

CAPTURING PCI BUS TRANSACTIONS

USING FUTUREPLUS SYSTEMS PASSIVE PCI ANALYSIS PROBES AND AGILENT TECHNOLOGIES LOGIC ANALYZERS

© 2000 FUTUREPLUS SYSTEMS

This application note is designed to help PCI Analysis probe users trigger their logic analyzers in order to capture PCI bus transactions. Effectively using the trigger specification of the logic analyzer can be a big help in tracking down hard to find bugs. The logic analyzer memory is not infinite so you want to catch the bug even if it occurred much after or much before the event you're triggering on. The PCI Analysis probe software comes with pre-configured symbols and resource terms to help get you started.

Filtering Data before it is acquired

To help filter out unwanted PCI bus cycles before they are acquired by the logic analyzer, three resource terms have been defined in the trigger menu and included in the default trigger specification.

They are:

Resource Terms	FRAME	IRDY	TRDY	STOP	DEVSEL
IDLE	1	1	X	X	X
T_WAIT	X	0	1	1	0
DF_WAIT	0	0	X	1	1

all other labels are DON'T CARE (X)

IDLE is defined per the PCI Bus Specification as when both FRAME# and IRDY# are de-asserted. T_WAIT is defined as a target initiated wait state. DF_WAIT is defined as FRAME# and IRDY# asserted, DEVSEL released and STOP# released. This filters out the case of a slow target asserting DEVSEL# while FRAME# is asserted. Using these terms as part of the trigger specification can greatly enhance the readability of the state listing and save logic analyzer acquisition memory.

There are two additional wait states that often occur on the PCI local bus. The wait state that occurs as a result of FRAME# being asserted and both IRDY# and TRDY# released following the command/address phase cannot be filtered out by the trigger specification. The logic analyzer cannot distinguish between this state and a valid command/address state. This state will be listed as "WAIT-TARGET AND INITIATOR NOT READY". This state can be removed from the resulting acquired state listing by using the INVASM OPTIONS feature explained later in this application note. The other wait state that is often seen is that when DEVSEL is not asserted while FRAME# is released. D_WAIT can be defined as a resource term in the trigger menu and can be included in the trigger specification in order to filter out these states. It was deleted from the default trigger specification because it can cause Inverse Assembly errors in the case of a master abort. A master abort is not common but is seen at boot time during configuration transactions. The user can include D_WAIT instead of DF_WAIT in the "While Storing"

specification by selecting COMBINATION and then negating the term D_WAIT and “anding” it into the default term ≠IDLE* ≠T_WAIT*. The term ≠DF_WAIT can then be turned off.

Triggering on the first occurrence of a transaction

Figure 1 shows a STATE mode trigger specification that will trigger on the first occurrence of an interrupt acknowledge cycle. Note that the first state sequence level has been set to use a COMBINATION of terms as the object of the “While Storing”. The combination of ≠IDLE • ≠T_WAIT • ≠DF_WAIT simply means that those states will not be stored. The term XACTION has been set to INTACK. This was done using the pre-configured symbols that were supplied for the CYCLE variable by the PCI Analysis probe software. All the transaction types for the PCI bus were given symbols under the CYCLE variable. To select these symbols press the button that corresponds to the intersection of the CYCLE column and the resource term row you wish to define. You will then see a list of PCI cycles that can be selected.

Figure 1

100/500MHz LA A
Trigger 1
Cancel
Run

State Sequence Levels		Timer		
		1	2	
1	While storing "≠IDLE•≠T_WAIT•≠DF_WAIT" Find "≠FRAME" 1 time	-	-	Arming Control
2	While storing "≠IDLE•≠T_WAIT•≠DF_WAIT" TRIGGER on "XACTION" 1 time	-	-	Acquisiti Control
3	Store "≠IDLE•≠T_WAIT•≠DF_WAIT"	-	-	Count Time
3		-	-	Modify Trigger

	CYCLE	ADDR	FRAME	DEVSEL
←Label→	Symbol	Hex	Binary	Binary
↑Terms↓				
STOP	absolute XXXXXXXX00	XXXXXXXX	X	0
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1
XACTION	INTACK	XXXXXXXX	0	1
FRAME	absolute XXXXXXXXXX	XXXXXXXX	0	X

Figure 2 shows the resulting state listing.

Figure 2

100/500MHz LA A Listing 1 Invasm Options Cancel Run

Markers Time Trig to X 704.9 us Trig to 0 704.9 us X to 0 0 s

Label>	FUTUREPLUS SYSTEMS c 1996	Time	GNT	R
Base>	PCI BUS TRANSACTIONS REV 2.4	Relative	Bin	B
-5	I/O READ ADR=00000064	1.368 us	1	1
-4	WAIT-NO DEVICE SELECT	32 ns	1	1
-3	WAIT-NO DEVICE SELECT	40 ns	1	1
-2	WAIT-NO DEVICE SELECT	32 ns	1	1
-1	D32=xxxxxxx74	1.064 us	1	1
0	INTERRUPT ACK CYCLE	6.408 us	1	1
1	WAIT-NO DEVICE SELECT	32 ns	1	1
2	STOP-NO DATA XFERED-RETRY	32 ns	1	1
3	INTERRUPT ACK CYCLE	136 ns	1	1
4	WAIT-NO DEVICE SELECT	32 ns	1	1
5	STOP-NO DATA XFERED-RETRY	32 ns	1	1
6	INTERRUPT ACK CYCLE	128 ns	1	1

Trigger and save only Configuration Transactions

In the previous example the trigger specification was used to trigger on a certain transaction and save the bus traffic after and before that transaction occurred minus the IDLE and target initiated wait states. In some cases, engineers working on a particular problem want to acquire only a certain type of transaction and no others. This can be a tricky request since maneuvering the trigger menu can be at times a challenge.

