Silicon Control Inc.
PCI Bus Analyzer
Application Programming Interface

Version 1.3

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 programmatically 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.
Sample3	Sample console application which writes and reads setup memory.
Sample4	Simple application which writes a single byte to setup memory.
TestApp	Test application which provides a GUI interface to exercise the API.

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.
c_readTarget	address	length	pointer to buffer	Reads length bytes from target memory into buffer.
c_writeTarget	address	length	pointer to buffer	Writes length bytes from buffer to target

				memory.
c_selfTestGo	n/a	n/a	n/a	Starts self test.
c_selfTestStop	n/a	n/a	n/a	Stops self test.
c_backplaneTestGo	n/a	n/a	n/a	Starts backplane test.
c_backplaneTestStop	n/a	n/a	n/a	Stops backplane test.

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.

Setup Memory Space Definitions

The setup memory contains all the parameters to implement a specific analyzer command and command status. This section starts with a summary of all the setup memory locations and is followed by details of each parameter. The text in RED describes how the Analyze PC software implements the analyzer functions and can be used for reference purposes.

Setup Memory Address Summary

Address (decimal)	Fields
0 - 7	Board Name
8	Model Number
9 -10	Trace Buffer Size
11	Firmware Version Code
12	Hardware Version Code
13	Reserved
14	USB Control
15	RS232 Control
16	Serial Number Low
17	Serial Number High
18	Bus Info
19	Bus Speed
20 - 49	Trace Condition E1
50 - 79	Trace Condition E2
80 - 109	Trace Condition E3
110 - 139	Trace Condition E4
140 - 169	Trace Condition E5
170 - 199	Trace Condition E6
200 - 229	Trace Condition E7
230 - 259	Trace Condition E8
260 - 279	Trace Control L1
280 - 299	Trace Control L2
300 - 319	Trace Control L3
320 - 339	Trace Control L4
340 - 359	Trace Control L5
360 - 379	Trace Control L6
380 - 399	Trace Control L7
400 - 419	Trace Control L8
420	Sample Clock
421	Trigger Position
422	Power Zoom Sample Clock
423	Reserved
424	GO Control
425	GO Status
426 - 428	# Samples before trigger
429 - 431	# Samples after trigger
432 - 439	Reserved
440	Search Control

441 - 444	Search Status
445 - 474	Search Condition
475 - 479	Reserved
480	Self Test Control
481	Self Test Status
482	Self Test Result
483-496	Reserved
497	Performance Analysis Control
498	Performance Analysis Status
499	Performance Analysis Type
500 - 529	Performance Item 1 Setup
530 - 559	Performance Item 2 Setup
560 - 589	Performance Item 3 Setup
590 - 619	Performance Item 4 Setup
620 - 649	Performance Item 5 Setup
650 - 679	Performance Item 6 Setup
680 - 709	Performance Item 7 Setup
710 - 739	Performance Item 8 Setup
740 - 747	Performance Total Sample #
748 - 751	Performance Item 1 Results
752 - 755	Performance Item 2 Results
756 - 759	Performance Item 3 Results
760 - 763	Performance Item 4 Results
764 - 767	Performance Item 5 Results
768 - 771	Performance Item 6 Results
772 - 775	Performance Item 7 Results
776 - 799	Performance Item 8 Results
800	Master Control
801	Master Status
802	Master Address Space
803	Master Address Size
804 - 811	Master Start Address
812	Master Start Address Length
813	Master Data Size
814	Master Data Cycle
815	Master Data Type
816 - 847	Master Data
848	AM code (VME only)
849	Bus Request Level (VME only)
850	Configuration Scan Control
851	Configuration Scan Status
852 - 915	Configuration Data
916 - 923	Configuration Address
924 – 927	Reserved
928	Stimulus Control
929	Stimulus Status
930 - 959	Stimulus Condition S1
960 -989	Stimulus Condition S2
990 - 1019	Stimulus Condition S3

1020 - 1049	Stimulus Condition S4
1050 - 1079	Stimulus Condition S5
1080 - 1109	Stimulus Condition S6
1110 - 1139	Stimulus Condition S7
1140 - 1169	Stimulus Condition S8
1170 - 1199	Stimulus Condition S9
1200 - 1229	Stimulus Condition S10
1230 - 1259	Stimulus Condition S11
1260 - 1289	Stimulus Condition S12
1290 - 1319	Stimulus Condition S13
1320 - 1349	Stimulus Condition S14
1350 - 1379	Stimulus Condition S15
1380 - 1409	Stimulus Condition S16
1410 - 1414	Stimulus Control SL1
1415 - 1419	Stimulus Control SL2
1420 - 1424	Stimulus Control SL3
1425 - 1429	Stimulus Control SL4
1430 - 1434	Stimulus Control SL5
1435 - 1439	Stimulus Control SL6
1440 - 1444	Stimulus Control SL7
1445 - 1449	Stimulus Control SL8
1450 - 1454	Stimulus Control SL9
1455 - 1459	Stimulus Control SL10
1460 - 1464	Stimulus Control SL11
1465 - 1469	Stimulus Control SL12
1470 - 1474	Stimulus Control SL13
1475 - 1479	Stimulus Control SL14
1480 - 1484	Stimulus Control SL15
1485 - 1489	Stimulus Control SL16
1490 - 1497	Target Bus Address
1498	Target Bus Address Length
1499	Target Bus Address Parameters
1500	Target Bus Data Parameters
1501	Target Control
1502	Target Status
1503-1510	Target Address
1511	Target Address Length
1512	Target Data Value
1513	Target Data Parameters
1514 - 1515	Reserved
1516	Compliance Control
1517	Compliance Status
1518 -1521	Compliance Test
1522 - 1529	Reserved
1530	Anomaly Control
1531	Anomaly Status
1532 - 1547	Anomaly Checklist
1548	Anomaly Results
1549	Reserved

1550	Backplane Test Control
1551	Backplane Test Status
1552 - 1581	Backplane Test Signals
1582 - 1599	Reserved
1600 - 1629	Stimulus Event 1
1630 - 1659	Stimulus Event 2
1660 - 1689	Stimulus Event 3
1690 - 1719	Stimulus Event 4
1720 - 1749	Stimulus Event 5
1750 - 1779	Stimulus Event 6
1780 - 1809	Stimulus Event 7
1810 - 1839	Stimulus Event 8
1840 - 2047	Reserved

Setup Memory Detailed Bit Definitions

Board Information

Board Information identifies the name and model of the analyzer, trace buffer size and Firmware and Hardware versions.

