HEWLETT PACKARD

HP 75000 Model D20 Programming Note

Using Simulator Vectors and Fault Dictionaries

Overview

Test vectors and fault dictionaries generated by logic simulators can be used with the HP 75000 Model D20 to provide a powerful test system, and a fast, convenient way to enter test vectors into the test

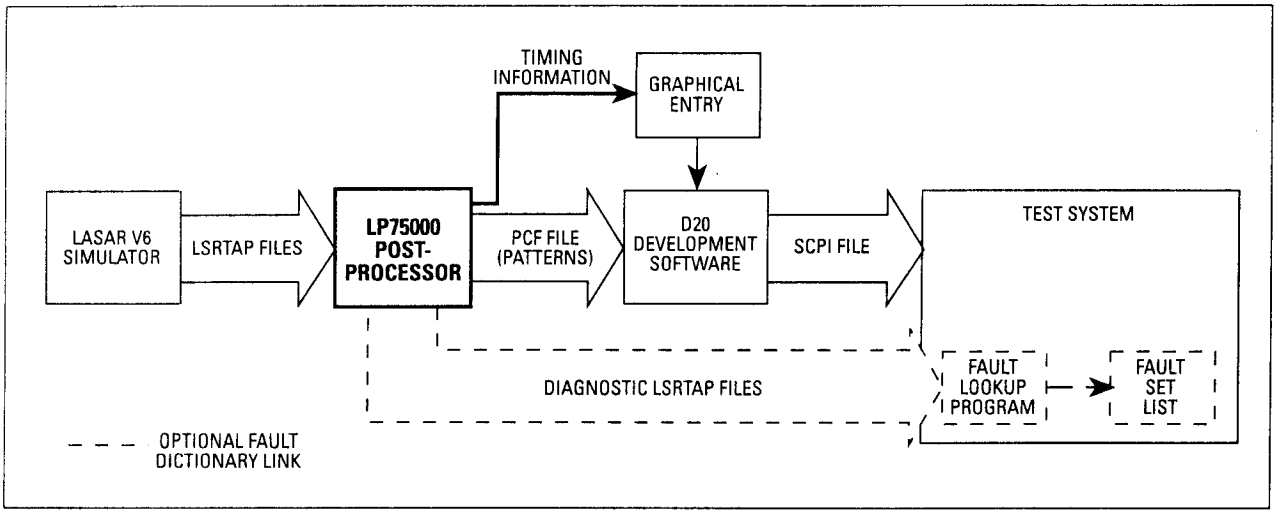

Figure 1. LASAR simulator link overview

system. This programming note describes this process using the LASAR Version 6 simulator; the process is similar for other simulators. An example circuit is simulated on the LASAR system generating LSRTAP files which are processed by the LP75000 post processor from Lewis Systems[1]. The post processor insures compatibility with the Model D20 and creates an output file in the standard HP VCL/PCF format. Vector Control Language (VCL) and Pattern Capture Format (PCF) were developed by HP's Manufacturing Test Division (Loveland, Colorado) to provide a standard for importing test vectors into a tester. After the VCL/PCF file is imported into the Model D20 development software, the test is debugged and a SCPI (Standard Commands for Programmable Instruments) file, used in the run time environment, is generated. A test executive, constructed using HP VEE, then loads the SCPI file, runs the test, and invokes a fault dictionary lookup program to perform the fault diagnosis. Figure 1 shows an overview of the entire process. Each of these above steps will now be examined in more detail.

[1] For more information, contact: Lewis Systems, Inc. 1915 Peters Road, Suite 113, Irving Texas, 75061. Phone: (214)438-2177

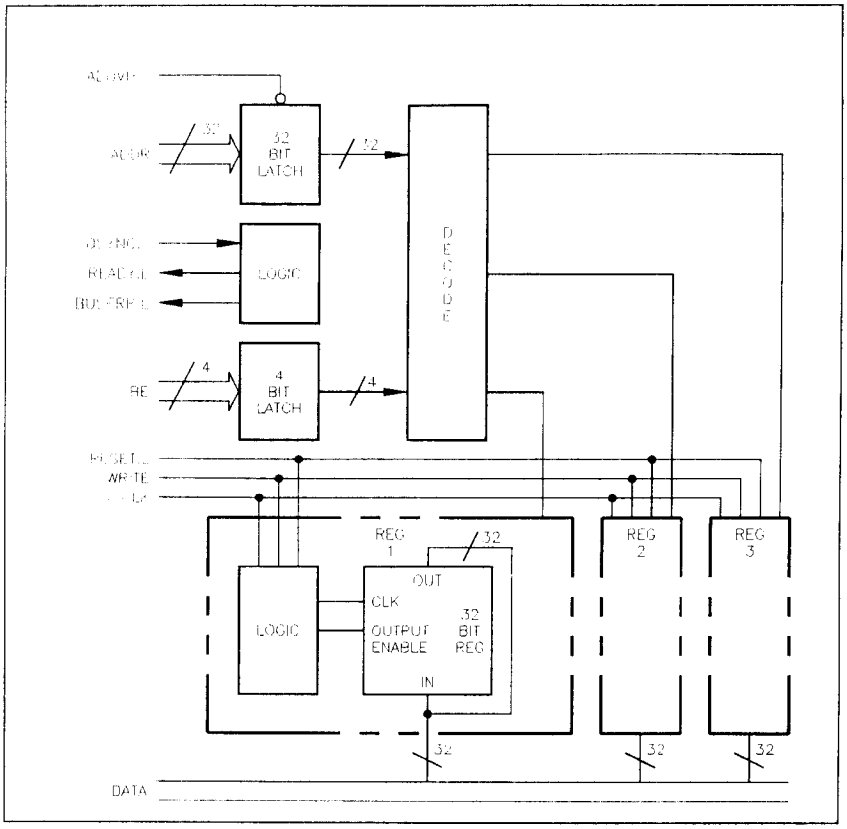

Figure 2. Example circuit to be simulated.