Figure 3

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2
1	While storing "no state" Find "IDLE" 1 time	-	-	-
2	While storing "no state" TRIGGER on "XACTION" 1 time	-	-	-
3	Store "≠T_WAIT≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 2	-	-	-

Arming Control
Acquisition Control
Count Time
Modify Trigger

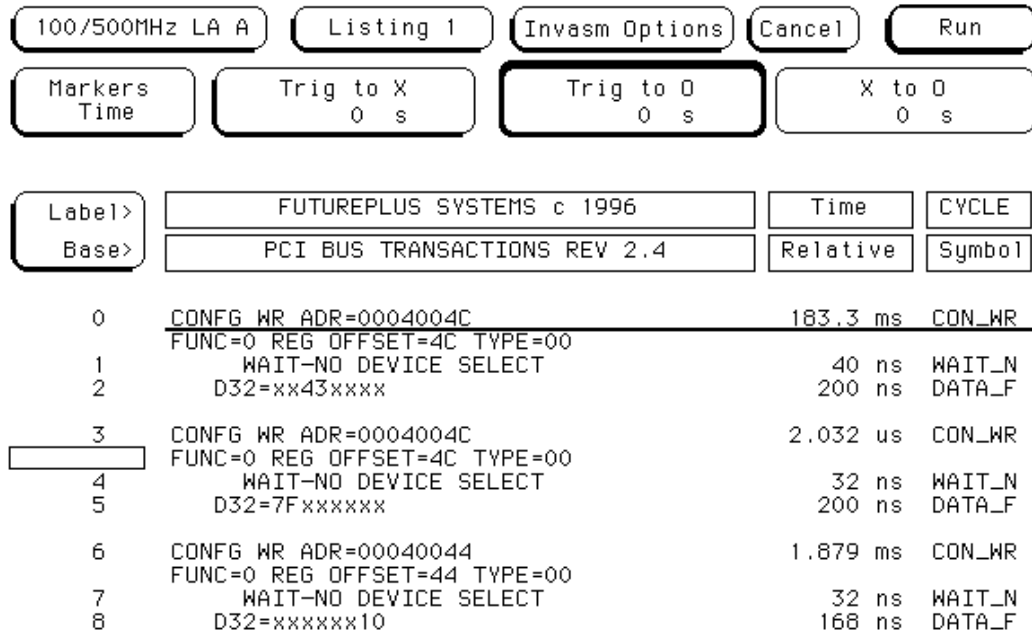
Label	CYCLE	ADDR	FRAME	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
STOP	absolute XXXXXXXX00	XXXXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXXXX	X	1	a
XACTION	CONFIG_XACTIONS	0XXXXXXXXX	0	1	a
FRAME	absolute X0XXXXXXXX	XXXXXXXXXX	0	X	a

Figure 3 shows the trigger menu for a trigger to capture only Configuration Reads and Configuration Writes. The terms IDLE, T_WAIT and DF_WAIT are defined by the PCI Analysis probe software and used in the above trigger to indicate the PCI cycles that will not be stored. In most cases these cycles are not interesting to capture and filtering them out saves valuable trace memory. The term XACTION can be easily defined by selecting the button that corresponds to the CYCLE label and then select CONFIG_XACTIONS from the list of cycle types defined by the PCI Analysis probe software. The end of transaction case is handled by the terms END_DAT1, STOP and END_DAT2. END_DAT1 refers to a normal transaction termination. The term STOP refers to those conditions where STOP is asserted and END_DAT2 refers to the target abort case. Figure 4 shows the resulting state listing.

This trigger is not bullet proof. The logic analyzer trigger specification is level sensitive and due to this fact there exists the possibility of acquiring master initiated wait states as valid address cycles. This can be minimized by qualification of the address. In systems without a bridge only type 0 configuration transactions (bits 1 and 0=0) can occur. In a system with a bridge type 0 and type 1 configuration transactions can be found (AD[1:0]=01, 00). Also AD[31:11] will have only one bit set during the address phase of a configuration cycle. This is the IDSEL pin being mapped to an address bit.

In the example trigger shown above the upper AD byte has been set to 0 as has bit 1. This will help qualify the trigger and help avoid acquiring false cycles.

Figure 4



It is interesting to note that even though Idol's are filtered out of the listing according to the trigger, they still appear once between each transaction. This is due to sequence level 3 in the trigger specification. The statement "on anystate" allows for one intervening cycle to be captured and displayed.