Address

0 - 7	Board Name
8	Model Number
9 -10	Trace Buffer Size
11	Firmware Version Code
12	Hardware Version Code
13	Reserved
14	USB Control
15	RS232 Control
16	Serial Number Low
17	Serial Number High
18	Bus Info
19	Bus Speed

Board Name – Name of product in ASCII. Byte 0 is first letter. (CPCI850)

byte	D7	D6	D5	D4	D3	D2	D1	D0
0								
1								
2								
3								
4								
5								
6								
7								

Model number – A number from 1 to 6 is displayed with a dash after the Board name (2 is a CPCI850-2)

byte	D7	D6	D5	D4	D3	D2	D1	D0
8								

Trace Buffer Size – The trace buffer size in Kbytes. (512 is 512K bytes)

byte	D7	D6	D5	D4	D3	D2	D1	D0
9								LSB
10								

Firmware Version – Use 1 decimal point. (11 is 1.1)

byte	D7	D6	D5	D4	D3	D2	D1	D0
11								

Hardware Version – Use 1 decimal point (10 is 1.0)

byte	D7	D6	D5	D4	D3	D2	D1	D0
12								

RS232 Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
15						BAUD2	BAUD1	BAUD0

BAUD2-0 0=9600, 1=19200, 2=28800, 3=38400, 4=57600, 5=115.2K*, 6=750K*, 8=4800

* error is greater then 2% which may not be reliable.

To take effect you must issue a Change RS232 register command (command #29).

Serial Number

byte	D7	D6	D5	D4	D3	D2	D1	D0
16								LOW
17								HIGH

Serial Number in HEX. For example SN# 2164, byte 16 = 64 hex and byte 17 = 21 hex.

Bus Info

byte	D7	D6	D5	D4	D3	D2	D1	D0
18				VOLTAGE		WIDTH		MODE

VOLTAGE 0 = 5V, 1=3.3V

WIDTH 0=32 bits, 1=64 bits

MODE 0=PCI, 1=PCI-X

Bus Speed

byte	D7	D6	D5	D4	D3	D2	D1	D0
19								

Bus speed in Mhz (decimal).

Capture - Trace Conditions

Trace Conditions define what bus events to look for. They are used in trigger and store specifications in the Trace Control definitions. There are 8 conditions and they can be used individually or logically combined. For example E1.(E2+E3) means E2 OR E3 ANDED with E1.

Address

20 - 49	Trace Condition E1
50 - 79	Trace Condition E2
80 - 109	Trace Condition E3
110 - 139	Trace Condition E4
140 - 169	Trace Condition E5
170 - 199	Trace Condition E6
200 - 229	Trace Condition E7
230 - 259	Trace Condition E8

Trace Condition Bit Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1								
2	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
3								
4	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
5								
6	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
7								
8	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
9								
10	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
11								
12	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
13								
14	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
15								
16	CBE7	CBE6	CBE5	CBE4	CBE3	CBE2	CBE1	CBE0
17								
18	RST	LOCK	IDSEL	DEVSEL	STOP	IRDY	TRDY	FRAME
19								
20	PAR64	PAR	ACK64	REQ64	GNT	REQ	SERR	PERR
21								
22	ADR/DATA	ANOMALY	SBO	SDONE	INTD	INTC	INTB	INTA
23								
24	EXT7	EXT6	EXT5	EXT4	EXT3	EXT2	EXT1	EXT0
25								
26	Range 1	Range 0	CLK	TRST	TMS	TCK	TDO	TDI
27								

All signals (even bytes) can be set to a 0 or 1 and are followed by a mask (odd bytes). If a mask bit is set to 1 the corresponding signal is used otherwise it is a don't care.

The Range bits specify an additional condition for the AD signals. They define if a match should be <,>, or = .

Range 1	Range 0	Condition	ADR/DATA	ANOMALY
0	0	=	1 = Address	0 = None
0	1	>	0 = Data	1 = Use Checklist
1	0	<		

Capture - Trace Controls

Trace Controls define what the analyzer does based on trace conditions. A single trace control can be used or multiple trace control levels can be used. Each control specifies a trace condition (in the IF field), trace condition count (in the OCCURS field), store condition (in the STORE field) and jumps (in the THEN and ELSE fields) to other Trace Controls based on the trace condition. Since logical combinations of trace conditions can be used for both the trace and store fields a compact decoding using nibbles for each element in the expression is used to save space.

Address

260 - 279	Trace Control L1
280 - 299	Trace Control L2
300 - 319	Trace Control L3
320 - 339	Trace Control L4
340 - 359	Trace Control L5
360 - 379	Trace Control L6
380 - 399	Trace Control L7
400 - 419	Trace Control L8

Trace Control Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	TRACE7	TRACE6	TRACE5	TRACE4	TRACE3	TRACE2	TRACE1	TRACE0
1	TRACE15	TRACE14	TRACE13	TRACE12	TRACE11	TRACE10	TRACE9	TRACE8
2	TRACE23	TRACE22	TRACE21	TRACE20	TRACE19	TRACE18	TRACE17	TRACE16
3	TRACE31	TRACE30	TRACE29	TRACE28	TRACE27	TRACE26	TRACE25	TRACE24
4	TRACE39	TRACE38	TRACE37	TRACE36	TRACE35	TRACE34	TRACE33	TRACE32
5	TRACE47	TRACE46	TRACE45	TRACE44	TRACE43	TRACE42	TRACE41	TRACE40
6	TRACE55	TRACE54	TRACE53	TRACE52	TRACE51	TRACE50	TRACE49	TRACE48
7	TRACE63	TRACE62	TRACE61	TRACE60	TRACE59	TRACE58	TRACE57	TRACE56
8	STORE7	STORE6	STORE5	STORE4	STORE3	STORE2	STORE1	STORE0
9	STORE15	STORE14	STORE13	STORE12	STORE11	STORE10	STORE9	STORE8
10	STORE23	STORE22	STORE21	STORE20	STORE19	STORE18	STORE17	STORE16
11	STORE31	STORE30	STORE29	STORE28	STORE27	STORE26	STORE25	STORE24
12	STORE39	STORE38	STORE37	STORE36	STORE35	STORE34	STORE33	STORE32
13	STORE47	STORE46	STORE45	STORE44	STORE43	STORE42	STORE41	STORE40
14	STORE55	STORE54	STORE53	STORE52	STORE51	STORE50	STORE49	STORE48
15	STORE63	STORE62	STORE61	STORE60	STORE59	STORE58	STORE57	STORE56
16	OCCUR7	OCCUR6	OCCUR5	OCCUR4	OCCUR3	OCCUR2	OCCUR1	OCCUR0
17	OCCUR15	OCCUR14	OCCUR13	OCCUR12	OCCUR11	OCCUR10	OCCUR9	OCCUR8
18	F_THEN3	F_THEN2	F_THEN1	F_THEN0	TRIG_THEN	JMP_THEN2	JMP_THEN1	JMP_THEN0
19	F_ELSE3	F_ELSE2	F_ELSE1	F_ELSE0	TRIG_ELSE	JMP_ELSE2	JMP_ELSE1	JMP_ELSE0