LASAR Simulation

The simulation process runs in the LASAR environment and will not be discussed in detail here other than to illustrate some of the key points as they pertain to the example circuit. The example circuit consists of three data registers connected to a 32-bit data bus, a 32-bit address bus, address decoding, and timing circuitry. Figure 2 is a diagram of the circuit which was entered into the LASAR simulation system.

Once the circuit was entered, the source program to generate the simulation vectors on the LASAR system was written, as shown in Table 1 at the end of this note (page 8). From this listing it can be seen that LASAR is compatible with cycle-based testers since it allows tsets (timing sets) consisting of one or more phases to be defined. Phases are timing specifications that can be assigned to a group of nodes. A node is any signal such as an address line. All address lines can be grouped together and assigned to a phase. Similarly, all data lines can be grouped together and assigned to another phase. This allows the address and data lines to have different timing. In this example the address group is assigned to phase 1 and the data lines to phase 3; therefore, within tset 1, these lines can change at different times. LASAR also allows subroutines to be defined which can emulate different bus cycles. **Write_data** and **read_short** are two subroutines shown in Table 1 and are used to generate vectors easily when called in the LASAR source program.

Using the LASAR Rules (.RUL) and Environment (.ENV) files provided with the LP75000 Postprocessor, the example circuit was simulated. These files define the hardware and software boundaries for the target Model D20 system and help ensure that the resulting vectors are compatible with the tester. After a successful simulation run, the .TAP files needed by the postprocessor were created by executing the **POSTPROCESS LSRTAP** command on the LASAR system. The following files were created:

HEADER.TAP	**TIMESETS.TAP**
STIMULUS.TAP	**TIMPERPAT.TAP**
RESPONSE.TAP	**PHASECONN.TAP**
PINAMES.TAP	**PIFORMATS.TAP**
PONAMES.TAP	**FORMATTRS.TAP**

Since fault dictionaries are being used in this example, a **LASAR JUDGE** run on the simulator must be completed. This will create the additional files used by the LP75000 lookup program to generate the fault diagnosis. The additional output files from the **LASAR JUDGE** run are as follows:

FDPOPATS.TAP	**FDFLTSIG.TAP**
FDPRINT.TAP	

At this point all of the LSRTAP files needed by the postprocessor and the fault dictionary lookup program were available. These files are the .TAP files listed above. The *LP75000 LASAR to HP 75000 Model D20 Postprocessor Users Manual* contains additional suggestions on making the simulation compatible with the Model D20.

Post Processor

In general, the post processing operation is performed for three main reasons:

1. To verify that tester rules are not violated by the simulation vectors.
2. To efficiently use the tester resources.
3. To translate the simulator output into VCL/PCF format so that it can be input into the tester.

The options file is useful to control the post processor and should be set up prior to invoking the LP75000. The options file allows you to change the names of groups that were created by LASAR to make them more meaningful to your test. In addition, the lines that are to be assigned to the timing module as control lines can be specified. In general, any single line that has its own timing should be assigned to a control line. Typical control type lines are clocks and handshaking lines. Finally, the option file can specify that a software compare will be performed. This allows individual nodes in a group to be set to the 'don't care' state, which is necessary when using the fault dictionary option.

When software compare is not specified, the post processor will set all nodes in a group to 'don't care' if one or more nodes in the group are set to 'don't care' by the simulator. This allows real-time compare in the tester.

The LP75000 post processor was run on the same system that the LASAR simulator was installed on, in this case a DEC VAX machine. Before running the post processor, the options file HP_TESTC.OPT was created. The options file was used to specify that the following lines were to be assigned to control lines on the Model D20 — **BUSSER_L, ADDVH_L, RESET_L, GCLK, WRITE**, and **DSYNCH_L**. This was done to use the tester resources efficiently. If this were not specified in the options file, the postprocessor may assign them to pattern I/O lines and, depending upon the timing, could use up one 8-bit port for each line. The other item specified in the options file was to change the name of phase1 to addr since that was more meaningful to the DUT. Finally, software compare was specified since the fault dictionary option was used. Table 2 shows a listing of the options file.

```
phase 1 = addr
control = BUSSER_L
control = ADDVH_L
control = RESET_L
control = GCLK
control = WRITE
control = DSYNCH_L

software compare
```

Table 2. LP75000 Options File

The post processor was run with the following command:

```
lp75000
```

The output file is called HPTIME and is in standard VCL/PCF format, as shown in Table 3 (starting on page 9). There are three sections of the output file — definition section, timing section, and vector execution. The first two sections are used to set up the Model D20 development software prior to importing the file. This will be discussed in the next section.

Model D20 Development System

Up to this point, all processing was done on the LASAR system using a DEC VAX machine. The VCL/PCF and .TAP files will need to be transferred to other systems via a LAN or other means prior to running the tests. In this example the VCL/PCF file was stored on a flexible disc and then entered into the UNIX workstation running the Model D20 Development Software. Prior to importing vectors into the Model D20 Development System, the software must be configured by creating the pingroups and timing cycles specified in the VCL/PCF file.

In this example the Model D20 Development Software was configured for four pattern I/O modules and one timing module using the Hardware Configuration menu. The number of modules required is determined from the Definition Section of the VCL/PCF file. In this case, there are 32 address lines, 32 data in lines, 32 data out lines and four BE lines. This requires four pattern I/O modules. The remaining lines are control lines and will be assigned to the timing module.

The next step was to define the following pin groups: **addr, phase2, phase3, FMT4, FMT5, FMT6, FMT7, FMT8, FMT9** and **window10**. These names must be exactly as specified in the VCL/PCF file. The types of pingroups (stimulus, response and control) are also specified in the VCL/PCF file definition section. Groups designated as "inputs formatted" in the VCL/PCF file are configured as control in the Model D20 development software. Similarly, "outputs" are configured as response groups, and "inputs" are configured as stimulus groups. In order to determine which timing generator to assign to each group, the timing sets must be examined.

