timing Violations Checked**Unstable Signals**

- 50 AD[7:0] Unstable during address phase
- 51 AD[15:8] Unstable during address phase
- 52 AD[23:16] Unstable during address phase
- 53 AD[31:24] Unstable during address phase
- 54 AD[39:32] Unstable during address phase
- 55 AD[47:40] Unstable during address phase
- 56 AD[55:48] Unstable during address phase
- 57 AD[63:56] Unstable during address phase
- 58 AD[7:0] Unstable during data phase
- 59 AD[15:8] Unstable during data phase
- 60 AD[23:16] Unstable during data phase
- 61 AD[31:24] Unstable during data phase
- 62 AD[39:32] Unstable during data phase
- 63 AD[47:40] Unstable during data phase
- 64 AD[55:48] Unstable during data phase
- 65 AD[63:56] Unstable during data phase
- 66 CBE[3:0] Unstable during address phase
- 67 CBE[7:4] Unstable during address phase
- 68 CBE[3:0] Unstable during data phase
- 69 CBE[7:4] Unstable during data phase

Setup and Hold

- 70 AD[7:0] not valid within 12ns of clock
- 71 AD[15:8] not valid within 12ns of clock
- 72 AD[23:16] not valid within 12ns of clock
- 73 AD[31:24] not valid within 12ns of clock
- 74 AD[39:32] not valid within 12ns of clock
- 75 AD[47:40] not valid within 12ns of clock
- 76 AD[55:48] not valid within 12ns of clock
- 77 AD[63:56] not valid within 12ns of clock
- 78 CBE[3:0] not valid within 12ns of clock
- 79 CBE[7:4] not valid within 12ns of clock
- 80 DEVSEL not valid within 12ns of clock
- 81 FRAME not valid within 12ns of clock
- 82 IRDY not valid within 12ns of clock
- 83 TRDY not valid within 12ns of clock
- 84 LOCK not valid within 12ns of clock
- 85 STOP not valid within 12ns of clock
- 86 REQ64 not valid within 12ns of clock
- 87 ACK64 not valid within 12ns of clock
- 88 PERR not valid within 12ns of clock
- 89 PAR not valid within 12ns of clock
- 90 PAR64 not valid within 12ns of clock

General

- 91 CLK Period < 15ns (66 Mhz system)
- 92 CLK Period < 30ns (33 Mhz system)
- 93 CLK High < 6ns (66 Mhz system)
- 94 CLK Low < 6ns (66 Mhz system)
- 95 CLK High < 11ns (33 Mhz system)
- 96 CLK Low < 11ns (33 Mhz system)
- 97 Glitches on CLK
- 98 RESET high to first Configuration access invalid
- 99 RESET high to first FRAME active invalid
- 100 RESET to REQ64 time invalid
- 101 Glitches on RESET

Performance Analysis

- Functions:
- Bus Utilization
 - Transfer Rate
 - Latency
 - Burst Distribution
 - Statistics

- Measurement Displays:
- Moving Time Graph
 - Average Bar Chart
 - Minimum Bar Chart
 - Maximum Bar Chart

- Number of Measurements:
- 8 Simultaneous Conditions

- Measurement Conditions:
- All Bus Signals
 - Commands
 - Address Ranges
 - Initiator Specific
 - Target Specific
 - Arbitration Specific

Front Panel Interfaces

- RS232 Port: DB9 connector, 110 to 115K Baud (cable included)
- USB Port: Series B connector, 12 MB/s (cable included)
- Indicators: GO LED, User LED
- Pushbutton: Reset Analyzer or System
- External Power: 2 Conductor front panel (cable included)
- Trigger: 10 pin socket (8 in, 1 out, 1 ground) (cable included)
- Fuses: Main power and External power

Power Requirements

- Operating—5V at 3 amps max
- Standby—5V at 1 Amp max
- Generated on-board— 3.3V and 2.5V

Dimensions:

- PCI650—PCI Short Card
- PMC650— Single slot PMC card
- CPCI650— 3U Compact PCI card

Ordering Information:

PCI Analyzers

- PCI650-1 PCI Bus Analyzer/Exerciser
32K Trace Buffer
- PCI650-2 PCI Bus Analyzer/Exerciser
64K Trace Buffer
- PCI650-3 PCI Bus Analyzer/Exerciser
128K Trace Buffer
- PCI650-4 PCI Bus Analyzer/Exerciser
256K Trace Buffer

Compact PCI Analyzers

- CPCI650-1 CPCI Bus Analyzer/Exerciser
32K Trace Buffer
- CPCI650-2 CPCI Bus Analyzer/Exerciser
64K Trace Buffer
- CPCI650-3 CPCI Bus Analyzer/Exerciser
128K Trace Buffer
- CPCI650-4 CPCI Bus Analyzer/Exerciser
256K Trace Buffer

PMC Analyzers

- PMC650-1 PMC Bus Analyzer/Exerciser
32K Trace Buffer
- PMC650-2 PMC Bus Analyzer/Exerciser
64K Trace Buffer
- PMC650-3 PMC Bus Analyzer/Exerciser
128K Trace Buffer
- PMC650-4 PMC Bus Analyzer/Exerciser
256K Trace Buffer

All analyzers include Windows software, cables and manuals.

