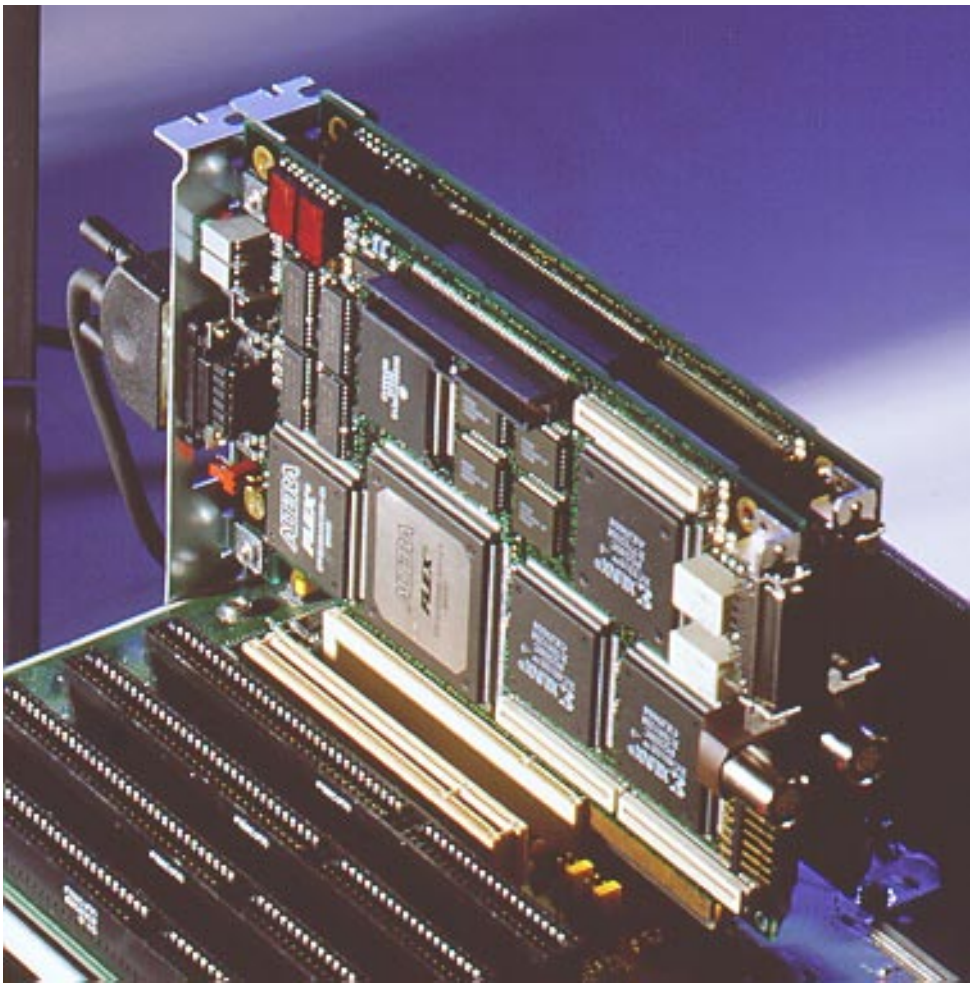
HP E2920 Software Demo

Version: 23. Jan 97



1.0 Introduction

This demo guide shows how the HP E2925A 33 MHz, 32 bit PCI Bus Exerciser & Analyzer, together with the HP E2970A Analyzer graphical user interface (GUI) and the HP E2971A Exerciser GUI, can be used in various applications. You don't need the HP E2925A hardware in order to work through this guide, as all the example setup files, as well as logic analyzer trace files, are available in the software.

1.1 Who should use this guide?

This demo guide is intended for the engineer who plans to use the HP E2920 PCI tools for his PCI bring-up, debug or validation tasks and wants to understand how this product works and how it can be integrated into his own test routines.

1.2 List of examples shown in this guide

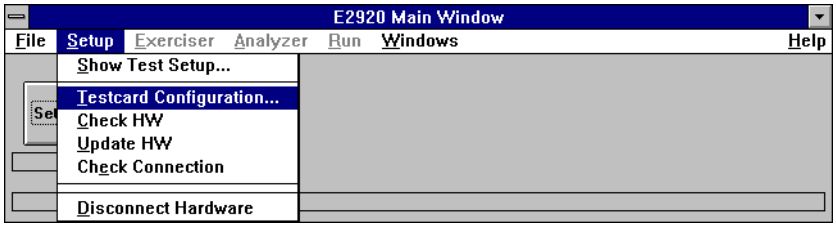
- Analyzing the PCI bus.
  - Example: Analyzing PCI traffic to a VGA graphics controller.
  - Example: Detecting and analyzing a protocol violation.
- Viewing and modifying system memory, I/O and config.
  - Example: Poking into VGA frame buffer memory.
- Deterministically generating PCI transactions.
  - Example: Generating master transactions using the GUI.
  - Example: Setting up target transactions using the GUI.
- Chip and system validation using a C test program.
  - Example: Using the C-API.
  - Example: Setting up a memory test using the C-API.

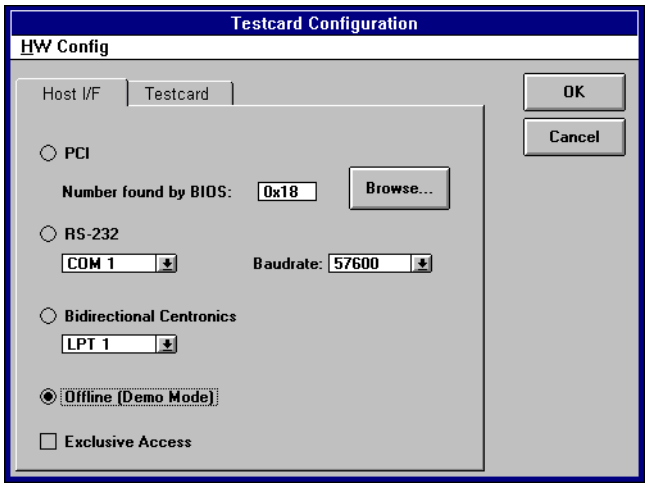
1.3 How to obtain the HP E2920 software

As you work through this guide, you should have the HP E2920 software (**Rev. 3.21.00** or later) installed on a Windows NT or Windows 95 system, and work with the software as you go through the guide. The software is downloadable from the world wide web at **http://www-europe.hp.com/dvt/products/E2925A**. Please refer to the installation procedure described on the CD cover or in the readme file on the web on how to install the software on your system, and to the online "**User Guide**" for a reference of all features of the product.