It can be seen from looking at **TSET1**, in Table 3, that the **addr, phase2** and **phase 3** groups are all driven together on the same line. This will be consistent in all tsets. These groups all can be assigned the same timing generator to conserve tester resources. Each drive and receive line in the tset should be assigned to a different timing generator. In this case the only other line is the **receive window10** line which was assigned to response generator 0. The final consideration is whether to set the response groups to record or compare. This is determined by the option used in the options file. In this case software compare was specified in the options file, indicating that the response groups should be set to record so that the data can be read into the test executive for software comparison.

Figure 3 shows the pin group definitions which were input into the development software.

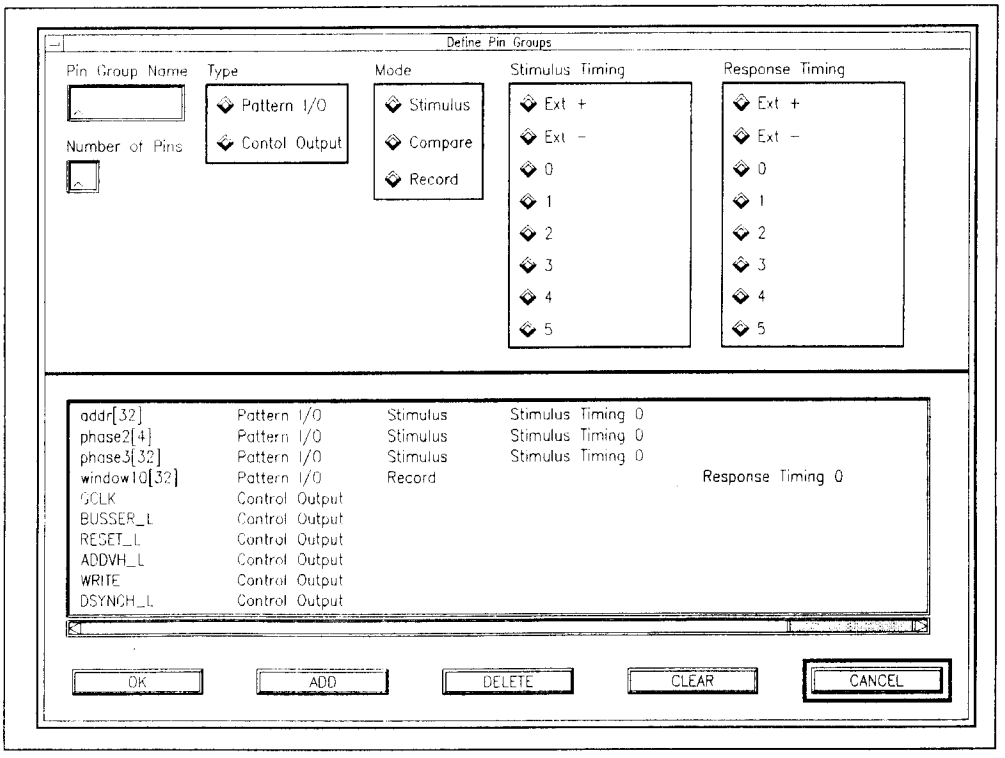

Figure 3. Defining Pin Groups

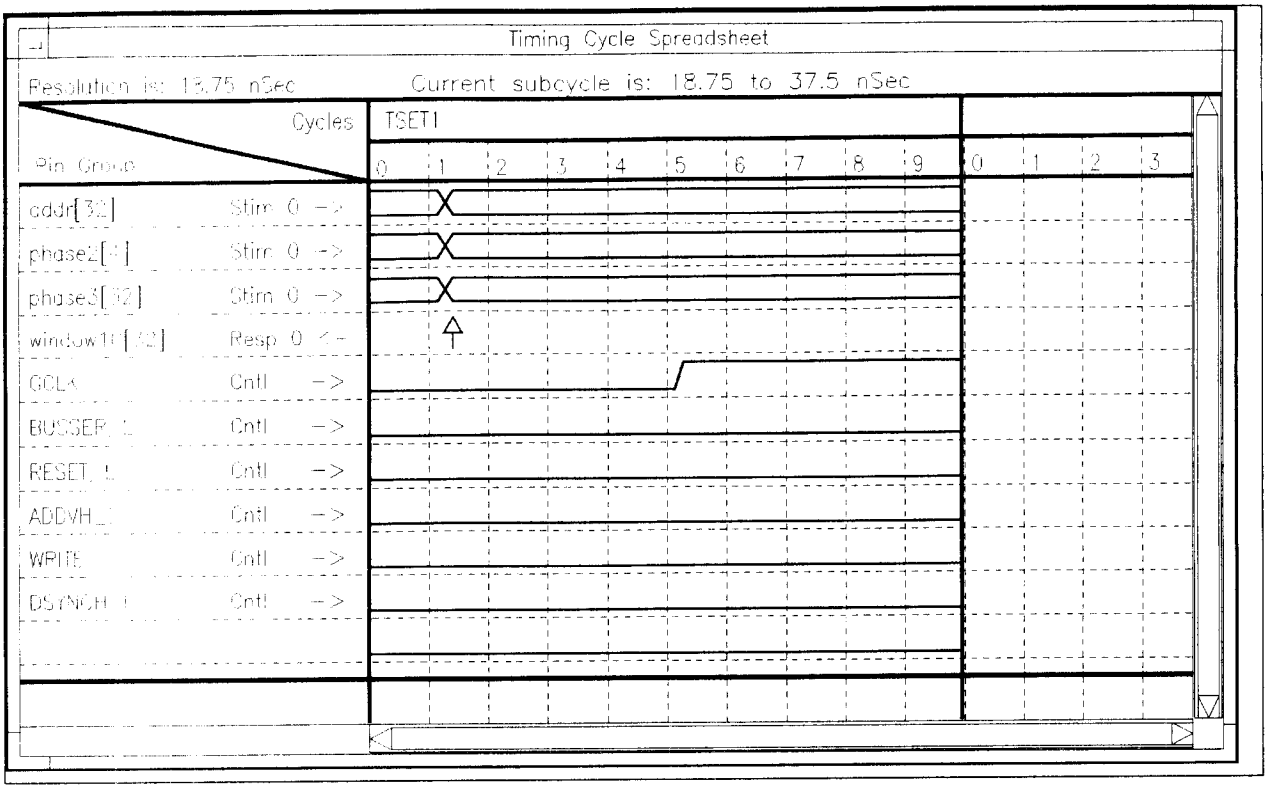

Figure 4. Tset1 timing cycle

At this point the timing cycles can be graphically entered by the user. Figure 4 indicates how **TSET1** was entered.