PCI Bus Analyzer

Application Programming Interface

Version 1.1

Scope

This document describes the communications interface to the Silicon Controls PCI8XX and PCI6XX bus analyzers. The interface is intended for use by custom applications that wish to programatically control the operation of the analyzer. Communications functions are provided to configure the communications port, send commands and return responses. It is the responsibility of the custom application to format commands and parse data that is returned.

Environment

The API is provided as a library to be linked with a custom C or C++ application. The library has been compiled with the Microsoft Developer Studio compiler, version 6. The library may be used on Windows 9X, NT and 2000 systems. Note that not all Windows operating systems support the USB analyzer interface.

The API uses the Microsoft multi-threaded run-time library. Applications that use the API should specify code generation with the multi-threaded or multi-threaded DLL library.

Directories

The API is delivered with the following directory structure:

Docs	The API interface document.
PciApi	The API library code, with debug and release builds.
Sample	Sample console application which reads setup memory.
Sample2	Sample console application which starts and stops a trace, and reads trace samples.
TestApp	Test application which provides a GUI interface to exercise the API.

References

“PCI850 Interface Document”

Overview

The API starts with a set of functions to manage the communications port. *PciInitPort* is used to specify the port to use, which may be a serial port or USB interface. If a serial port is requested, the baud rate is also specified. The *PciClosePort* function is called when the communications port is no longer needed.

The custom application is the master of the communications channel. All commands are initiated by the application, and any data from the analyzer is sent as a response to a command.

Two levels of communications are provided by the API. A low-level interface is available through the *PciWriteRawData* and *ReadRawData* interfaces. Data is sent without modification by the *PciWriteRawData* function. Data from the analyzer is then read by calling *PciReadRawData* repeatedly until all expected data has been received.

A higher-level interface is also available which implements a series of commands, and interprets the responses. An example of such a command would be reading a portion of the analyzer Setup memory and placing the data in a user supplied buffer. This operation would require a series of commands to set the setup memory address, length, and initiate the read operation. The returned responses would also need to be parsed to test for valid acknowledgements and to extract the returned data. This sequence of commands and responses is handled automatically by this higher-level interface.

Functions in the higher-level interface start with *PciAllocCommand*, which is used to create a command request. The application receives a pointer to the request, and is required to fill in certain parameters in the request. *PciSendCommand* is then called to initiate the request. The application can then call *PciWaitForCompletion* if it wishes to wait for the command to be completed. Or a callback function may be specified when the request is allocated, which will be called when the command is complete, or if an error occurs.

Function Reference

PciInitPort

Prototype:

```
t_pciComError PciInitPort(char *portName, int baud);
```

Parameters:

portName - A string that specifies the communications port to use to communicate with the analyzer. Valid strings are of the form "COM#" (# = 1, 2, 3, ...) or "USB".

baud - If the portName is "COM#", this parameter specifies the baud rate. Valid values are 9600, 19200, 28800, 38400, and 57600.

Returns:

PCICOM_SUCCESS, or error code as specified in pciComError.h.

Description:

This function initializes a communications channel to the analyzer. A serial port (specified by "COM#"), or a USB interface may be specified. For a serial port, the specified baud rate is used to configure the port. This baud rate is also used to configure the analyzer serial port.

This function also starts a thread to receive data from the specified port.

PciClosePort

Prototype:

```
void PciClosePort();
```

Parameters:

None

Returns:

None

Description:

This function terminates the communications channel to the analyzer.

PciWriteRawData

Prototype:

```
t_pciComError PciWriteRawData(unsigned char *data, int len);
```

Parameters:

data - One or more bytes of data to send to the analyzer.

len - Number of bytes to be sent.

Returns:

PCICOM_SUCCESS, or error code as specified in pciComError.h.

Description:

This function writes the specified data bytes to the analyzer. The function does not return until all bytes have been sent.

PciReadRawData

Prototype:

```
t_pciComError PciReadRawData(unsigned char *data, int bufferSize,  
int *retLen);
```

Parameters:

data - Pointer to a buffer to receive the data from the analyzer.

bufferSize - Size of data buffer.

Returns:

retLen - Number of bytes copied to the data buffer.

PCICOM_SUCCESS, or error code as specified in pciComError.h.

Description:

This function checks if any data has been received from the analyzer. The data is copied, up to a maximum of bufferSize bytes, to the buffer specified by the data parameter. The retLen parameter is set to the number of bytes copied by this function.