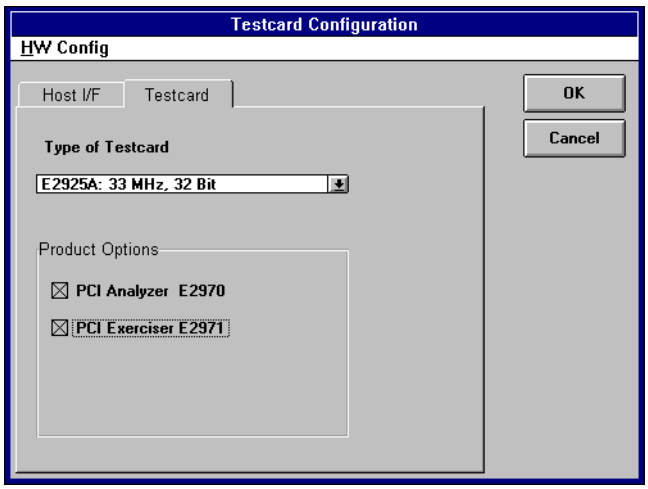
1.4 Setting up the HP E2920 software

In order to use the HP E2920 software without hardware, open the "**Setup->Testcard Configuration...**" window, select "**Offline (Demo Mode)**" in the "**Host I/F**" tab

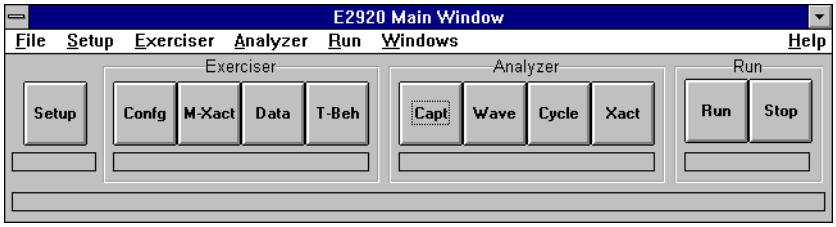




and check both the "**PCI Analyzer E2970**" and "**PCI Exerciser E2971**" checkboxes in the "**Testcard**" tab of the same window.



After changing the configuration, the main window should look like this.



2.0 Application Examples

The following application examples explain how the HP E2925A can be used in various analyzing, debugging and validation tasks. All the setup files (*.bst) and logic analyzer trace files (*.wfm) that are mentioned in the following text can be found under **<your_installation_directory>\samples\demo**. If you did not change the default setting during installation, <your_installation_directory> will be **c:\hpbest.30**.

2.1 Analyzing the PCI bus

If you are:

- **designing a PCI chip** and you need to do **bring-up or debugging,**
- using a third party **PCI chip** on your motherboard or adapter card which you need to **evaluate,**
- trying to **find the root cause of a failure** that occured during your **chip or system level validation,**
- writing and **debugging low level software** (e.g. BIOS code, device drivers),

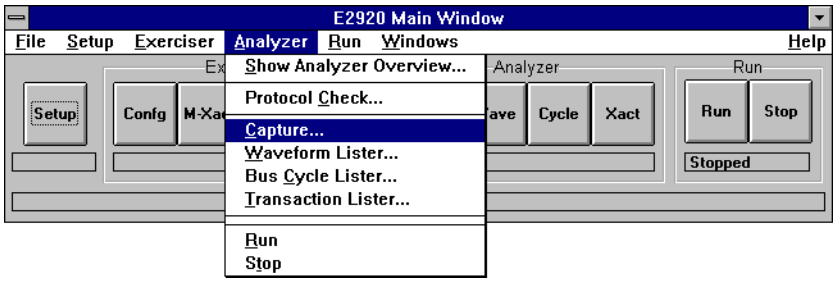
you probably need to **monitor the PCI bus** to find out whether your software generates the correct PCI transactions, and also whether your device-under-test reacts correctly, both on the **protocol and the data level**. Monitoring the PCI transactions of your device also allows you to judge its performance in relation to its partners.

The built-in 32k state logic analyzer of the HP E2925A allows you to capture PCI traffic and view it as a state waveform, a bus cycle listing or as a transaction listing. The following examples show you how to set up the analyzer and how to interpret the results.

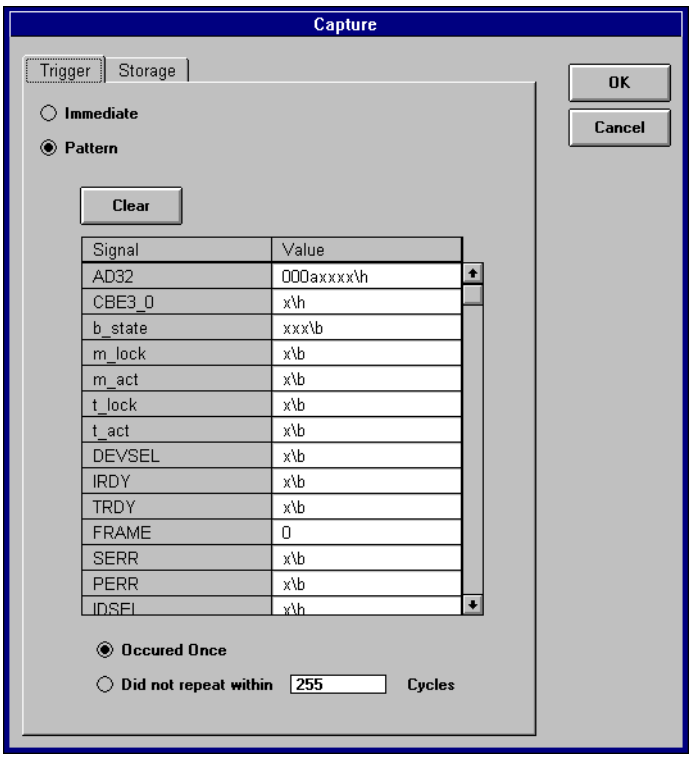
2.1.1 Example: Analyzing PCI traffic to a VGA graphics controller

This example shows how to set up the HP E2925A to trigger on a particular address range and capture PCI traffic that occurs around this trigger point. Afterwards, the captured data can be viewed in various levels of abstraction to analyze the PCI behavior of the participating devices (in this case a Host-PCI bridge and a PCI graphics controller).

To set up a trigger and storage qualifier for the built-in logic analyzer, use the "**Capt**" button in the main window or the "**Analyzer->Capture...**" command in the main menu.