The names of the timing cycles defined in the development software are the same as in the VCL/PCF file (Table 3). The number of subcycles for the timing set is the same as the number of events specified for that tset in the TIMING SECTION of the VCL/PCF file. For each group, the event indicated in the drive statement in the VCL/PCF file corresponds to a change data entry in the timing spread sheet. Similarly, receive indicates a sample data in the timing spread sheet. The control lines use to 0 or to 1 as specified in the tset definition. After all of the timing sets were defined, the VCL/PCF file was imported from the File Import menu of the development software.

The breakpoint and vector single-stepping features of the development software were used to ensure that the test ran properly on the Model D20 with the example circuit hardware. Once the test was completely debugged, a SCPI file that can be used by the test executive was generated from the File SCPI menu. This file was named LASAR2.ASC in this example.

Test Executive

At this point all of the necessary files have been created to allow the test executive to utilize the diagnostic features of LASAR. The following files were transferred to the system running the test executive: LASAR2.ASC, FDFLTSIG.TAP, FDPOPATS.TAP, FDPRINT.TAP, HEADER.TAP, and PONAMES.TAP. In addition, the WEIGHTS.FAU file was created to determine how to weight the fault dictionary matches and mismatches to determine the most likely fault set. The LP75000 manual describes the weightings and how they are used.

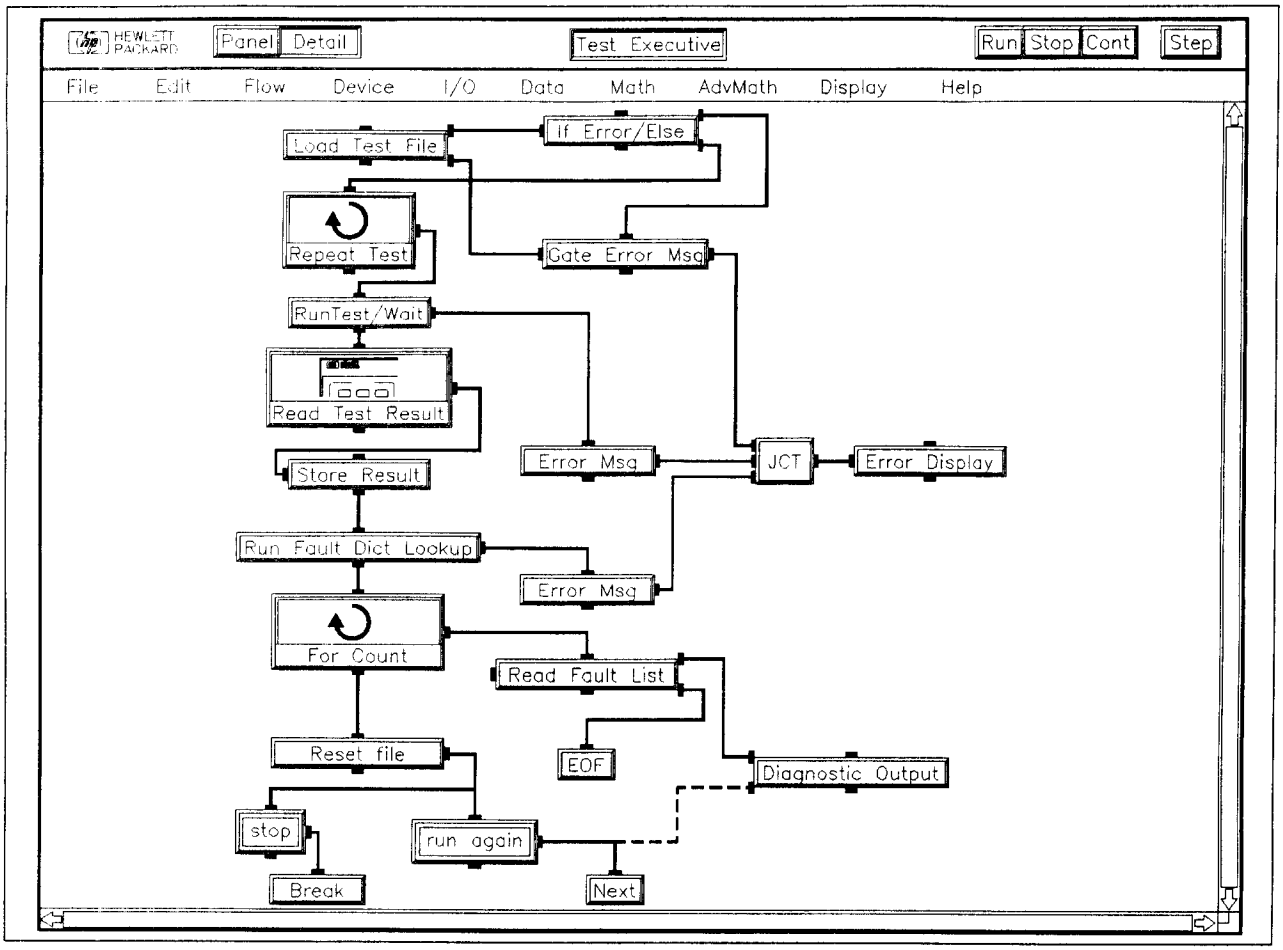

Figure 5. Test executive built with HP VEE

The test executive in this example was developed using the HP Visual Engineering Environment (HP VEE), a powerful tool for controlling and analyzing the test flow. Figure 5 is a diagram of the HP VEE model used.