Save a certain transaction type (no trigger)

The need often arises to save only a certain transaction type or types and to not actually trigger on an event. The logic analyzer trigger menu will not allow you to delete the sequence level that has the word *trigger* embedded in it. This makes the task of fooling the logic analyzer a bit confusing. The key is to create a loop in the sequence levels prior to the sequence level that has the word trigger in it. Figure 5 shows a trigger that saves only configuration transactions and does not trigger on anything. It should be noted that the Acquisition Control Field will control where the trace starts and ends. Also the analyzer will not stop and display the acquired data until the trigger condition is met or the acquisition memory is full. If the store conditions set do not fill the memory simply press STOP and the analyzer will display the captured data.

This trigger does have its problems. If the system you are observing has back to back cycles where FRAME# is asserted and IRDY# is released this trigger will mistakenly acquire the second of these cycles as a valid address cycle. Only the first assertion of FRAME# is the valid address cycle. Qualification of the address is the easiest solution.

Figure 5

100/500MHz LA D Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
1	While storing "no state" Find "XACTION" 1 time	-	-		Arming Control
2	While storing "≠T_WAIT≠DF_WAIT" Then find "no state" 1 time Else on "IDLE" go to level 1	-	-		Acquisition Control
3	While storing "no state" TRIGGER on "no state" 1 time	-	-		Count Time
					Modify Trigger

Label	CYCLE	ADDR	STOP	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
END_DAT2	TARGET_ABORT	XXXXXXXX	0	1	a
J	MEM_WR	XXXXXXXX	1	1	a
XACTION	CONF IG_XACTIONS	XXXXXXXX	1	1	a
DF_WAIT	WAIT_NODEVSL/F_0	XXXXXXXX	1	1	a

Trigger and save only Memory Read and Write transactions

Figure 6 shows a trigger defined to trigger on the first Memory Read or Memory Write transaction and then save only Memory Read and Memory Write transactions. Note that the terms XACTION was set to MEM_XACTIONS by selecting the intersection of the XACTION term row and the CYCLE column.

As in the previous trigger the end of transaction is indicated by END_DAT1, STOP and END_DAT2.

Figure 7 shows the resulting state listing.

Figure 6 Trigger to store only Memory transactions

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
1	While storing "no state" Find "IDLE" 1 time	--	--		Arming Control
2	While storing "no state" TRIGGER on "XACTION" 1 time	--	--		Acquisition Control
3	Store "≠T_WAIT≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 2	--	--		Count Time
					Modify Trigger

Label	CYCLE	ADDR	FRAME	DEVSEL	
Terms	Symbol	Hex	Binary	Binary	
STOP	absolute XXXXXXXX00	XXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1	a
XACTION	MEM_XACTIONS	XXXXXXXX	0	1	a
FRAME	absolute X0XXXXXXXX	XXXXXXXX	0	X	a

Figure 7 Resulting state listing

100/500MHz LA A	Listing 1	Invasm Options	Cancel	Run
Markers Time	Trig to X 0 s	Trig to 0 0 s	X to 0 0 s	
Label>	FUTUREPLUS SYSTEMS c 1996	Time	CYCLE	
Base>	PCI BUS TRANSACTIONS REV 2.4	Relative	Symbol	
0	MEM READ ADR=FFFFFFF0	182.9 ms	MEM_RD	
1	D32=00E05BEA	3.536 us	DATA_F	
2	MEM READ ADR=FFFFFFF4	200 ns	MEM_RD	
4	D32=2F3430F0	3.536 us	DATA_F	
5	MEM READ ADR=FFFFFFF8	296 ns	MEM_RD	
6	D32=392F3531	3.568 us	DATA_F	
7	MEM READ ADR=FFFFFFFC	200 ns	MEM_RD	
9	D32=00FC0034	3.536 us	DATA_F	
10	MEM READ ADR=000FE058	464 ns	MEM_RD	
11	D32=E9000000	3.536 us	DATA_F	

In this listing screen the term CYCLE added to the display. This can be useful in determining the type of PCI cycle present on the bus. Also note that INVASM OPTIONS has been used to remove all WAIT and IDLE states acquired by this trigger.

This trigger does have its problems. If the system you are observing has back to back cycles where FRAME# is asserted and IRDY# is released this trigger will mistakenly acquire the second of these cycles as a valid address cycle. Only the first assertion of FRAME# is the valid address cycle. Qualification of the address is the easiest solution.

Trigger and save only I/O Read and Write transactions

This trigger is the same as the previous one except the XACTION term has been changed to I/O transactions and the acquired wait states have not been removed via the INVASM OPTIONS menu.

Figure 8 Trigger to capture only I/O transactions

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2
1	While storing "no state" Find "IDLE" 1 time	--	--	Arming Control
2	While storing "no state" TRIGGER on "XACTION" 1 time	--	--	Acquisition Control
3	Store "≠T_WAIT≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 2	--	--	Count Time
				Modify Trigger

Label	CYCLE	ADDR	FRAME	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
STOP	absolute XXXXXXXX00	XXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1	a
XACTION	ID_XACTIONS	0000XXXX	0	1	a
FRAME	absolute X0XXXXXXXX	XXXXXXXX	0	X	a