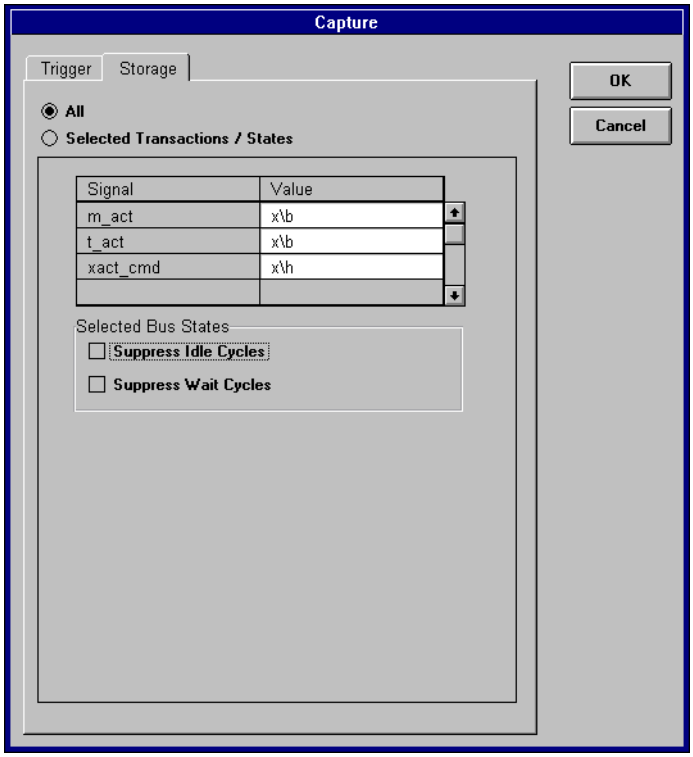


In the Capture window, select the "**Trigger**" tab and choose trigger on "**Pattern**", set the **AD32** field to **000axxxx\h** and the **FRAME** field to **0**. This tells the HP E2925A to trigger on FRAME being asserted with the AD matching the VGA graphics physical memory address.



Now, switch to the "**Storage**" tab of the "**Capture**" window. In this window, you can define the storage qualifier. The default is "**All**", which instructs the built-in logic analyzer to unconditionally capture one sample per PCI clock. Alternatively, you can choose "Selected Transactions/States" which allows you, for example, to suppress IDLE cycles on the PCI bus to be stored (i.e. PCI transactions are stored back-to-back). With the xact_cmd, m_act and t_act fields, you can further restrict what is captured in the trace memory by storing only particular transaction types (e.g. only memory writes) or storing only transactions where the HP E2925A is participating as a master or target respectively.

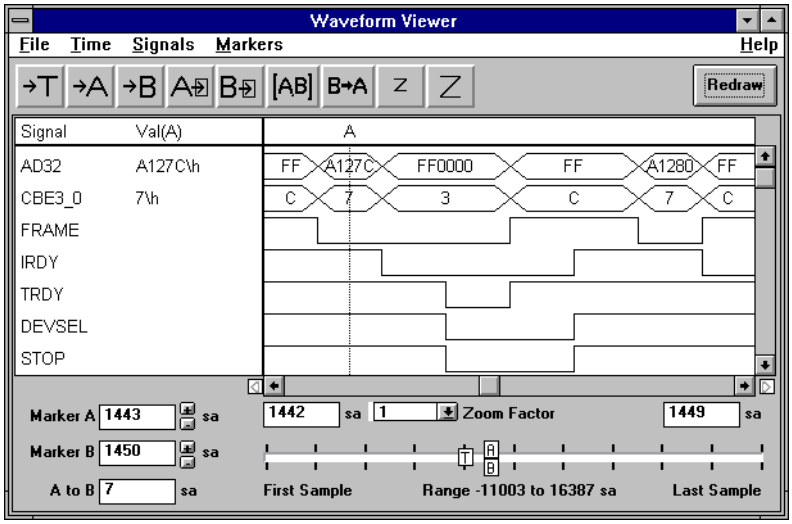
For this example, set the storage qualifer to “**All**”.



Press “**OK**” in the “Capture” window, and the built-in logic analyzer is ready to run. Press the “**Run**” button in the main window or select the “**Analyzer->Run**” command in the main menu. The status bar in the main window changes to “Running...” to indicate that the Analyzer is running and waiting for a trigger. In the demo mode that you have selected, the status line will indicate “Stopped” after a while[1].

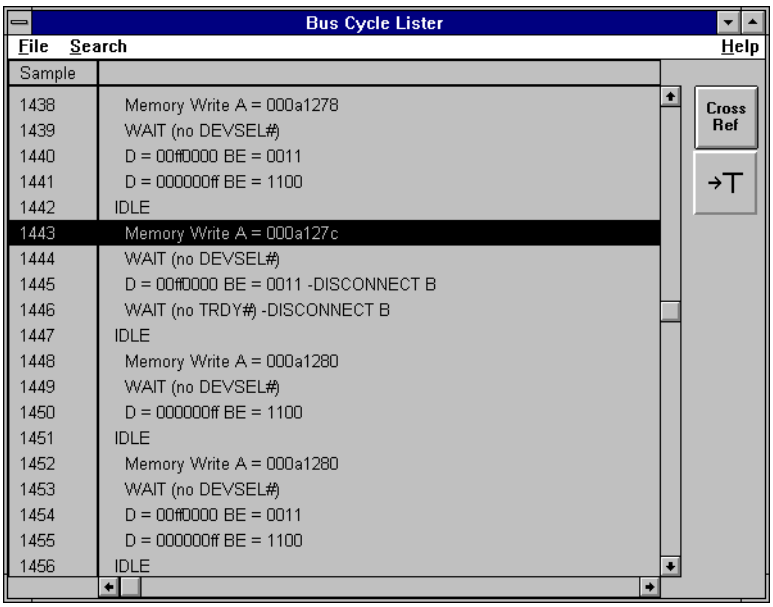
Click on the “**Wave**” button in the main window (or use the “**Analyzer->Waveform Lister...**” command in the main menu) to open the waveform viewer. Use the “**File->Load from file...**” command in the waveform window and load the trace file “**vga1.wfm**”[2]. To see the same section of the waveform as shown in the following picture, type “**1443**” into the **Marker A** text box at the lower left corner of the window, press **RETURN** and click on the “**->A**” button in the menu bar.

1. If you had the HP E2925A connected, it would of course only stop after the trigger event had actually occured on the bus.
2. All the logic analyzer trace files (*.wfm) can be found under **<your_installation_directory>\samples\demo**. If you did not change the default setting during installation, <your_installation_directory> will be **c:\hpbest.30**.