The HP VEE model makes extensive use of the Model D20 state drivers to control the instrument during the test. For example in the "Load Test File" block, the Model D20 driver takes the SCPI file generated by the development software and loads it into the Model D20 hardware so that it can be run by the "Run Test/Wait" block. The "Read Test Result" and "Store Result" blocks read data out of the Model D20 and load it into a result file for use by the fault dictionary lookup program. The format of the result file is a series of IEEE 488.2 block definite binary blocks. This is one of the formats available to output data from the Model D20. The job of the test executive is to read out each response group in the order that it is specified in the VCL/PCF file **"pcf order is"** statement (Table 3) and store them in block definite format in the result file. The fault dictionary lookup program called fd, expects the result file to be named hpfd.

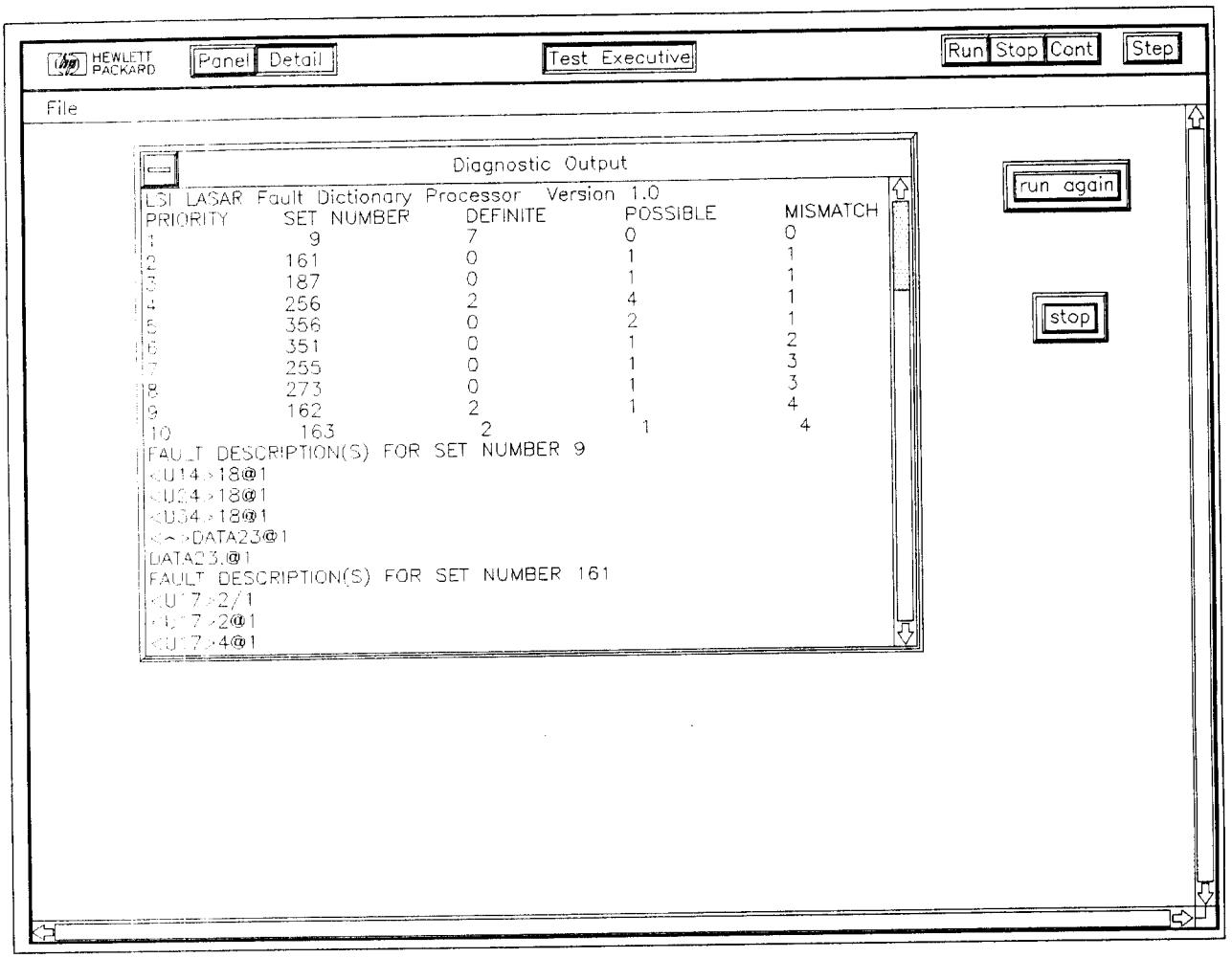

Figure 6. Diagnostic output

Table 4 (page 14) is a description of the result file created by the test executive. The fd program used the result file just created, along with the LSRTAP files to create the FAULTS.LIS file. Finally, the test executive can display the diagnostic output contained in the FAULTS.LIS file and ask the operator if the test should be run again.

Figure 6 contains an example of the diagnostic output resulting from a fault placed in the example circuit.