Figure 9 Resulting state listing

100/500MHz LA A Listing 1 Invasm Options Cancel Run

Markers Trig to X Trig to 0 X to 0

Time 2.768 us 2.768 us 0 s

Label	Base	Time	GNT	R
	FUTUREPLUS SYSTEMS c 1996			
	PCI BUS TRANSACTIONS REV 2.4	Relative	Bin	B
0	I/O READ ADR=00000073	183.1 ms	1	1
1	WAIT-NO DEVICE SELECT	32 ns	1	1
2	WAIT-NO DEVICE SELECT	32 ns	1	1
3	WAIT-NO DEVICE SELECT	40 ns	1	1
4	D32=00XXXXXX	1.064 us	1	1
5	I/O WRITE ADR=00000073	432 ns	1	1
6	WAIT-NO DEVICE SELECT	32 ns	1	1
7	WAIT-NO DEVICE SELECT	32 ns	1	1
8	WAIT-NO DEVICE SELECT	40 ns	1	1
9	D32=00XXXXXX	1.064 us	1	1
10	I/O READ ADR=00000064	23.21 us	1	1
11	WAIT-NO DEVICE SELECT	40 ns	1	1
12	WAIT-NO DEVICE SELECT	32 ns	1	1

Trigger on the first Memory Read and then save only Memory Write transactions

Figures 10 and 11 show a single trigger specification that triggers on the first Memory Read transaction and then stores only Memory Write transactions after that point. Figure 12 shows the resulting state listing display.

Figure 10

100/500MHz LA D Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
1	While storing "≠IDLE≠T_WAIT≠DF_WAIT" TRIGGER on "XACTION" 1 time	-	-		Arming Control
2	While storing "≠IDLE≠T_WAIT≠DF_WAIT" Then find "END_DAT1+STOP+END_DAT2" 1 time	-	-		Acquisition Control
3	While storing "no state" Then find "J" 1 time	-	-		Count Time
					Modify Trigger

Label	CYCLE	ADDR	STOP	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
END_DAT2	TARGET_ABORT	XXXXXXXX	0	1	a
J	MEM_WR	XXXXXXXX	1	1	a
XACTION	MEM_RD	XXXXXXXX	1	1	a
DF_WAIT	WAIT_NODEVSL/F_0	XXXXXXXX	1	1	a

Figure 11

100/500MHz LA D Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
4	While storing "≠IDLE≠T_WAIT≠DF_WAIT" Then find "END_DAT1+STOP+END_DAT2" 1 time	-	-		Arming Control
5	While storing "no state" Then find "J" 1 time	-	-		Acquisition Control
6	Store "≠IDLE≠T_WAIT≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 3	-	-		Count Time
					Modify Trigger

Label	CYCLE	ADDR	STOP	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
END_DAT2	TARGET_ABORT	XXXXXXXX	0	1	a
J	MEM_WR	XXXXXXXX	1	1	a
XACTION	MEM_RD	XXXXXXXX	1	1	a
DF_WAIT	WAIT_NODEVSL/F_0	XXXXXXXX	1	1	a

Figure 12

100/500MHz LA D Listing 1 Cancel Run

Markers Time Trig to X 6.074 s Trig to 0 6.074 s X to 0 0 s

Label>	PCI BUS TRANSACTIONS	CYCLE
Base>	REV 2.3	Symbol
0	MEM READ ADR=FFFFFFF0	MEM_RD
1	D32=00E05BEA	DATA_FINALXFER
2	MEM WRITE ADR=000F2308	MEM_WR
3	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
4	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
5	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
6	STOP-NO DATA XFERED-RETRY	STOP_RETRY
7	MEM WRITE ADR=000F2308	MEM_WR
8	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
9	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
10	WAIT-NO DEVICE SELECT	WAIT_NODVSEL
11	STOP-NO DATA XFERED-RETRY	STOP_RETRY

Using Trigger Macro's

The 16500 series of logic analyzers contains *Trigger Macros* that can be invoked when editing the state sequence levels in the trigger screen. These macros can be customized using the cycle variable and other resource terms. In some PCI systems the Master will keep FRAME# asserted with IRDY# de-asserted for two cycles at the beginning of a transaction. The first of these two cycles is the valid and true Command/Address cycle. The second of these is a master initiated wait state. The PCI Analysis probe Inverse Assembler describes this as "WAIT-TARGET AND INITIATOR NOT READY" because neither the Master nor the Targets have their ready signals asserted. This cycle can be very difficult for the logic analyzer trigger specification to distinguish. In order to be sure and trigger on the first occurrence of FRAME# asserting use the trigger below.

Figure 13 Use of Trigger Macros

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	Arming Control
1	While storing "anystate" TRIGGER on "XACTION" immediately after "≠FRAME"	-	-		Acquisition Control
2	Store "≠T_WAIT+≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 1	-	-		Count Time
					Modify Trigger

Label	CYCLE	ADDR	FRAME	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
STOP	absolute XXXXXXXX00	XXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1	a
XACTION	MEM_XACTIONS	XXXXXXXX	0	1	a
FRAME	absolute X0XXXXXXXX	XXXXXXXX	0	X	a

Figure 14 shows the trigger in figure 13 expanded into the individual sequence levels. Macros can be expanded by selecting the MODIFY TRIGGER button to the right of the trigger menu display. This will allow "anystate" in sequence level 1 in figure 13 to be changed. Figure 15 shows just such changes.