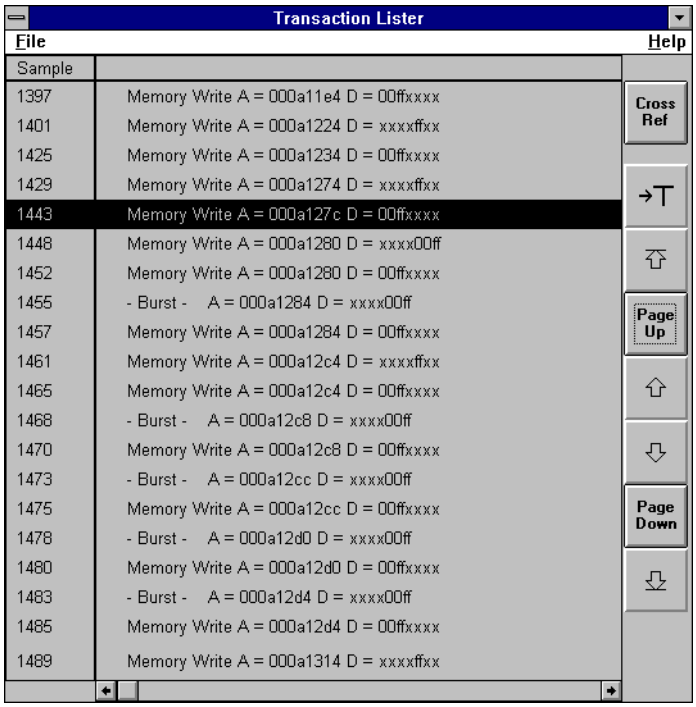
If you need more information about the buttons and other controls in the waveform viewer, use the "**Help->Help on Window...**" command in the waveform viewer.

Although the waveform viewer is appropriate for analyzing single transactions or when you need to check the state of individual control signals, it is tedious to "read" PCI transactions from looking at the waveform viewer. This is where the bus cycle lister can help. Click on the "**Cycle**" button in the main window (or use the "**Analyzer->Bus Cycle Lister...**" command in the main menu) to open the bus cycle lister. In the bus cycle lister window, press the "**T**" button to go to the trigger point. Scroll around to see how it works.



If you want to find out what the waveform is for a given line in the bus cycle lister, select the line in the bus cycle lister and press the "**CrossRef**" button (make sure that the waveform viewer is still open).

To get a more compressed overview of the transactions that occured on the bus, click on the “Xact” button in the main window (or use the “Analyzer->Transaction Lister...” command in the main menu).



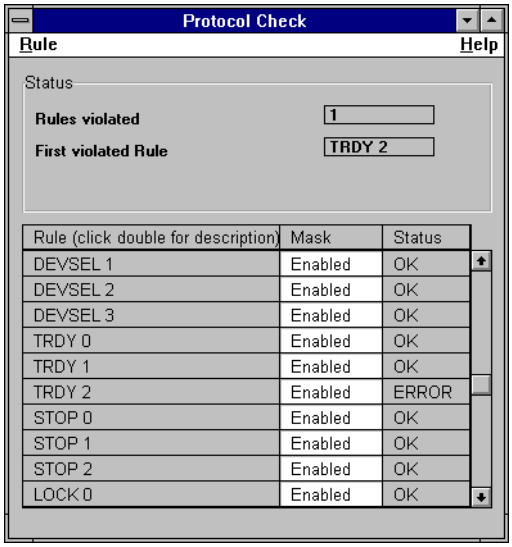
2.1.2 Detecting and analyzing a protocol violation

In this example, the protocol observer[1] of the HP E2925A is used in conjunction with the built-in logic analyzer to trigger on a protocol violation that occured during a stress test of a VGA graphics adapter. You can load the setup for this example from **vga2.bst**, and the trace data from **vga2.wfm** (load vga2.bst with File->Load in the main menu and vga2.wfm with File->Load in the waveform viewer).

From the GUI, the status of the protocol observer can be checked by opening the protocol check window with the “**Analyzer->Protocol Check...**” command in the main menu. This window indicates the first rule that was violated, as well as the total number. Each rule can be individually enabled or disabled to prevent flagging of known errors. In demo mode,

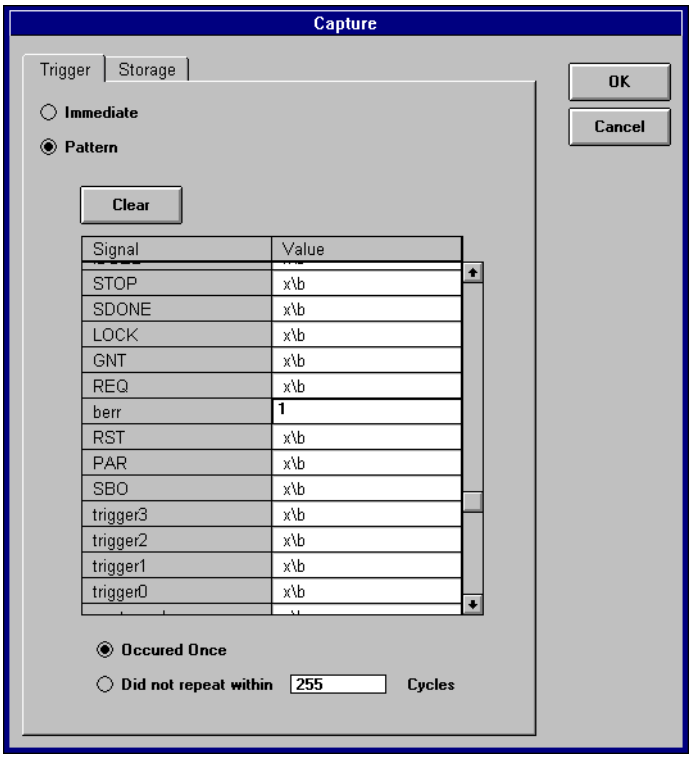
1. The protocol observer of the HP E2925A monitors 25 protocol rules from Appendix C of the PCI Spec. 2.1 in real-time and flags the violation of an error with an LED on the rear panel of the board, as well as an error number indicated on a HEX display on the board.

the software displays random errors to give you an idea of how it works. You can click on the rule name field to get an explanation of what the rule is checking.

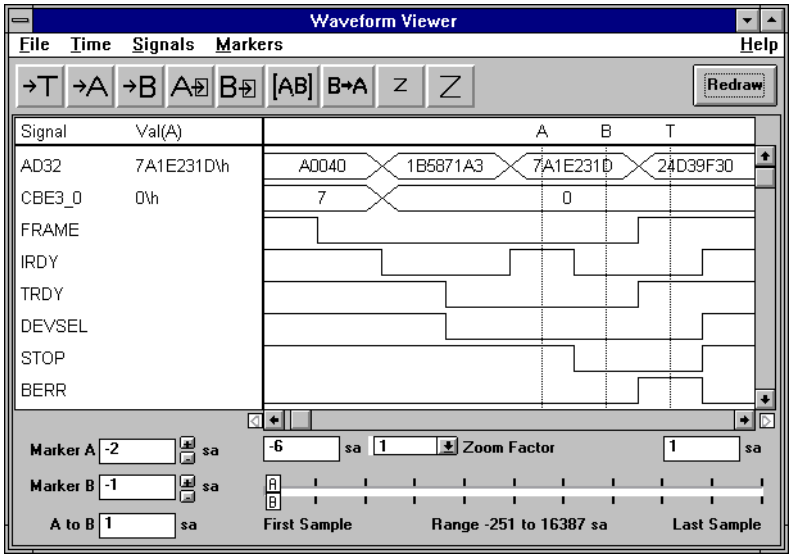


In order to analyze a protocol violation and find out which PCI device caused the violation, you can use the protocol observer's output "berr" (= bus error) to trigger the build-in logic analyzer. The "berr" signal is asserted when any one of the protocol rules has been violated.