TRACE (xx) Trace Expression – A decoded representation of a trace expression in the IF field of the trace control using nibbles for each element in the expression. The elements are defined as:

TRACE3	TRACE2	TRACE1	TRACE0	
0	0	0	0	None*
0	0	0	1	E1
0	0	1	0	E2
0	0	1	1	E3
0	1	0	0	E4
0	1	0	1	E5
0	1	1	0	E6
0	1	1	1	E7
1	0	0	0	E8
1	0	0	1	**.** (AND)
1	0	1	0	+ (OR)
1	0	1	1	~ (NOT)
1	1	0	0	X (XOR)
1	1	0	1	(
1	1	1	0	)
1	1	1	1	All*

* The entire expression is set to zeros if the expression is None
* The entire expression is set to F's if the expression is All

As an example the expression E1+E2**.**E5 would be decoded as:

Byte 7 00011010
(E1) (+)

Byte 6 00101001
(E2) (**.**)

Byte 5 01010000
(E5)

STORE (xx) Store Expression – A decoded representation of a store expression in the STORE field of the trace control using nibbles for each element in the expression. See TRACExx above for decoding.

OCCUR (15-0) Occurrence Count – The number of occurrences of the trace expression before executing the THEN field. The default value is 1 and the maximum value is 64K.
Special case: A value of zero is valid and indicates no trigger should be used. In this case the board never triggers, Buffer Full never occurs and the only way to stop tracing is by clicking the STOP button. This is a useful feature called GO FOREVER.

JMP_THEN (2-0) Jump to another trace control level if the trace expression is true. Values 0 to 7.

JMP_ELSE (2-0) Jump to another trace control level if the trace expression is false. Values 0 to 7.

TRIG_THEN Trigger if trace expression is true. Value is 1 if you want to trigger.

TRIG_ELSE Trigger if trace expression is false. Value is 1 if you want to trigger.

F_THEN (3-0) Function if true. Perform additional functions if trace expression is true.

F_ELSE (3-0) Function if true. Perform additional functions if trace expression is false

.

Capture - Miscellaneous Trace Controls

Address

420	Sample Clock
421	Trigger Position
422	Power Zoom Sample Clock
423	Reserved
424	GO Control

Sample Clock – Sample clock source

byte	D7	D6	D5	D4	D3	D2	D1	D0
420					CLK3	CLK2	CLK1	CLK0

CLK3-0

0	Synchronous	8	240 ns (4 Mhz)
1	Transfer	9	480 ns (2 Mhz)
2	System Clock	10	960 ns (1 Mhz)
3	7.5 ns (133 Mhz)	11	1.92 us (520 Khz)
4	15 ns (66 Mhz)	12	3.84 us (260 Khz)
5	30 ns (33 Mhz)	13	7.68 us (130 Khz)
6	60 ns (16 Mhz)	14	15.36 us (65 Khz)
7	120 ns (8 Mhz)	15	30.72 us (32.5 Khz)

Trigger Position

byte	D7	D6	D5	D4	D3	D2	D1	D0
421						POS2	POS1	POS0

POS2-0

0	½
1	¾
2	End
3	Start
4	¼

Power Zoom Sample Clock

byte	D7	D6	D5	D4	D3	D2	D1	D0
422						CLK2	CLK1	CLK0

CLK2-0

0	1.875 ns
1	3.75 ns
2	7.5 ns
3	15 ns
4	30 ns
5	60 ns
6	120 ns
7	240 ns

GO Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
424						LEV2	LEV1	LEV0

LEV2-0 Start GO at Trace Control Level L(# + 1). For example if LEV2-0 is 0 then start capturing trace data using Trace Control 1.

Capture - GO Status

Address

425	GO Status
426 - 428	# Samples before trigger
429 - 431	# Samples after trigger

Go Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
425	ACTIVE		TRIGGERED?	FULL?		LEV2	LEV1	LEV0

LEV2-0 Current Active Trace Control Level L(# + 1)
FULL Buffer Full = 1. When Buffer is full tracing stops automatically. Display trace data around trigger.
TRIGGERED Trigger Encountered = 1
ACTIVE Tracing in Progress = 1, Trace Stopped = 0

Samples Before Trigger

byte	D7	D6	D5	D4	D3	D2	D1	D0
426								LSB
427								
428								MSB

Number of samples captured before trigger.

Samples After Trigger

byte	D7	D6	D5	D4	D3	D2	D1	D0
429								LSB
430								
431								MSB

Number of samples captured after trigger.

Capture - Search Trace

Address

440	Search Control
441 - 444	Search Status
445 - 474	Search Condition

Search Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
440							DIR1	DIR0

DIR1	DIR0	Search Direction
0	0	Forward
0	1	Backward
1	0	From Beginning

Search Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
441	ACTIVE						B/A	MATCH
442	S7	S6	S5	S4	S3	S2	S1	S0
443	S15	S14	S13	S12	S11	S10	S9	S8
444	S23	S22	S21	S20	S19	S18	S17	S16

ACTIVE	Search in progress = 1
B/A	Before (0) / After (1) Trigger
MATCH	Search Match Found = 1
S23-S0	Search Sample Number Found (read)
	Start Search at Sample Number (write)

Search Condition

byte	D7	D6	D5	D4	D3	D2	D1	D0
445-474	same	as	trace	condition				

The search condition has the same definition as the trace condition. Signals are either 0, 1 or Don’t Care using the mask bit.