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
1	While storing "anystate" Find "≠FRAME" 1 time	--	--		Arming Control
2	While storing "anystate" TRIGGER on "XACTION" 1 time Else on "≠XACTION+FRAME" go to level 1	--	--		Acquisition Control
3	Store "≠T_WAIT+≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 1	--	--		Count Time
					Modify Trigger

Label	CYCLE	ADDR	FRAME	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
STOP	absolute XXXXXXX00	XXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1	a
XACTION	CONFIG_XACTIONS	XXXXXXXX	0	1	a
FRAME	absolute XXXXXXXXX	XXXXXXXX	0	X	a

Figure 14 Trigger in figure 13 with the macros expanded

Figure 15 Expanded trigger macros changed

100/500MHz LA A Trigger 1 Cancel Run

State Sequence Levels		Timer	1	2	
1	While storing "no state" Find "≠FRAME" 1 time	--	--		Arming Control
2	While storing "no state" TRIGGER on "XACTION" 1 time Else on "≠XACTION+FRAME" go to level 1	--	--		Acquisition Control
3	Store "≠T_WAIT+≠DF_WAIT" On "END_DAT1+STOP+END_DAT2" go to level 1	--	--		Count Time
					Modify Trigger

Label	CYCLE	ADDR	FRAME	DEVSEL	S
Terms	Symbol	Hex	Binary	Binary	S
STOP	absolute XXXXXXX00	XXXXXXXX	X	0	a
END_DAT2	TARGET_ABORT	XXXXXXXX	X	1	a
XACTION	CONFIG_XACTIONS	XXXXXXXX	0	1	a
FRAME	absolute XXXXXXXXX	XXXXXXXX	0	X	a

The Acquisition Control Field

In the 16500 mainframe, the Acquisition Control field can be found on the right hand side of the trigger menu and can be used to access the Acquisition Control menu. The Acquisition Control menu is used to set the acquisition mode and trigger position within available memory.

The Acquisition Mode Field

The Acquisition Mode field toggles between Manual and Automatic. When set to Automatic in STATE mode, the trigger position is computed based on the sequence specification. In TIMING mode, the trigger position and the sample period is computed based on the sec/Div and the delay settings in the Waveform menu.

When the Acquisition Mode field is set to Manual, additional configuration fields become available. Use these fields to further qualify what data is stored.

The Trigger Position Field

The trigger Position field accesses a selection menu with the options of Start, Center, End or User Defined. When an option is selected, that point of the available memory is positioned relative to the trigger. In TIMING mode, the start of the acquisition can also be delayed.

Branches Taken Stored/Not Stored

The Branches Taken field is a toggle field that sets the analyzer to store, or not to store, the resource term that sent the analyzer off on a branch. The trigger specifications shown in this application note assume that this field is set to Branches Taken Stored.

For more detailed information please refer to the User's Reference of your Agilent Logic Analyzer.

Using Symbols

In the 16500 mainframe, the SYMBOLS button can be found in the upper right hand corner of the FORMAT menu. Once selected, the user can set up symbols for any label listed in the FORMAT menu. These symbols can be added to or deleted. The user can create new labels in the format menu and these labels can be any combination of PCI signals that are acquired by the PCI Analysis probe.

The CYCLE Label

The CYCLE label is defined in the format menu. The CYCLE variable is made up of the following PCI signals: TRDY#, FRAME#, IRDY#, C/BE(3:0), DEVSEL# and STOP#. This variable has 31 symbols defined that can be used to help make triggering, timing analysis and pattern filtering (16505A) easier. The following lists the bit pattern and the corresponding symbol.

Symbol	TRDY#	FRAME#	IRDY#	C/BE(3:0)	DEVSEL#	STOP#
INTACK	1	0	1	0000	1	1
SPEC_CYC	1	0	1	0001	1	1
I/O_RD	1	0	1	0010	1	1
I/O_WR	1	0	1	0011	1	1
RESVRD	1	0	1	0100	1	1
RESVRD	1	0	1	0101	1	1
MEM_RD	1	0	1	0110	1	1
MEM_WR	1	0	1	0111	1	1
RESRVD	1	0	1	1000	1	1
RESRVD	1	0	1	1001	1	1
CON_RD	1	0	1	1010	1	1
CON_WR	1	0	1	1011	1	1
MEMRDM	1	0	1	1100	1	1
DAD_CY	1	0	1	1101	1	1
MEMRDL	1	0	1	1110	1	1
MEMWRI	1	0	1	1111	1	1
IO_XACTION	1	0	1	001X	1	1
MEM_XACTION	1	0	1	011X	1	1
CONFIG_XACTION	1	0	1	101X	1	1
ADD_CYCLE	1	0	1	XXXX	1	1
DATA_XFER	0	0	0	XXXX	0	1
WAIT_TARGET	1	X	0	XXXX	0	1
WAIT_INITIATOR	0	X	1	XXXX	0	1
DATA_FINALXFER	0	1	0	XXXX	0	1
STOP_NOXFER	X	0	1	XXXX	0	0
STOP_DATAXFER	0	X	0	XXXX	0	0
STOP_RETRY	1	X	0	XXXX	0	0
TARGET_ABORT	1	0	1	XXXX	1	0
IDLE	X	1	1	XXXX	X	X
WAIT_NODVSEL	X	X	0	XXXX	1	1
WAIT_NODVSEL/F_0	X	0	0	XXXX	1	1

The ADDR Label

Adding symbols via the FORMAT menu to the ADDR label can also help find problems when doing system testing. This feature allows you to replace the address in the PCI inverse assembly display with a symbol. For example, the symbol could be a register name or a software routine name. With version 2.1 or later of the PCI Analysis probe software, the symbol name will appear in place of the address right in the PCI inverse assembly display. Figure 16 shows an example of a symbol inserted into the STATE listing display. In this case, the symbol is for the MST_INT_RQ_REG at address 20. Please note that the symbol functionality is limited in that it will not distinguish between an I/O address and a memory address. It will print the symbol whenever the address matches.