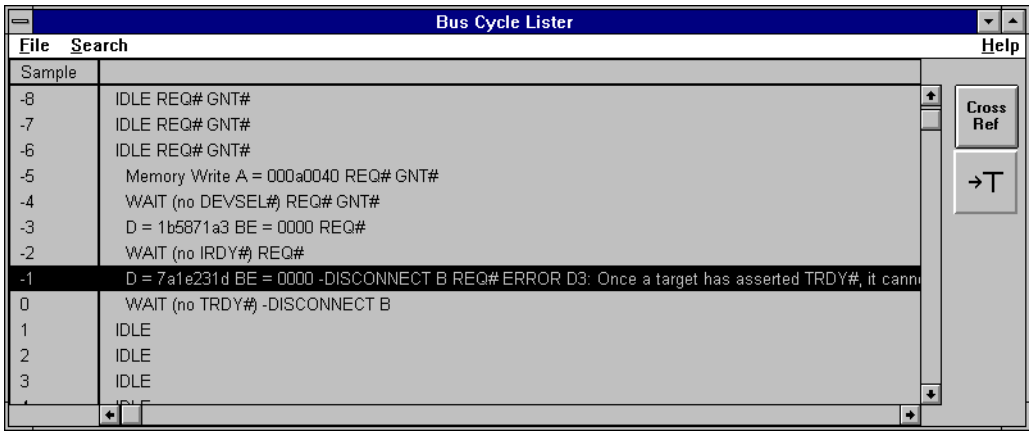
Press the "**Clear**" button in the "**Capture**" window to reset all pattern terms to "x" (= don't care), then set the "**berr**" field to **1**.



When the analyzer is started while the above mentioned stress test is performed, it triggers and shows the following result. Note that the “berr” signal in the waveform viewer is asserted on the trigger point. The error in this particular case is that the target commits itself to accept a transfer in the clock cycle marked by the A marker. The data phase does not complete in this clock cycle, because the master inserts a wait state (by deasserting IRDY#). In the next clock cycle (marked by marker B), the target “changes its mind” and asserts STOP#. This is a violation of rule 12d in Appendix C of the PCI Spec. 2.1.



The bus cycle lister shows the associated transaction, along with the protocol error message.



2.2 Viewing and modifying system memory, I/O and config

If you are:

- isolating a failure in a system (especially when the CPU is hung),
- trying to find the root cause of a failure that occurred during your chip or system level validation,

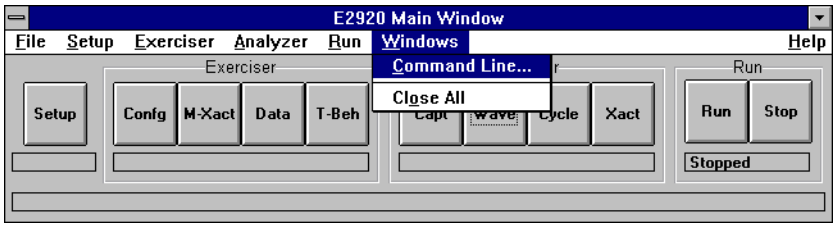
- writing and debugging low level software (e.g. BIOS code, device drivers),

it is often the case, that you need to peek and poke into physical memory locations and I/O or configuration registers in your system under test. The HP E2925A provides a number of **host access functions** that allow you to read and write any memory location, I/O or configuration[1] register as either byte, word or Dword transfers.

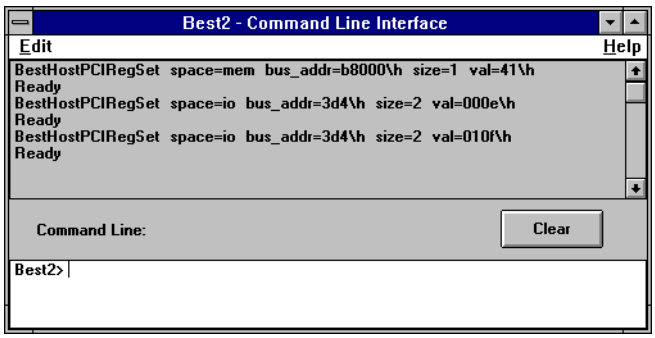
2.2.1 Example: Poking into VGA frame buffer memory

In this example, the HP E2925A is used to poke some data values into the video frame buffer memory of a VGA graphics adapter.

If you are working interactively, the HP E2925A's **command line interface** is the easiest way to use these host access functions. To access the command line interface, use the "**Windows->Command Line Interface**" command in the main menu.



In this example, the HP E2925A is used to write a character into the top left corner of the VGA text screen (by writing a single byte to physical memory address 0xb8000) and set the cursor next to this character (by doing two I/O word writes to I/O address 0x3d4). In the screen shot below, the three commands were typed in on the **Best2>** prompt. Alternatively, CLI commands can be stored in an ASCII text file and executed in batch mode. The commands shown here will "only" do the desired data transfer. You have no control over how the transfers are performed (e.g. how many wait states will be inserted, etc.). Functions to control this behavior in more detail are shown in later examples.



When the HP E2920 software is running in demo mode, typing commands in the command line interface window has, of course, no effect.

1. The HP E2925A can only read/write configuration space if the IDSEL signals are connected to AD lines.

2.3 Deterministic generation of PCI transactions

If you:

- are integrating a PCI chip into an adapter or system,
- have a report from your validation team (or even from your customer) that your PCI chip/system is not working under certain circumstances,

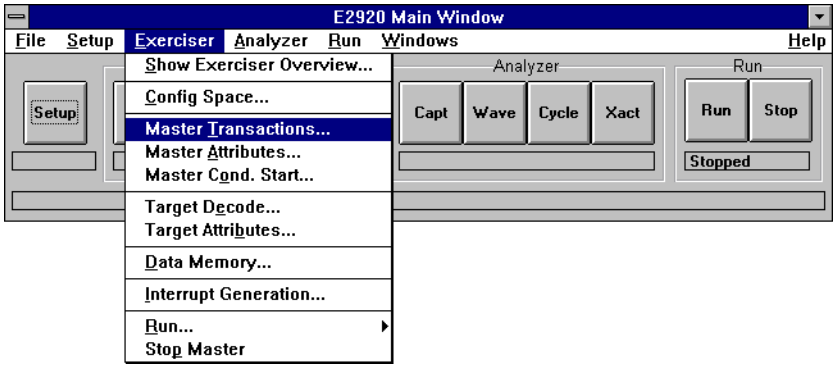
you probably need to generate/reproduce a given PCI scenario and find out how your chip behaves in the real environment on certain PCI commands, certain protocol variations, such as master wait states, or error conditions such as wrong parity.