Windows Operation for Capture

Tracing and Displaying Bus Data

The basic steps to capture and display bus activity are as follows:

1. Load Setup Memory
2. Send Go Command
3. Monitor Status
4. Send STOP Command
5. Read and Display Trace Data

During each step the Status Bar in the Trace Window reflects the state of trace activity. A text message is displayed on the far right side such as IDLE, LOADING, ACTIVE, SAVED, CAPTURED, READING.

1. Load Setup Memory

Setup memory may be loaded any time prior to issuing the GO register command. Use the WRITE SETUP MEMORY register command to load setup information into the analyzer. Writing multiple bytes of setup information by increasing the length parameter will save load time. Display LOADING in the Status Bar.

The setup locations involved in tracing are the:

Trace Conditions
Trace Controls
Sample Clock
Trigger Position
GO Control

2. Send GO Command

To start tracing send the GO register command. Display ACTIVE in the Status Bar.

3. Monitor Status

Read the GO status, Samples Before and Samples After the Trigger located in the setup memory to monitor trace progress. The sample numbers and a bar graph showing the amount of samples in the trace buffer are displayed in the Status Bar in the Trace Window.

4. Send STOP Command

The STOP command should be sent if the BUFFER FULL status is 1 or the user clicks the STOP button.

5. Read and Display Trace Data

Issue Read Trace Memory register commands to read trace data. Use the Read Trace Memory Before Trigger register command to read trace data before the trigger and the Read Trace Memory After Trigger register command to read trace data after the trigger. Set the starting address to the sample number. Set the Length register for the amount of bytes to transfer (each sample is 16 bytes).

The trace window should display the trigger as sample number 0, samples before the trigger are negative and samples after the trigger positive.

Examples:

Read 64 trace samples after the trigger.
Write 0 to start address register
Write 1024 to length register (64 x 16 bytes/sample)
Send Read Trace After Trigger Command

Read Sample –10 to +10
Write 10 to start address register
Write 320 to length register (20 x 16 bytes/sample)
Send Read Trace Before Trigger Command

Performance Analysis

Address

497	Performance Analysis Control
498	Performance Analysis Status
499	Performance Analysis Type
500 - 529	Performance Item 1 Setup
530 - 559	Performance Item 2 Setup
560 - 589	Performance Item 3 Setup
590 - 619	Performance Item 4 Setup
620 - 649	Performance Item 5 Setup
650 - 679	Performance Item 6 Setup
680 - 709	Performance Item 7 Setup
710 - 739	Performance Item 8 Setup
740 - 747	Performance Total Sample #
748 - 751	Performance Item 1 Results
752 - 755	Performance Item 2 Results
756 - 759	Performance Item 3 Results
760 - 763	Performance Item 4 Results
764 - 767	Performance Item 5 Results
768 - 771	Performance Item 6 Results
772 - 775	Performance Item 7 Results
776 - 799	Performance Item 8 Results

Performance Analysis Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
497								

Performance Analysis Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
498	ACTIVE							

ACTIVE Performance Analysis in progress = 1

Performance Analysis Type

byte	D7	D6	D5	D4	D3	D2	D1	D0
499						TYPE2	TYPE1	TYPE0

TYPE (2-0) 0=Utilization, 1=Transfer Rate, 2=Latency, 3=Burst Distribution, 4=Statistics

Performance Item

byte	-	D7	D6	D5	D4	D3	D2	D1	D0
0	**L**	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1	**O**	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
2	**W**	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
3		AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
4	**A**	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
5	**D**	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
6	**R**	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
7	**-**	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
8	**H**	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
9	**I**	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
10	**G**	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
11	**H**	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
12		AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32

13	**A**	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
14	**D**	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
15	**R**	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
16					CYCLE4	CYCLE3	CYCLE2	CYCLE1	CYCLE0
17					ITEM4	ITEM3	ITEM2	ITEM1	ITEM0

AD (63-0) Address with mask bits

CYCLE (4-0) Cycle Type

0	All Cycles
1	Memory
2	Memory Read
3	Memory Write
4	I/O
5	I/O Read
6	I/O Write
7	Configuration
8	Configuration Read
9	Configuration Write
10	Memory Read Multiple
11	Dual Address Cycle
12	Memory Read Line
13	Memory Write and Invalidate
14	Interrupt Acknowledge
15	Special Cycle

16	Event1 – E1
17	Event2 – E2
18	Event3 – E3
19	Event4 – E4
20	Event5 – E5
21	Event6 – E6
22	Event7 – E7
23	Event8 – E8

Note: Events are the 8 trace conditions.

ITEM (4-0) Item selection is different for each performance analysis type:

ITEM (4-0)	Utilization	Transfer Rate	Latency	Burst Distribution	Statistics
0	None	None	None	None	None
1	Bus Busy *	Cycle *	Master *	Cycle *	Cycle *
2	Bus Idle *	GNT0 *	Target (Init) *		Master Abort
3	Transfers *	GNT1 *	Target *		Target Abort
4	Address Phase *	GNT2 *	Arbitration *		Target Retry
5	Data Phase *	GNT3 *			Disconnect w/data
6	Wait States *				Disconnect wo/data
7	Master Efficiency *				Split Response
8	Target Efficiency *				
9					
10					
11					
12					
13					
14					

* Load these items as the default

For burst distribution defaults are Cycle: All Cycle: Mem Cycle: I/O Cycle : Cfg

For Statistics defaults are Cycle: Mem Rd Cycle: Mem Wr Cycle: I/O Rd Cycle: I/O Wr Cycle: Cfg Rd Cycle: Cfg Cfg Wr