In this case the priority 1 fault set is fault set 9 which consists of three devices U14, U24, and U34, and one primary input line DATA23. These are the most likely causes for the failure. For more information on using the fault dictionary features the *LP75000 LASAR to HP 75000 Model D20 Postprocessor Users Manual* can be consulted.

```
SET TSET 1 CLOCK = 187500 PS
PHASE 1 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 2 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 3 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 4 ASSERT = 0 PS  RETURN = 93750 PS
PHASE 5 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 6 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 7 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 8 ASSERT = 18750 PS  RETURN = 168750 PS
PHASE 9 ASSERT = 18750 PS  RETURN = 168750 PS
WINDOW 10 OPEN = 0 NS  CLOSE = 27500 PS ;

SET PHASE 1 TRIGGER = $TOPAT ;
SET PHASE 2 TRIGGER = $TOPAT ;
SET PHASE 3 TRIGGER = $TOPAT ;
SET PHASE 4 TRIGGER = $TOPAT ;
SET PHASE 5 TRIGGER = $TOPAT ;
SET PHASE 6 TRIGGER = $TOPAT ;
SET PHASE 7 TRIGGER = $TOPAT ;
SET PHASE 8 TRIGGER = $TOPAT ;
SET PHASE 9 TRIGGER = $TOPAT ;
  !
  !
SET DIGITAL
(ADDR31,ADDR30,ADDR29,ADDR28,ADDR27,ADDR26,ADDR25,
ADDR24,ADDR23,ADDR22,ADDR21,ADDR20,ADDRR19,ADDR18,
ADDR17,ADDR16,
ADDR15,ADDR14,ADDR13,ADDR12,ADDR11,ADDR10,ADDR9,
ADDR8,ADDR7,ADDR6,
ADDR5,ADDR4,ADDR3,ADDR2,ADDR1,ADDR0)
PHASE=1
FORMAT=$NRET ;

SET DIGITAL (
BE0   ,BE1   ,BE2   ,BE3)
PHASE=2
FORMAT=$NRET ;

SET DIGITAL
(DATA31,DATA30,DATE29,DATA28,DATA27,DATA26,DATA25,
DATA24,DATA23,DATA22,DATA21,DATA20,DATA19,DATA18,
DATA17,DATA16,
DATA15,DATTA14,DATA13,DATA12,DATA11,DATA10,DATA9,
DATA8,DATA7,DATA6,
DATA5,DATA4,DATA3,DATA2,DATA1,DATA0)
PHASE=3
FORMAT=$NRET
WINDOW = 10;

SET DIGITIAL (GDLK)
PHASE = 4
FORMAT = $RONE ;

SET DIGITAL (BUSSER_L)
PHASE=5
FORMAT=$NRET ;

SET DIGITAL (RESET_L)
PHASE=6
FORMAT=$NRET ;

SETE DIGITAL (ADDVH_L)
PHASE=7
FORMAT=$NRET ;

SET DIGITAL (WRITE)
PHASE=8
FORMAT=$NRET ;

SET DIGITAL (DSYNCH_L)
PHASE=9
FORMAT=$NRET ;

DECLARATIONS

GROUPNAME addr FOR
addr31, addr30, addr29, addr28, addr27  addr26,
addr25, addr24, addr23, addr22, addr21, addr20, addr19, addr18,
addr17, addr16, addr15, addr14, addr13, addr12, addr11, addr10,
addr9,  addr8,  addr7,  addr6,  addr5,  addr4,  addr3,  addr2,
addr1, addr0

GROUPNAME data FOR
data31, data30, data29, data28, data27, data26, data25, data24
data23, data22, data21, data20, data19, data18, data17, data16,
data15, data14, data13, data12, data11, data10, data9,  data8,
data7,  data6,  data5,  data4,  data3,  data2,  data1,  data0

GROUPNAME BE FOR be3, be2, be1, be0

ENDDECLS

SUBROUNTINE WRITE_DATA (ADDRESS, DATIN, BEIN)
INTEGER ADDRESS
INTEGER DATAIN
INTEGER BEIN
BEGIN
    LO ADDVH_L HI WRITE
    DRIVE ADDR TO ADDRESS
    drive date to datain
    DRIVE BE TO BEIN ;
    HI ADDVH_L ;
    LO WRITE ;
RETURN
END

SUBROUTINE READ-SHORT (ADDRESS, BEIN)
INTEGER ADDRESS
INTEGER BEIN
BEGIN
    off data
    lo addvh_l write
    drive addr to address
    DRIVE BE TO BEIN ;
    HI ADDVH_L;
    ;;;;
RETURN
END

START:
!
! FIRST DOING INIT
!
HIGHSPEED
USE TEST 1
CPP=1
LO GCLK
HI ADDVH_L BUSSER_L RESET_L DSYNCH_L
LO WRITE
OFF ADDR BE DATA ;
LO DSYNCH_L ;
CALL WRITE_DATA    (%HF4100800,%H00000000,%HC)
CALL WRITE_DATA    (%HF4100800,%H00000000,%H3)
CALL WRITE_DATA    (%HF4100804,%H05000000,%HC)
CALL WRITE_DATA    (%HF4100804,%H00000080,%H3)
CALL WRITE_DATA    (%HF4100820,%H00000000,%HC)
CALL WRITE_DATA    (%HF4100820,%H00000008,%H3)

CALL READ_SHORT    (%HF4100800,%HC)
CALL READ_SHORT    (%HF4100800,%H3)
CALL READ_SHORT    (%HF4100804,%HC)
CALL READ_SHORT    (%HF4100804,%H3)
CALL READ_SHORT    (%HF4100820,%HC)
CALL READ_SHORT    (%HF4100820,%H3)

END
```