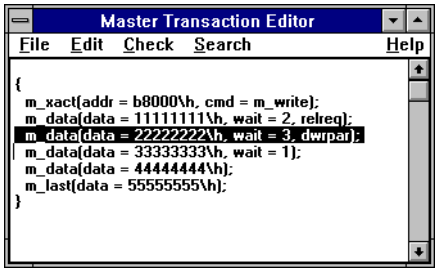
The HP E2925A, together with the HP E2971A Exerciser GUI, is an ideal tool in this situation. It allows you to set up and run any PCI transaction as a master or target with deterministic control over the protocol behavior. For a complete reference of the available protocol parameters, refer to the user's guide.

2.3.1 Example: Generating master transactions using the GUI

The following example shows how to set up the HP E2925A as a master to generate burst transfers into the VGA frame buffer memory whilst varying certain protocol parameters (the example is stored under **vga4.bst**, the trace file under **vga4.wfm**).

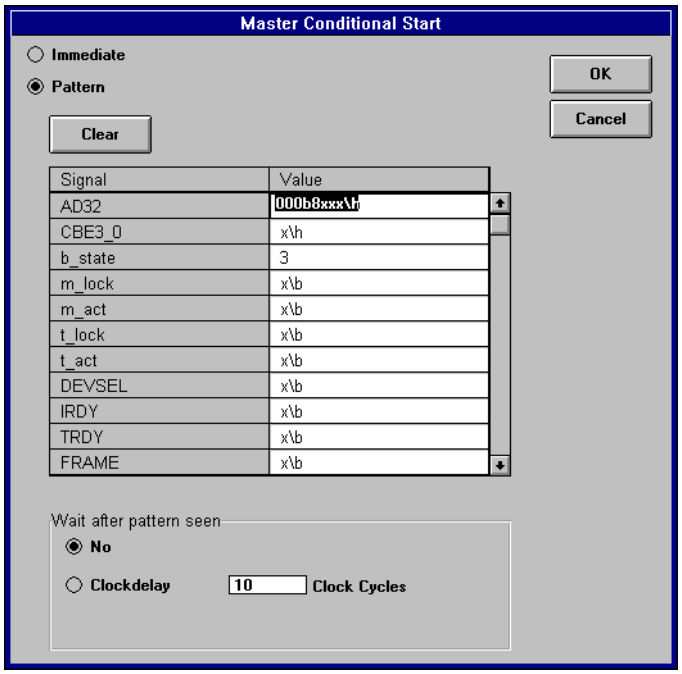
Click on the "**M-Xact**" button in the main window (or use "**Exerciser->Master Transactions...**" menu command in the main window) to open the Master Transactions window.





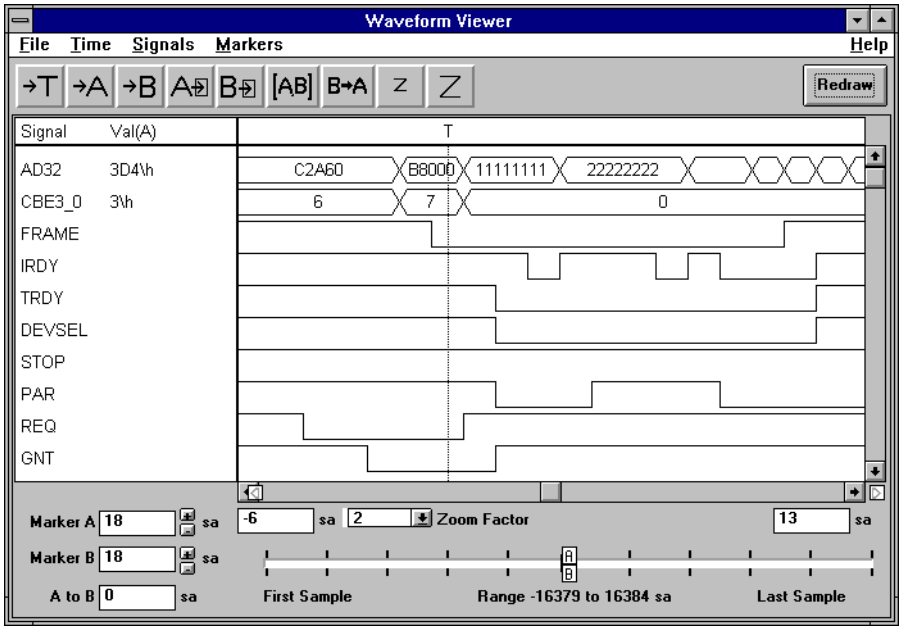
This transaction block generates a memory write transaction to address b8000\h and to perform a burst of 5 Dwords (if the target, i.e. in this case the graphics card, does not disconnect earlier). The m_data statements specify the data to be transferred, along with protocol attributes[1] such as wait states, wrong parity, release REQ#.

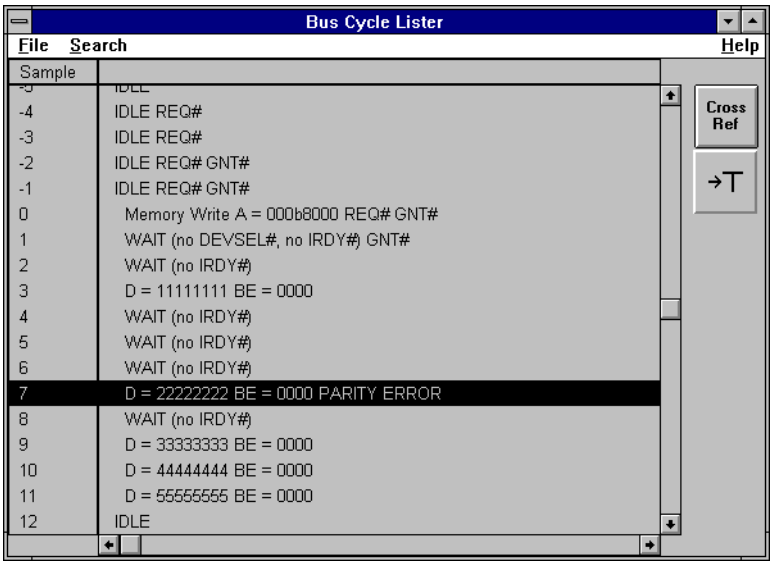
When the "**Run**" button is pressed in the main window, the software downloads this transaction block into the HP E2925A's internal memory and starts execution either immediately or after a "Master Conditional Start" pattern has occured on the bus. For our example, use the "**Exerciser->Master Cond. Start...**" menu command in the main window to open the Master Conditional Start window. The pattern in this window is set to an address phase (**b_state** = **3**) to the address range b8000...b8fff (**AD32** = **000b8xxx\h**).