Figure 16

100/500MHz LA E	Listing 1	Invasm Options	Print	Run
Markers Time	Trig to X 768 ns	Trig to 0 768 ns	X to 0 0 s	
Label>	PCI BUS TRANSACTIONS	Time	C/B3_0	
Base>	REV 2.2	Relative	Hex	
4	INTERRUPT ACK CYCLE	54.90 ms	0	
5	D32=xxxxxx08	304 ns	E	
6	I/O READ ADR=00000073	9.064 us	2	
7	D32=23xxxxxx	1.168 us	7	
8	I/O WRITE ADR=MST_INT_RQ_REG	9.800 us	3	
9	D32=xxxxxx20	160 ns	E	
10	INTERRUPT ACK CYCLE	54.91 ms	0	
11	D32=xxxxxx08	296 ns	E	
12	I/O READ ADR=00000073	9.072 us	2	
13	D32=23xxxxxx	1.168 us	7	

Post Processing acquired PCI data

Once PCI transactions have been acquired, you need to be able to scan the display quickly in order to spot any anomalies. The INVASM OPTIONS feature of the state listing display can selectively filter out of the display selected PCI transactions. For example, if after acquiring a trace an engineer only wanted to see I/O transactions, the INVASM OPTIONS filters could suppress all other transactions from the display leaving only the I/O transactions and their correct time stamp. Now these transactions can be quickly viewed. The INVASM OPTIONS selections for filtering are:

- I/O Reads
- I/O Writes
- Configuration Reads

- Configuration Writes
- Memory Reads
- Memory Writes
- IDLE states
- WAIT states
- All other transactions (Interrupt Acknowledge and Special cycles)

Figure 13 shows the PCI INVASM OPTIONS menu. INVASM OPTIONS is new in REV 2.1 or better of the PCI Analysis probe software and only available on the 16550A, 16540/541 and 16555A in the 16500B main frame and the 166x series with REV 2.0 system software. To get to the INVASM OPTIONS menu select the INVASM OPTIONS button in the upper right hand corner of the state listing display.

Figure 13

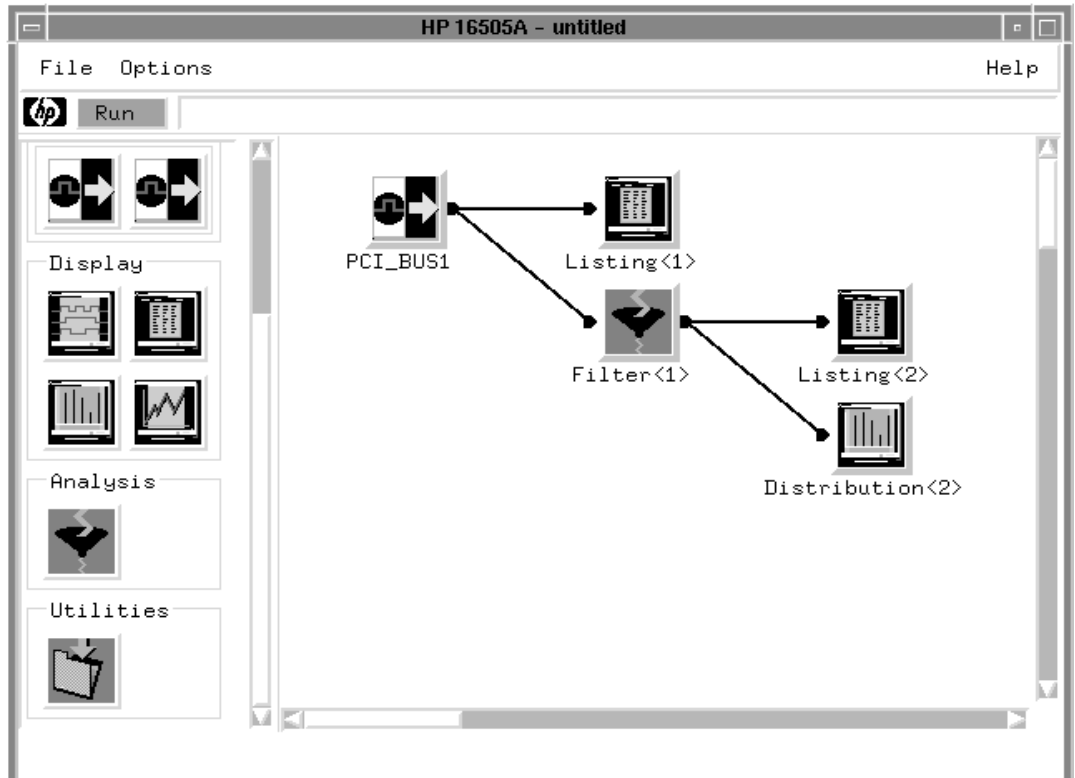
PCI Bus Inverse Assembly Options

Wait Cycles:	<input type="button" value="Show"/>	
I/O Reads:	<input type="button" value="Show"/>	
I/O Writes:	<input type="button" value="Show"/>	
Configuration Reads:	<input type="button" value="Show"/>	
Configuration Writes:	<input type="button" value="Show"/>	
Memory Reads:	<input type="button" value="Show"/>	
Memory Writes:	<input type="button" value="Show"/>	
Idle Cycles:	<input type="button" value="Suppress"/>	
All Other Transactions:	<input type="button" value="Show"/>	<input type="button" value="Done"/>