Table 1. LASAR Source Listing

```
!LSI LASAR - HP75000 Postprocessor Version 1.0     6-FEB-1992 09:27
! Timing file for "hp_testcase"

!— DEFINITION SECTION —

!*assign addr   to nodes "ADDR31", "ADDR30", "ADDR29", "ADDR28", "ADDR27"
!*assign addr   to nodes "ADDR26", "ADDR25", "ADDR24", "ADDR23", "ADDR22"
!*assign addr   to nodes "ADDR21", "ADDR20", "ADDR19", "ADDR18", "ADDR17"
!*assign addr   to nodes "ADDR16", "ADDR15", "ADDR14", "ADDR13", "ADDR12"
!*assign addr   to nodes "ADDR11", "ADDR10", "ADDR9", "ADDR8", "ADDR7", "ADDR6"
!*assign addr   to nodes "ADDR5", "ADDR4", "ADDR3", "ADDR2", "ADDR1", "ADDR0"

!*assign phase2  to nodes "BE3", "BE2", "BE1", "BE0"

!*assign phase3    to nodes "DATA31_in", "DATA30_in", "DATA29_in", "DATA28_in"
!*assign phase3    to nodes "DATA27_in", "DATA26_in", "DATA25_in", "DATA24_in"
!*assign phase3    to nodes "DATA23_in", "DATA22_in", "DATA21_in", "DATA20_in"
!*assign phase3    to nodes "DATA19_in", "DATA18_in", "DATA17_in", "DATA16_in"
!*assign phase3    to nodes "DATA15_in", "DATA14_in", "DATA13_in", "DATA12_in"
!*assign phase3    to nodes "DATA11_in", "DATA10_in", "DATA9_in", "DATA8_in"
!*assign phase3    to nodes "DATA7_in", "DATA6_in", "DATA5_in", "DATA4_in"
!*assign phase3    to nodes "DATA3_in", "DATA2_in", "DATA1_in", "DATA0_in"

!*assign FMT4 to nodes "GCLK"

!*assign FMT5 to nodes "BUSSER_L"

!*assign FMT6 to nodes "RESET_L"

!*assign FMT7 to nodes "ADDVH_L"

!*assign FMT8 to nodes "WRITE"

!*assign FMT9 to nodes "DSYNCH_L"

!*assign window10   to nodes "DATA31_out", "DATA30_out", "DATA29_out"
!*assign window10   to nodes "DATA28_out", "DATA27_out", "DATA26_out"
!*assign window10   to nodes "DATA25_out", "DATA24_out", "DATA23_out"
!*assign window10   to nodes "DATA22_out", "DATA21_out", "DATA20_out"
!*assign window10   to nodes "DATA19_out", "DATA18_out", "DATA17_out"
!*assign window10   to nodes "DATA16_out", "DATA15_out", "DATA14_out"
!*assign window10   to nodes "DATA13_out", "DATA12_out", "DATA11_out"
!*assign window10   to nodes "DATA10_out", "DATA9_out", "DATA8_out",
"DATA7_out"
!*assign window10   to nodes "DATA6_out", "DATA5_out", "DATA4_out", "DATA3_out"
!*assign window10   to nodes "DATA2_out", "DATA1_out", "DATA0_out"

!*inputs formatted FMT4
!*inputs formatted FMT5
!*inputs formatted FMT6
!*inputs formatted FMT7
!*inputs formatted FMT8
!*inputs formatted FMT9

!*inputs addr
!*inputs phase2
!*inputs phase3
!*outputs window10


pcf order is addr , phase2 , phase3 , window10

!— TIMING SECTION —

!*events every 18750p internal
```

Table 3. VCL/PCF Postprocessor Output File...**Continued on page 10**

Table 3. VCL/PCF Postprocessor Output File...**Continued from page 9**

```
!*timing set default TSET1 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "0"
!*        at event 0 set FMT6 to "0"
!*        at event 0 set FMT7 to "0"
!*        at event 0 set FMT8 to "0"
!*        at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET2 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "1"
!*        at event 0 set FMT6 to "1"
!*        at event 0 set FMT7 to "1"
!*        at event 0 set FMT8 to "0"
!*        at event 0 set FMT9 to "1"
!*end timing set

!*timing set TSET3 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "1"
!*        at event 0 set FMT6 to "1"
!*        at event 0 set FMT7 to "1"
!*        at event 0 set FMT8 to "0"
!*        at event 0 set FMT9 to "1"
!*        at event 1 set FMT9 to "0"
!*end timing set

!*timing set TSET4 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "1"
!*        at event 0 set FMT6 to "1"
!*        at event 0 set FMT7 to "1"
!*        at event 1 set FMT7 to "0"
!*        at event 0 set FMT8 to "0"
!*        at event 1 set FMT8 to "1"
!*        at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET5 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "1"
!*        at event 0 set FMT6 to "1"
!*        at event 0 set FMT7 to "0"
!*        at event 1 set FMT7 to "1"
!*        at event 0 set FMT8 to "1"
!*        at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET6 is 10 events
!*      drive addr ,phase2 ,phase3 at event 1
!*      receive window10 at event 1
!*        at event 0 set FMT4 to "0"
!*        at event 5 set FMT4 to "1"
!*        at event 0 set FMT5 to "1"
!*        at event 0 set FMT6 to "1"
!*        at event 0 set FMT7 to "1"
!*        at event 0 set FMT8 to "1"
!*        at event 1 set FMT8 to "0"
!*        at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET7 is 10 events
!* drive addr ,phase2 ,phase3 at event 1
!* receive window10 at event 1
!* at event 0 set FMT4 to "0"
!* at event 5 set FMT4 to "1"
!* at event 0 set FMT5 to "1"
!* at event 0 set FMT6 to "1"
!* at event 0 set FMT7 to "1"
!* at event 1 set FMT7 to "0"
!* at event 0 set FMT8 to "0"
!* at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET8 is 10 events
!* drive addr ,phase2 ,phase3 at event 1
!* receive window10 at event 1
!* at event 0 set FMT4 to "0"
!* at event 5 set FMT4 to "1"
!* at event 0 set FMT5 to "1"
!* at event 0 set FMT6 to "1"
!* at event 0 set FMT7 to "0"
!* at event 1 set FMT7 to "1"
!* at event 0 set FMT8 to "0"
!* at event 0 set FMT9 to "0"
!*end timing set

!*timing set TSET9 is 10 events
!* drive addr ,phase2 ,phase3 at event 1
!* receive window10 at event 1
!* at event 0 set FMT4 to "0"
!* at event 5 set FMT4 to "1"
!* at event 0 set FMT5 to "1"
!* at event 0 set FMT6 to "1"
!* at event 0 set FMT7 to "1"
!* at event 0 set FMT8 to "0"
!* at event 0 set FMT9 to "0"
!*end timing set
! VECTOR EXECUTION

!*unit "PCF blocks"

pcf use timing set TSET2

!START:
"ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 1 BURST 0

use timing set TSET3
"................................................................"
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 2 BURST 0
```