Here is the waveform and bus cycle listing of the transaction block being executed. Note the transaction that triggers the HP E2925A's transaction block.

1. Please refer to the **Bus Transaction Language Reference** section of the User's Guide for a complete list of protocol attributes.

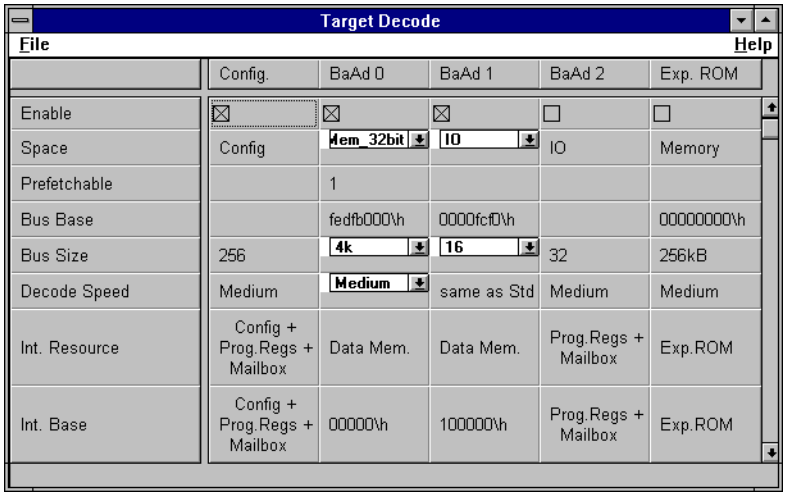




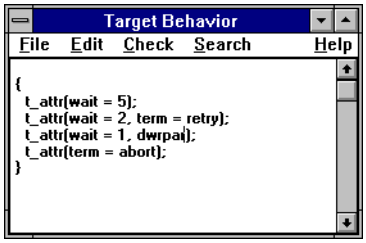
2.3.2 Example: Setting up target transactions using the GUI

If you are developing, debugging, validating or characterizing a PCI bus mastering device, you need a programmable target that can react deterministically in the required fashion. The HP E2925A provides this functionality. It has 128kB on-board read/writeable memory that can be accessed with either memory or I/O transactions. The target protocol behavior (i.e. number of wait states, termination type, error insertion, etc.) is fully programmable on a per data phase basis. The setup for this example is available in **target1.bst**, the trace file in **target1.wfm**.

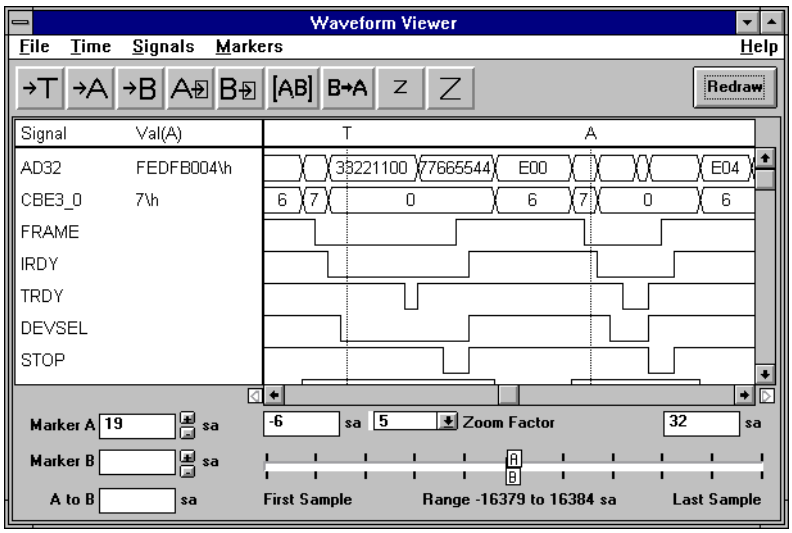
To view the target decoders, use the "**Exerciser->Target Decode...**" menu command in the main window.



The target protocol behavior is defined in a similar way to that of the master. Click on the "**T-Attr**" button in the main window (or use the "**Exerciser->Target Attributes...**" menu command in the main window) to view the Target Attributes window. Here is an example of a target setup.



Here is what happens when the card is accessed by a master.



2.4 Chip and system validation using a C test program

If you are:

- validating a PCI chip set or application chip (graphics, hard disk, network controller, etc.),
- integrating a PCI system or PCI adapter,

you probably have your own testprogram(s) running on the target platform to validate your device-under-test automatically after new releases or bug fixes. Of course, the goal of your chip/system validation is to have the test coverage as close as possible to 100%, which is never possible in a reasonably complex system.

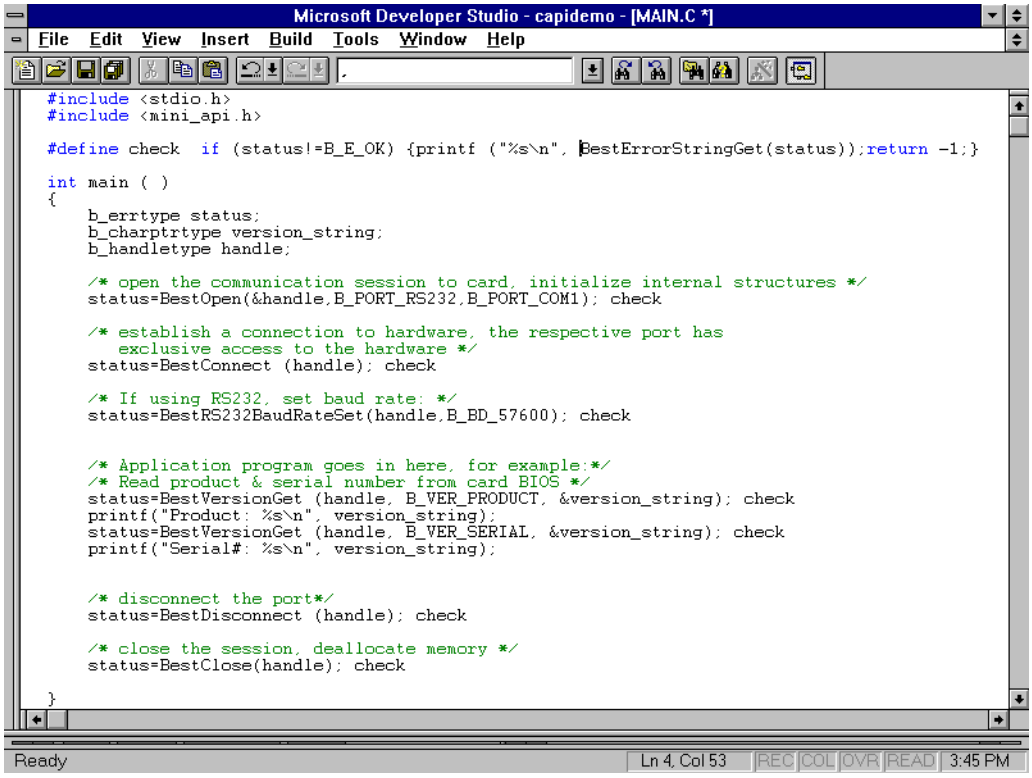
The HP E2925A can help you to increase the test coverage of your chip/system validation by providing ready-to-use test functions that can be controlled from within your test program. The HP E2925A comes with a C-application programming interface (“C-API”), a library of C functions that allows you to control the complete functionality of the card, with the controlling software running either on the target system itself or on an external WinNT, Win95 or MS-DOS host. The C-API is even available in source code, so you can port it to your specific hardware if it is not running one of the Microsoft operating systems.