Anomaly

Address

1530	Anomaly Control
1531	Anomaly Status
1532 - 1547	Anomaly Checklist
1548	Anomaly Results

Anomaly Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
1530								

Not Used

Anomaly Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
1531	ACTIVE							

ACTIVE Anomaly detection in progress = 1

Anomaly Checklist

byte	D7	D6	D5	D4	D3	D2	D1	D0
1532	Anomaly 8	Anomaly 7	Anomaly 6	Anomaly 5	Anomaly 4	Anomaly 3	Anomaly 2	Anomaly 1
1533	Anomaly 16	Anomaly 15	Anomaly 14	Anomaly 13	Anomaly 12	Anomaly 11	Anomaly 10	Anomaly 9
1534	Anomaly 24	Anomaly 23	Anomaly 22	Anomaly 21	Anomaly 20	Anomaly 19	Anomaly18	Anomaly 17
1535	Anomaly 32	Anomaly 31	Anomaly 30	Anomaly 29	Anomaly 28	Anomaly 27	Anomaly 26	Anomaly 25
1536	Anomaly 40	Anomaly 39	Anomaly 37	Anomaly 37	Anomaly 36	Anomaly 35	Anomaly 34	Anomaly 33
1537	Anomaly 48	Anomaly 47	Anomaly 46	Anomaly 45	Anomaly 44	Anomaly 43	Anomaly 42	Anomaly 41
1538	Anomaly 56	Anomaly 55	Anomaly 54	Anomaly 53	Anomaly 52	Anomaly 51	Anomaly 50	Anomaly 49
1539	Anomaly 64	Anomaly 63	Anomaly 62	Anomaly 61	Anomaly60	Anomaly59	Anomaly 58	Anomaly 57
1540	Anomaly 72	Anomaly 71	Anomaly 70	Anomaly 69	Anomaly 68	Anomaly 67	Anomaly 66	Anomaly 65
1541	Anomaly 80	Anomaly 79	Anomaly 78	Anomaly 77	Anomaly 76	Anomaly 75	Anomaly 74	Anomaly 73
1542	Anomaly 88	Anomaly 87	Anomaly 86	Anomaly 85	Anomaly 84	Anomaly 83	Anomaly 82	Anomaly 81
1543	Anomaly 96	Anomaly 95	Anomaly 94	Anomaly 93	Anomaly 92	Anomaly 91	Anomaly 90	Anomaly 89
1544	Anomaly 104	Anomaly 103	Anomaly 102	Anomaly 101	Anomaly 100	Anomaly 99	Anomaly 98	Anomaly 97
1545	Anomaly 112	Anomaly 111	Anomaly 110	Anomaly 109	Anomaly 108	Anomaly 107	Anomaly 106	Anomaly 105
1546	Anomaly 120	Anomaly 119	Anomaly 118	Anomaly 117	Anomaly 116	Anomaly 115	Anomaly 114	Anomaly 113
1547	Anomaly 128	Anomaly 127	Anomaly 126	Anomaly 125	Anomaly 124	Anomaly 123	Anomaly 122	Anomaly 121

Each bit corresponds to an anomaly. To detect an anomaly set its bit to zero, to ignore the anomaly set it to one.

Anomaly Results

byte	D7	D6	D5	D4	D3	D2	D1	D0
1548								

Anomaly error number found. This byte contains a zero if no anomaly is found; otherwise it has the anomaly number of the anomaly shown in the checklist above. The anomaly number is cleared from this byte after it is read.

Windows Operation for Anomaly Detection

After clicking GO:

1. Load the anomaly checklist.
2. Clear the anomaly results window.
3. Send the Start Anomaly Detection command. (Display **ACTIVE** in the status bar)
4. Poll the anomaly results byte every second.
5. If the anomaly results byte is zero, don't display anything in the results window.
6. If the anomaly results byte is non-zero, display the anomaly name that corresponds to the anomaly number.

After clicking STOP:

1. Send the Stop Anomaly Detection command. (Display IDLE in the status bar)

The default anomaly checklist should be all anomalies checked (all bits set to zero in the anomaly checklist).

Master

Address

800	Master Control
801	Master Status
802	Master Address Space
803	Master Address Size
804 - 811	Master Start Address
812	Master Start Address Length
813	Master Data Size
814	Master Data Cycle
815	Master Data Type
816 - 847	Master Data

Master Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
800						CMD2	CMD1	CMD0

CMD2	CMD1	CMD0	Description
0	0	0	None
0	0	1	Read
0	1	0	Write
0	1	1	Display
1	0	0	Verify
1	0	1	Test
1	1	0	Compare

Master Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
801	ACTIVE	NO DSEL	NO TRDY	NO GNT		STATUS2	STATUS1	STATUS0

ACTIVE Master transfer in progress = 1

STATUS2	STATUS1	STATUS0	Description
0	0	0	None
0	0	1	Pass (for Verify, Test and Compare)
0	1	0	Fail (for Verify, Test and Compare)

NO TRDY No Target Acknowledge
NO DSEL No Device Select
NO GNT No Bus Grant

Display status in a small box in command section of master window.

Master Address Space

byte	D7	D6	D5	D4	D3	D2	D1	D0
802							SPACE1	SPACE0

SPACE1	SPACE0	Description
0	0	Memory
0	1	I/O
1	0	Configuration

Master Address Size

byte	D7	D6	D5	D4	D3	D2	D1	D0
803							SIZE1	SIZE0

SIZE1	SIZE0	Description
0	0	8
0	1	16
1	0	32
1	1	64

Master Start Address

byte	D7	D6	D5	D4	D3	D2	D1	D0
804-811								

Start Address – byte 804 is LSB

Master Start Address Length

byte	D7	D6	D5	D4	D3	D2	D1	D0
812								

Address Length max 32

Master Data Size

byte	D7	D6	D5	D4	D3	D2	D1	D0
813							SIZE1	SIZE0

SIZE1	SIZE0	Description
0	0	8
0	1	16
1	0	32
1	1	64

Master Data Cycle

byte	D7	D6	D5	D4	D3	D2	D1	D0
814								CYCLE

CYCLE 0 = Single, 1 = Burst

Master Data Type

byte	D7	D6	D5	D4	D3	D2	D1	D0
815							TYPE1	TYPE0

TYPE1	TYPE0	Description
0	0	Value
0	1	File
1	0	Target

Master Data

byte	D7	D6	D5	D4	D3	D2	D1	D0
816-847								

Master Data – byte 816 is LSB

Windows Operation for Master

After clicking GO:

1 Clear the results window.
2. For a Write command load master data into setup memory.
3. Load the master parameters into setup memory including the start address and length. The length cannot be greater then 32. If more then 32 bytes are needed repeat this procedure changing the start address.
4. Send the Start Exerciser command. (Display **ACTIVE** in the status bar)
5. Read the status byte to check if master is finished and if there were any errors.
6. Display any errors in results window. For the Display command show master data read in setup memory in the display window.
7. If the Repeat field is greater then 1 repeat these steps.
8. When all the bytes are complete display IDLE in the status bar

After clicking STOP:

9. Send the Stop Exerciser command to abort the current command. (Display IDLE in the status bar)

Target

Address

1490-1497	Target Bus Address
1498	Target Bus Address Length
1499	Target Bus Address Parameters
1500	Target Bus Data Parameters
1501	Target Control
1502	Target Status
1503-1510	Target Address
1511	Target Address Length
1512	Target Data Value
1513	Target Data Parameters

Target Bus Address

byte	D7	D6	D5	D4	D3	D2	D1	D0
1490-1497								

Bus Start Address – byte 1490 is LSB

Target Bus Address Length

byte	D7	D6	D5	D4	D3	D2	D1	D0
1498								

Value 0-64, Units MB

Target Bus Address Parameters

byte	D7	D6	D5	D4	D3	D2	D1	D0
1499							SPACE1	SPACE0

SPACE1	SPACE0	Description
0	0	Memory
0	1	I/O
1	0	Configuration

Target Bus Data Parameters

byte	D7	D6	D5	D4	D3	D2	D1	D0
1500	WS3	WS2	WS1	WS0	RESPONSE1	RESPONSE0	SIZE1	SIZE0

WS3 - 0 Wait States, value = 0 to 15 decimal

RESPONSE1	RESPONSE0	Description
0	0	Normal
0	1	Retry
1	0	Disconnect

SIZE1	SIZE0	Description
0	0	8 bit
0	1	16 bit
1	0	32 bit
1	1	64 bit

Target Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
1501	TARGET ENABLE					CMD2	CMD1	CMD0

TARGET ENABLE 0=Disable, 1=Enable

CMD2	CMD1	CMD0	Description
0	0	0	None
0	0	1	Read
0	1	0	Write
0	1	1	Compare

Target Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
1502	ACTIVE					STATUS2	STATUS1	STATUS0

ACTIVE Target transfer in progress = 1

STATUS2	STATUS1	STATUS0	Description
0	0	0	None
0	0	1	Pass (for Verify, Test and Compare)
0	1	0	Fail (for Verify, Test and Compare)

Display status in a small box in command section of target window.

Target Address

byte	D7	D6	D5	D4	D3	D2	D1	D0
1503-1510								

Target Start Address – byte 1503 is LSB

Target Address Length

byte	D7	D6	D5	D4	D3	D2	D1	D0
1511								

Target Address Length - max 255

Target Data Value

byte	D7	D6	D5	D4	D3	D2	D1	D0
1512								

Target Address Length – 8 bit hex value

Target Data Parameters

byte	D7	D6	D5	D4	D3	D2	D1	D0
1513							TYPE1	TYPE0

TYPE1	TYPE0	Description
0	0	Value
0	1	File

Windows Operation for Target

Always send the Bus Address, Bus Data and Bus Control parameters to setup memory whenever a field is modified. This includes any setup memory parameters from addresses 1490 to 1501.

The rest of the parameters, setup memory addresses 1502 to 1513, deal with reading and writing to the target memory.
After clicking GO:

1. Display **ACTIVE** in the status bar.
2. For a Write command perform a Write Target Memory command. For a Read command perform a Read Target Memory command and display the data in the display window.
3. When all the bytes are complete display IDLE in the status bar.

After clicking STOP:

10. Stop any read or write to target memory. (Display IDLE in the status bar)

Stimulus – Conditions and Controls

The stimulus function drives user defined signal patterns on the bus based on bus events. There are 16 stimulus conditions (S1 thru S16) and 16 stimulus control levels (SL1 thru SL16). Each stimulus level defines an event to look for (EV1 thru EV8) and the stimulus to drive, another level to jump to, a wait and duration time if the event occurs or does not occur. The stimulus definitions are very similar to the capture definitions.

Stimulus Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
929	ACTIVE				SL3	SL2	SL1	SL0

SL3-0 Current Active Stimulus Control Level SL(# + 1)
ACTIVE Stimulus in Progress = 1, Stimulus Stopped = 0

Stimulus Conditions

930 - 959	Stimulus Condition S1
960 -989	Stimulus Condition S2
990 - 1019	Stimulus Condition S3
1020 - 1049	Stimulus Condition S4
1050 - 1079	Stimulus Condition S5
1080 - 1109	Stimulus Condition S6
1110 - 1139	Stimulus Condition S7
1140 - 1169	Stimulus Condition S8
1170 - 1199	Stimulus Condition S9
1200 - 1229	Stimulus Condition S10
1230 - 1259	Stimulus Condition S11
1260 - 1289	Stimulus Condition S12
1290 - 1319	Stimulus Condition S13
1320 - 1349	Stimulus Condition S14
1350 - 1379	Stimulus Condition S15
1380 - 1409	Stimulus Condition S16

Stimulus Condition Bit Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1								
2	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
3								
4	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
5								
6	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
7								
8	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
9								
10	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
11								
12	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
13								
14	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
15								
16	CBE7	CBE6	CBE5	CBE4	CBE3	CBE2	CBE1	CBE0
17								
18	RST	LOCK	IDSEL	DEVSEL	STOP	IRDY	TRDY	FRAME
19								
20	PAR64	PAR	ACK64	REQ64	GNT	REQ	SERR	PERR
21								
22			SBO	SDONE	INTD	INTC	INTB	INTA
23								
24	EXT7	EXT6	EXT5	EXT4	EXT3	EXT2	EXT1	EXT0
25								
26			CLK	TRST	TMS	TCK	TDO	TDI
27								

All signals (even bytes) can be set to a 0 or 1 and are followed by a mask (odd bytes). If a mask bit is set to 1 the corresponding signal is used otherwise it is a don't care.