This function will not wait for data to be received. It will return immediately, and retLen will be set to zero if no received data is available.

PciAllocCommand

Prototype:

```
t_pciCommand *ciAllocCommand(t_pciCommandType commandType,
                             void (*callback)(t_pciCommand *cmd), long gpValue);
```

Parameters:

commandType - Indicates the type of command to perform with the analyzer.
callback - Specifies a function to be called when the command completes, or when an error occurs. This parameter may be NULL if no callback is required.
gpValue - General purpose 32-bit value which will be passed to callback function. The application may use the value for any purpose. Note that the callback function is also passed a pointer to the command structure.

Returns:

t_pciCommand - A pointer to the command structure is returned if the function is successful. Otherwise, NULL is returned.

Description:

This function allocates a command structure to perform an analyzer operation. The type of command is specified by the **commandType** parameter, according to the following table. The caller is responsible for setting the other parameters in the command structure according to the table.

The **t_pciCommand** structure is defined as follows. Normally, the caller only fills in the **m_param** values, as required for the specified command.

```
typedef struct
{
    t_PciCommandType  m_command;
    t_CommandError    m_error;        // Indicates result of operation
    long              m_gpVal;
    void              (*m_callback)(t_pciCommand *cmd);
    long              m_param[MAX_PCI_PARAMS];
} t_pciCommand;
```

This function does not initiate the command. The **SendCommand** function must be called to start sending the command.

t_pciCommandType	m_param[0]	m_param[1]	m_param[2]	Description
c_readSetup	address	length	pointer to buffer	Reads length bytes from setup memory into buffer.
c_writeSetup	address	length	Pointer to buffer	Writes length bytes from buffer to setup memory.
c_readTraceBeforeTrigger	Sample index	number of samples to read	pointer to buffer	Reads trace data samples before the trigger and places them in the buffer.
c_readTraceAfterTrigger	Sample index	number of samples to read	Pointer to buffer	Reads trace data samples after the trigger and places them in the buffer.
c_readHSTraceBeforeTrigger	sample index	number of samples to read	pointer to buffer	Reads high speed trace data samples before the trigger and places them in the buffer.
c_readHSTraceAfterTrigger	sample index	number of samples to read	pointer to buffer	Reads high speed trace data samples after the trigger and places them in the buffer.
c_traceGo	n/a	n/a	n/a	Starts trace capture.
c_traceStop	n/a	n/a	n/a	Stops trace capture.
c_startSearch	n/a	n/a	n/a	Starts trace search.
c_stopSearch	n/a	n/a	n/a	Stops trace search.
c_perfGo	n/a	n/a	n/a	Starts performance measurements.
c_perfStop	n/a	n/a	n/a	Stops performance measurements.
c_anomalyGo	n/a	n/a	n/a	Starts anomaly detection.
c_anomalyStop	n/a	n/a	n/a	Stops anomaly detection.
c_complianceTestGo	n/a	n/a	n/a	Starts compliance test.

c_complianceTestStop	n/a	n/a	n/a	Stops compliance test.
c_startConfigScan	n/a	n/a	n/a	Starts configuration scan.
c_stopConfigScan	n/a	n/a	n/a	Stops configuration scan.
c_exerciserGo	n/a	n/a	n/a	Starts exerciser.
c_exerciserStop	n/a	n/a	n/a	Stops exerciser.
c_stimulusGo	n/a	n/a	n/a	Starts stimulus.
c_stimulusStop	n/a	n/a	n/a	Stops stimulus.
c_writeFlash	address	length	Pointer to buffer	Writes length bytes from buffer to flash memory at the specified address.

PciSendCommand

Prototype:

```
t_pciComError PciSendCommand(t_pciCommand *cmd);
```

Parameters:

cmd - Command to perform. This is a pointer returned by the AllocCommand function.

Returns:

PCICOM_SUCCESS, or error code as specified in pciComError.h.

Description:

This function initiates the command specified by the cmd structure. If another command is already in progress, this command is added to a queue. This function returns without waiting for the comand to complete.

If specified when PciAllocCommand was called, the callback function will be called when the command completes, or when an error is encountered processing the command. The PciWaitForCompletion function may also be called to wait for any pending commands to be completed.

PciWaitForCompletion

Prototype:

```
t_pciComError PciWaitForCompletion();
```

Parameters:

None

Returns:

PCICOM_SUCCESS, or error code as specified in pciComError.h.

Description:

This function waits for any current or pending commands to complete before returning. A timeout error may be returned by this function if command processing does not complete within a maximum time period.

PciGetErrorString**Prototype:**

```
char *PciGetErrorString(t_pciComError errCode);
```

Parameters:

None

Returns:

Pointer to string which describes the error code.

Description:

This function translates an error code to a descriptive string. A pointer to the string is returned.

This page intentionally left blank