Post Processing PCI Cycles with the 16505A Pattern Filters

The 16505A Prototype Analyzer has additional features that help make analyzing PCI data easier. Shown below is how the Prototype Analyzer workspace can be configured during PCI test.

Figure 14



The 16505A Pattern Filters

The 16505A Pattern Filters can be used to filter out PCI cycles from the listing display. This is especially useful when using the 16555A cards that can acquire up to 1 million PCI cycles. Figure 15 shows an example of how the CYCLE variable can be used to filter out all of the data cycles from the acquired listing display using the 16505A Pattern Filters.

Figure 15

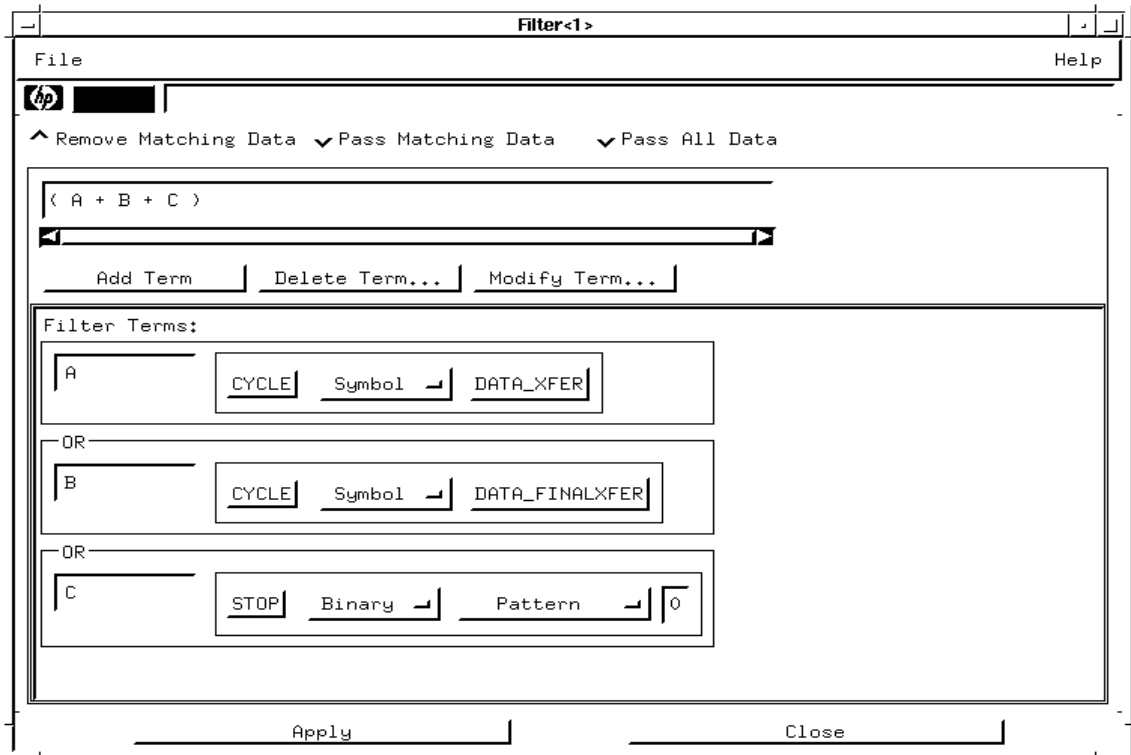


Figure 16 shows the Listing <1> display. The Listing<1> display has no pattern filters applied. Figure 17 shows Listing<2> display. Listing<2> has the pattern filters shown in figure 15 applied. The pattern filters have removed all of the data cycles from the display. This display can now be used to quickly pick out key access on the PCI.

Figure 16

State Number Decimal	ADDR Hex	PCI BUS TRANSACTIONS REV 2.2	CYCLE Symbol
7	00000000	INTERRUPT ACK CYCLE	INTACK
8	08080808	D32=xxxxxxxx08	DATA_FINALXFER
9	00000073	I/O READ ADR=00000073	I/O_RD
10	30FF80FF	D32=30xxxxxx	DATA_FINALXFER
11	00000020	I/O WRITE ADR=00000020	I/O_WR
12	00000020	D32=xxxxxxxx20	DATA_FINALXFER
13	00000000	INTERRUPT ACK CYCLE	INTACK
14	08080808	D32=xxxxxxxx08	DATA_FINALXFER
15	00000073	I/O READ ADR=00000073	I/O_RD
16	30FF80FF	D32=30xxxxxx	DATA_FINALXFER
17	00000020	I/O WRITE ADR=00000020	I/O_WR
18	00000020	D32=xxxxxxxx20	DATA_FINALXFER
19	00000000	INTERRUPT ACK CYCLE	INTACK
20	00000020	STOP-NO DATA XFERED-RETRY	STOP_RETRY
21	00000000	INTERRUPT ACK CYCLE	INTACK
22	00000020	STOP-NO DATA XFERED-RETRY	STOP_RETRY