2.4.1 Example: Using the HP E2925A C-API

The first piece of C-Code shows how the HP E2925A is controlled programmatically from a C program. The main routine opens a connection to the HP E2925A using the function **BestOpen()**. Instead of the RS232 port, the communication with the card could also be through the bi-directional parallel interface, or through PCI if the program is running on the target system itself. **BestConnect()** establishes a logical connection to the card. Now, all other C-API functions can be used. In this particular example, only the product and serial number are retrieved from the board. At the end of a program, **BestDisconnect()** and **BestClose()** must be called to close the logical connection with the card and free all resources. This sample C program is available as **template.c**.

You may notice that almost all functions of the HP E2925A C-API adhere to the same “style”: They start with “Best...” to distinguish them from other functions in your code, and they all return an error code. For readability purposes, the error checking is done in the macro “check”, which prints an error message and exits in this example.

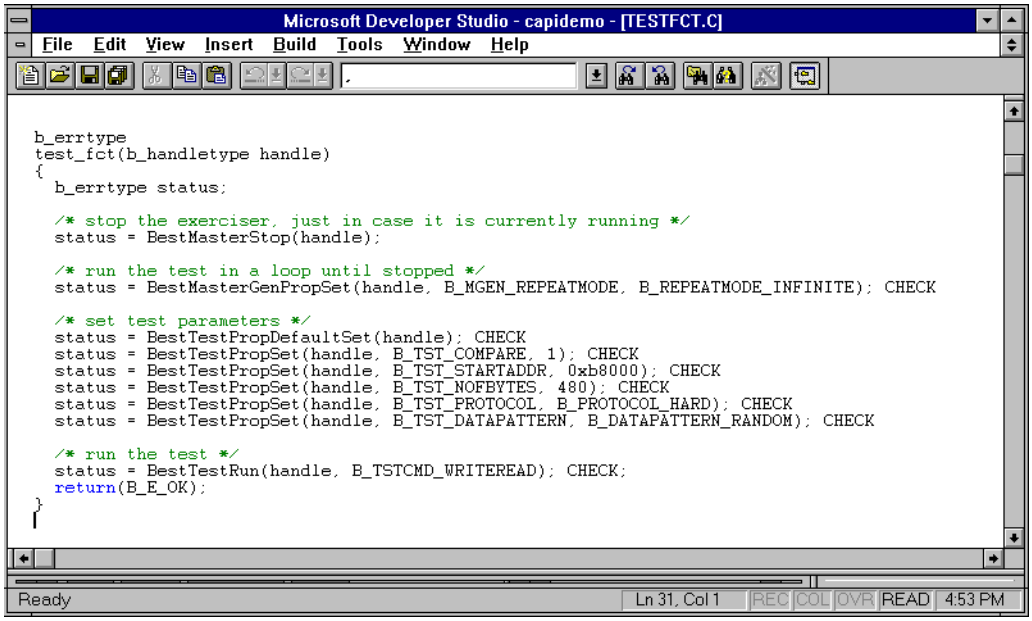
You can take a look at this example in template.c. If you have the HP E2925A hardware and a **Microsoft C++ 4.0 compiler**, you can compile and run the example yourself.



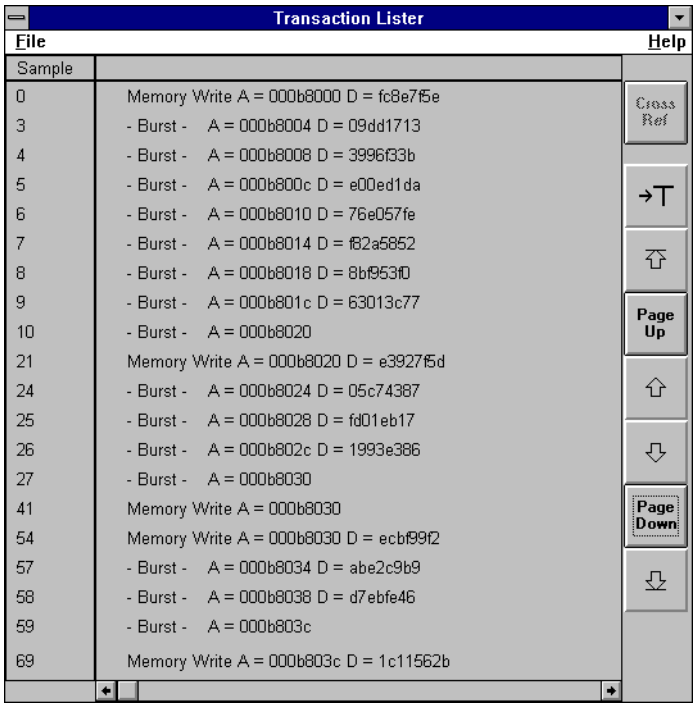
2.4.2 Example: Setting up a memory test using the C-API

This example shows how one of the built-in test functions of the HP E2925A can be used from within a C program. The sample C program is available as **test_fct.c**. The logic analyzer trace file is stored in **capi1.wfm**.

Before calling the function shown in the listing, a connection (BestOpen, BestConnect) must already be established.



When you load the logic analyzer trace file (**capi1.wfm**) into the waveform viewer or bus cycle lister, you can see which transactions the test function generates, and how the target (the graphics card in this case) reacts. Here, the transaction lister is shown.



.