Stimulus Events

1600 - 1629	Stimulus Event 1
1630 - 1659	Stimulus Event 2
1660 - 1689	Stimulus Event 3
1690 - 1719	Stimulus Event 4
1720 - 1749	Stimulus Event 5
1750 - 1779	Stimulus Event 6
1780 - 1809	Stimulus Event 7
1810 - 1839	Stimulus Event 8

Stimulus Event Bit Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1								
2	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
3								
4	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
5								
6	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
7								
8	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
9								
10	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
11								
12	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
13								
14	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
15								
16	CBE7	CBE6	CBE5	CBE4	CBE3	CBE2	CBE1	CBE0
17								
18	RST	LOCK	IDSEL	DEVSEL	STOP	IRDY	TRDY	FRAME
19								
20	PAR64	PAR	ACK64	REQ64	GNT	REQ	SERR	PERR
21								
22	ADR/DATA	ANOMALY	SBO	SDONE	INTD	INTC	INTB	INTA
23								
24	EXT7	EXT6	EXT5	EXT4	EXT3	EXT2	EXT1	EXT0
25								
26	Range 1	Range 0	CLK	TRST	TMS	TCK	TDO	TDI
27								

Stimulus Controls

1410 - 1414	Stimulus Control SL1
1415 - 1419	Stimulus Control SL2
1420 - 1424	Stimulus Control SL3
1425 - 1429	Stimulus Control SL4
1430 - 1434	Stimulus Control SL5
1435 - 1439	Stimulus Control SL6
1440 - 1444	Stimulus Control SL7
1445 - 1449	Stimulus Control SL8
1450 - 1454	Stimulus Control SL9
1455 - 1459	Stimulus Control SL10
1460 - 1464	Stimulus Control SL11
1465 - 1469	Stimulus Control SL12
1470 - 1474	Stimulus Control SL13
1475 - 1479	Stimulus Control SL14
1480 - 1484	Stimulus Control SL15
1485 - 1489	Stimulus Control SL16

Stimulus Control Bit Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	STIMULUS3	STIMULUS2	STIMULUS1	STIMULUS0	EVENT 3	EVENT 2	EVENT 1	EVENT0
1	JMP_ELSE3	JMP_ELSE2	JMP_ELSE1	JMP_ELSE0	JMP_THEN3	JMP_THEN2	JMP_THEN1	JMP_THEN0
2	OCCUR7	OCCUR6	OCCUR5	OCCUR4	OCCUR3	OCCUR2	OCCUR1	OCCUR0
3	DUR7	DUR6	DUR5	DUR4	DUR3	DUR2	DUR1	DUR0
4	WAIT7	WAIT6	WAIT5	WAIT4	WAIT3	WAIT2	WAIT1	WAIT0

EVENT 3	EVENT 2	EVENT1	EVENT0	
0	0	0	0	None
0	0	0	1	E1
0	0	1	0	E2
0	0	1	1	E3
0	1	0	0	E4
0	1	0	1	E5
0	1	1	0	E6
0	1	1	1	E7
1	0	0	0	E8
1	1	1	1	All

STIMULUS 3	STIMULUS 2	STIMULUS 1	STIMULUS0	
0	0	0	0	None
0	0	0	1	S1
0	0	1	0	S2
0	0	1	1	S3
0	1	0	0	S4
0	1	0	1	S5
0	1	1	0	S6
0	1	1	1	S7
1	0	0	0	S8

Windows Operation for Stimulus

After clicking GO:

1. Send the Start Stimulus command. (Display **ACTIVE** in the status bar)
2. Read the status byte to check if the stimulus is finished.
3. Display IDLE in the status bar if inactive.

After clicking STOP:
Send the Stop Stimulus command to abort the current test. (Display IDLE in the status bar).

Backplane Test

1550	Backplane Test Control
1551	Backplane Test Status
1552 - 1581	Backplane Test Signals

Backplane Test Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
1550								SENSITIVITY

SENSITIVITY - Normal=0, High=1 The High setting is a more sensitive test and is affected by any additional load on signals such as bus terminations.

Backplane Test Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
1551	ACTIVE							

ACTIVE Backplane Test in Progress = 1, Backplane Test Stopped = 0

Backplane Test Signal Bit Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1								
2	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
3								
4	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
5								
6	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
7								
8	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
9								
10	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
11								
12	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
13								
14	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
15								
16	CBE7	CBE6	CBE5	CBE4	CBE3	CBE2	CBE1	CBE0
17								
18	RST	LOCK	IDSEL	DEVSEL	STOP	IRDY	TRDY	FRAME
19								
20	PAR64	PAR	ACK64	REQ64	GNT	REQ	SERR	PERR
21								
22					INTD	INTC	INTB	INTA
23								
24								
25								
26								
27								

All signals (even bytes) are followed by a mask (odd bytes). Set a mask bit to test a signal. After the test, read the signal bits. A “1” indicates the signal is shorted to the test signal. A “1” in the test signal bit indicates that the signal could be driven. A “0” in the test signal bit indicates that the signal could be driven.

Windows Operation for Backplane Test

After clicking GO:

1. Clear the Backplane Test Results Window.
2. Set the mask bit for the signal to be tested.
3. Display the signal name in the results window.
4. Send the Start Backplane Test command. (Display **ACTIVE** in the status bar)
5. Read the Backplane Test Status ACTIVE bit. When zero read the Backplane Test Signal bits.
6. If the test signal bit is “0’ display “ - Cannot Drive Signal”. Continue to step 9.
7. If another bit is “1’ display the names of the signals followed by the word “SHORTED”. Continue to step 9.
8. If no bits are set to “1” then display “- OK”.
9. Advance to the next test signal. Go to step 2.

After clicking STOP:

11. Send the Stop Backplane Test command. (Display IDLE in the status bar)

The default backplane test checklist should be all signals checked.

Configuration Scan

Address

850	Configuration Scan Control
851	Configuration Scan Status
852 - 915	Configuration Data
916 - 923	Configuration Address

Configuration Scan Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
850						BACK	NEXT	FIND

FIND set if first time scan. Starts scan at beginning of configuration space.
NEXT set to find next device.
BACK set to find previous device.
Note: Only 1 bit can be set at a time.