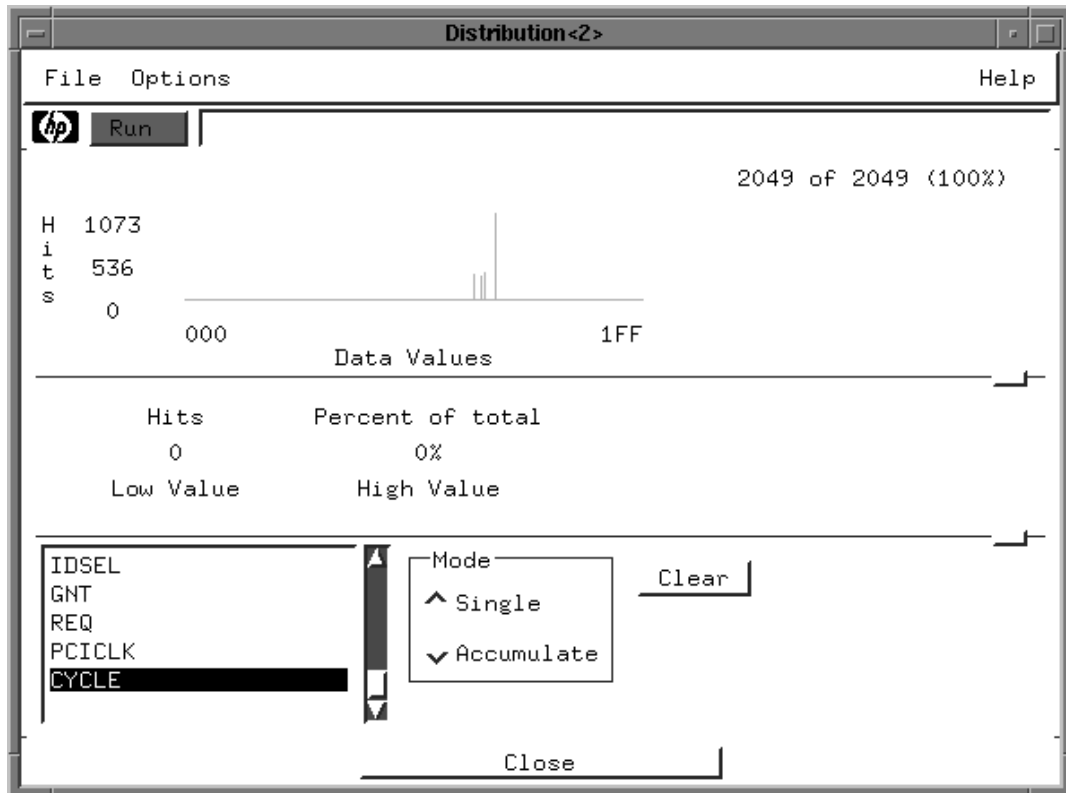
Figure 17

State Number Decimal	PCI BUS TRANSACTIONS REV 2.2	Time Relative
691	I/O WRITE ADR=00000020	200,000 ns
693	INTERRUPT ACK CYCLE	54,903 ms
695	I/O READ ADR=00000073	8,368 us
697	I/O READ ADR=00000073	304,000 ns
699	I/O READ ADR=00000073	296,000 ns
701	I/O READ ADR=00000073	304,000 ns
703	I/O READ ADR=00000073	296,000 ns
705	I/O WRITE ADR=00000020	10,136 us
707	INTERRUPT ACK CYCLE	54,907 ms
709	I/O READ ADR=00000073	8,376 us
711	I/O READ ADR=00000073	296,000 ns
713	I/O READ ADR=00000073	304,000 ns
715	I/O READ ADR=00000073	296,000 ns
717	I/O READ ADR=00000073	304,000 ns
719	I/O READ ADR=00000073	296,000 ns
721	I/O WRITE ADR=00000020	10,240 us
723	INTERRUPT ACK CYCLE	54,904 ms
725	I/O READ ADR=00000073	8,368 us
727	I/O WRITE ADR=00000020	10,136 us
729	INTERRUPT ACK CYCLE	54,907 ms
731	I/O READ ADR=00000073	9,200 us
733	I/O WRITE ADR=00000020	10,208 us
735	INTERRUPT ACK CYCLE	54,907 ms
737	I/O READ ADR=00000073	8,368 us

The 16505A Distribution Tool

The 16505A Distribution Tool can be used with or without the 16505A pattern filters. Figure 18 shows the Distribution Tool after the data has passed through the pattern filter shown in figure 15. By showing the distribution of the CYCLE variable the user can chart the transaction types and see the most frequently executed transaction types on the PCI.

Figure 18



Additional information of triggering can be found in Agilent Application note 1223 [Logic Analyzer Triggering Applications](#). This application note can be obtained from your local Agilent sales representative.

For more information on the PCI Analysis probe please contact:

FuturePlus Systems Corporation
TEL:719-278-3540
FAX:719-278-9586
internet: www.futureplus.com