Configuration Scan Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
851	ACTIVE				FUNCTION2	FUNCTION1	FUNCTION0	DEVICE FOUND

ACTIVE Configuration scan in progress = 1
DEVICE FOUND Device is found = 1
FUNCTION Function Number

Configuration Data

byte	D7 D6 D5 D4 D3 D2 D1 D0
852	VENDOR ID (lsb)
853	VENDOR ID (msb)
854	DEVICE ID (lsb)
855	DEVICE ID (msb)
856	COMMAND (lsb)
857	COMMAND (msb)
858	STATUS (lsb)
859	STATUS (msb)
860	REVISION ID
861	CLASS CODE (lsb)
862	CLASS CODE
863	CLASS CODE (msb)
864	CACHE LINE SIZE
865	LATENCY TIMER
866	HEADER TYPE
867	BIST
868 - 915	(The definition of these bytes is based on the HEADER TYPE - see PCI spec)

Configuration Address

byte	D7	D6	D5	D4	D3	D2	D1	D0
916-923								

Configuration Address – byte 916 is LSB

Windows Operation for Configuration Scan

After clicking FIND or NEXT or BACK:

1. Clear the Header and Analysis windows.
2. Set the Configuration Control byte for FIND, NEXT or BACK.
3. Send the Start Configuration Scan command. (Display **ACTIVE** in the status bar)
4. Read the status byte to check if the scan is finished and a device was found. If no device was found, display the message “No Device Found” in the Analysis Window.
5. Display the Configuration data and Device address.
6. Display IDLE in the status bar.

After clicking STOP:
Send the Stop Configuration Scan command to abort the current command. (Display IDLE in the status bar)

Compliance Test

Address

1516	Compliance Control
1517	Compliance Status
1518 -1521	Compliance Test

Compliance Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
1516						SKIP TEST	CONTINUE	INSTRUCTION PRESENT

Instruction Present = 1 if there is an instruction with a compliance test
Continue = 1 if Continue button pressed
Skip Test = 1 if Skip Test button is pressed

Compliance Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
1517	ACTIVE	CODE3	CODE2	CODE1	CODE0		FAIL	PASS

ACTIVE Compliance test in progress = 1
PASS Test Passed = 1
FAIL Test Failed = 1
CODE Error Code Number

Compliance Test

byte	D7	D6	D5	D4	D3	D2	D1	D0
1518								
1519								
1520								
1521								

Coded name of the test to be performed.
For example the test TPW4: byte 1518 = T, byte 1519 = P, 1520 = W, byte 1521 = 4. Space characters will pad blank bytes.

Windows Operation for Compliance Test

First click FIND or NEXT or BACK to find device to test. Use the Configuration Scan function. Display the Vendor name, Class, etc. If no device is found, display “No Device Found” in the Test Results Window.

After clicking GO:
1. Clear the Test Results windows.
2. Display TESTING in the status bar
3. Display the name of the test in the test results window.
4. Set the Compliance Test bytes with the test code.
5. Send the Start Compliance Test command.
6. Read the status byte to check if the test is finished and display the test result (PASS or FAIL).
7. Go back to step 3 for each test that is checked.
8. Display IDLE in the status bar.

After clicking STOP:
Send the Stop Compliance Test command to abort the current test. (Display IDLE in the status bar) Also use STOP during configuration scanning to abort scan.

Self Test

Address

480	Self Test Control
481	Self Test Status
482	Self Test Results

Self Test Control

byte	D7	D6	D5	D4	D3	D2	D1	D0
480	TEST7	TEST6	TEST5	TEST4	TEST3	TEST2	TEST1	TEST0

TEST7-0 Selects the test to be performed

Self Test Status

byte	D7	D6	D5	D4	D3	D2	D1	D0
481	ACTIVE						FAIL	PASS

ACTIVE Self test in progress = 1
PASS Test Passed = 1
FAIL Test Failed = 1

Self Test Results

byte	D7	D6	D5	D4	D3	D2	D1	D0
482	CODE7	CODE6	CODE5	CODE4	CODE3	CODE2	CODE1	CODE0

CODE7-0 Test result code

Windows Operation for Self Test

After clicking GO:
1. Clear the Test Results windows.
2. Display TESTING in the status bar
3. Display the name of the test in the test results window.
4. Set the Self Test Control byte with the test code.
5. Send the Start Self Test command.
6. Read the status byte to check if the test is finished and display the test result (PASS or FAIL).
7. Go back to step 3 for each test that is checked.
8. Display IDLE in the status bar.

After clicking STOP:
Send the Stop Self Test command to abort the current test. (Display IDLE in the status bar).

Trace Memory Definitions

This is the basic bit definition of the trace data when performing a Read Trace command. This structure repeats for every sample requested.

Basic Trace Data Definition

byte	D7	D6	D5	D4	D3	D2	D1	D0
0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
1	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8
2	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
3	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24
4	AD39	AD38	AD37	AD36	AD35	AD34	AD33	AD32
5	AD47	AD46	AD45	AD44	AD43	AD42	AD41	AD40
6	AD55	AD54	AD53	AD52	AD51	AD50	AD49	AD48
7	AD63	AD62	AD61	AD60	AD59	AD58	AD57	AD56
8	CBE7	CBE6	CBE5	CBE4	CBE3	CBE2	CBE1	CBE0
9	RST	LOCK	IDSEL	DEVSEL	STOP	IRDY	TRDY	FRAME
10	PAR64	PAR	ACK64	REQ64	GNT	REQ	SERR	PERR
11	ADR/DATA		SBO	SDONE	INTD	INTC	INTB	INTA
12	EXT7	EXT6	EXT5	EXT4	EXT3	EXT2	EXT1	EXT0
13	T_UNIT1	T_UNIT0	CLK	TRST	TMS	TCK	TDO	TDI
14	T7	T6	T5	T4	T3	T2	T1	T0
15	T15	T14	T13	T12	T11	T10	T9	T8
16	ANOM7	ANOM6	ANOM5	ANOM4	ANOM3	ANOM2	ANOM1	ANOM0
17	EXP7	EXP6	EXP5	EXP4	EXP3	EXP2	EXP1	EXP0

T0 thru T15 is a time measurement and T_UNIT1,0 are the units defined as follows:

T_UNIT1	T_UNIT0	
0	0	ns
0	1	us
1	0	ms
1	1	sec

Startup Operation

General

At power up all setup memory locations are cleared to zero (except for occurrence counts (=1) and board information). The bit definitions have been selected so the value of zero makes sense for a startup condition. For example, all trace conditions are don't care, trigger position is ½, sample clock is SYNC, etc.

Assume that the RS-232 port is initially set at 9600 baud, 8 bit, 1 stop bit, no parity.