Table 3. VCL/PCF Postprocessor Output File...**Continued on page 11**

Table 3. VCL/PCF Postprocessor Output File...Continued from page 10

```
use timing set TSET4
"1111010000010000000010000000000011000000000000000000000000000000000"
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 3 BURST 0

use timing set TSET5
"................................................................"
"LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
! PATTERN 4 BURST 0

use timing set TSET6
"................................................................"
"................................"
! PATTERN 5 BURST 0

use timing set TSET4
"..............................0011..............................."
"................................"
! PATTERN 6 BURST 0

use timing set TSET5
"................................................................"
"................................"
! PATTERN 7 BURST 0

use timing set TSET6
"................................................................"
"................................"
! PATTERN 8 BURST 0

use timing set TSET4
"..............................1..1100.....1.1....................."
"....X.X........................."
! PATTERN 9 BURST 0

use timing set TSET5
"................................................................"
".....H.H........................"
! PATTERN 10 BURST 0

use timing set TSET6
"................................................................"
"................................"
! PATTERN 11 BURST 0

use timing set TSET4
"..............................0011.....0.0.............1......."
".....X.X.................X......."
! PATTERN 12 BURST 0

use timing set TSET5
"................................................................"
".....L.L.................H......."
! PATTERN 13 BURST 0

use timing set TSET6
"................................................................"
"................................"
! PATTERN 14 BURST 0

use timing set TSET4
"..............................1..0..1100.....................0......."
"........................X......."
! PATTERN 15 BURST 0
```

Table 3. VCL/PCF Postprocessor Output File...Continued on page 12

Table 3. VCL/PCF Postprocessor Output File...Continued on page 11

```
use timing set TSET5
"..."
"....L..."
! PATTERN 16 BURST 0

use timing set TSET6
"..."
"..."
! PATTERN 17 BURST 0

use timing set TSET4
"....0011....1..."
"....X..."
! PATTERN 18 BURST 0

use timing set TSET5
"..."
"....H..."
! PATTERN 19 BURST 0

use timing set TSET6
"..."
"..."
! PATTERN 20 BURST 0

use timing set TSET7
"....0....1100ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 21 BURST 0

use timing set TSET8 "..."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 22 BURST 0

use timing set TSET9
"..."
"LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
! PATTERN 23 BURST 0

"..."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 24 BURST 0

"..."
"..."
! PATTERN 25 BURST 0

"..."
"..."
! PATTERN 26 BURST 0

use timing set TSET7
"....0011...."
"..."
! PATTERN 27 BURST 0

use timing set TSET8
"..."
"LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL"
! PATTERN 28 BURST 0

use timing set TSET9
"..."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 29 BURST 0

"..."
"..."
! PATTERN 30 BURST 0

"..."
"..."
! PATTERN 31 BURST 0

"..."
"..."
! PATTERN 32 BURST 0

use timing set TSET7
"....1..1100...."
"..."
! PATTERN 33 BURST 0

use timing set TSET8
"..."
"..."
! PATTERN 34 BURST 0

use timing set TSET9
"..."
"LLLLLHLHLLLLLLLLLLLLLLLHLLLLLLL"
! PATTERN 35 BURST 0

"..."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 36 BURST 0

"..."
"..."
! PATTERN 37 BURST 0

"..."
"..."
! PATTERN 38 BURST 0

use timing set TSET7
"....0011...."
"..."
! PATTERN 39 BURST 0

use timing set TSET8
"..."
"LLLLLHLHLLLLLLLLLLLLLLLHLLLLLLL"
! PATTERN 40 BURST 0

use timing set TSET9
"..."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
! PATTERN 41 BURST 0

"..."
"..."
! PATTERN 42 BURST 0

"..."
"..."
! PATTERN 43 BURST 0

"..."
"..."
! PATTERN 44 BURST 0
```

Table 3. VCL/PCF Postprocessor Output File...Continued on page 13

Table 3. VCL/PCF Postprocessor Output File...Continued from page 12

```
use timing set TSET7
"..........1..0..1100.........."
"..........."
! PATTERN 45 BURST 0

use timing set TSET8
"..........."
"..........."
! PATTERN 46 BURST 0

use timing set TSET9
"..........."
"LLLLLLLLLLLLLLLLLLLLLLLLLLLLHLLL"
! PATTERN 47 BURST 0

"..........."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XX"
! PATTERN 48 BURST 0

"..........."
"..........."
! PATTERN 49 BURST 0

"..........."
"..........."
! PATTERN 50 BURST 0

use timing set TSET7
"..........0011.........."
"..........."
! PATTERN 51 BURST 0

use timing set TSET8
"..........."
"LLLLLLLLLLLLLLLLLLLLLLLLLLLLHLLL"
! PATTERN 52 BURST 0

use timing set TSET9
"..........."
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XX"
! PATTERN 53 BURST 0

"..........."
"..........."
! PATTERN 54 BURST 0

"..........."
"..........."
! PATTERN 55 BURST 0

"..........."
"..........."
! PATTERN 56 BURST 0

end pcf

!*end unit

! END OF TEST
```

The format of the HP 75000 Result file will be in block definite form. The groups will be in the same order as in the pcf order is statement with all vectors for the first output group followed by all vectors of the second output group, etc.

Block definite form consists of an ASCII header followed by binary bytes. The header is defined as a # character followed by digits. The first digit indicates the number of remaining digits in the header; the remaining digits indicate the number of binary bytes in the block definite data.

The binary bytes in the block definite data will represent each bit in the output group with the most significant byte first. The number of bytes will depend upon the number of bits defined in the group. There can be one, two, three or four bytes per vector. If a group is defined to be other than a multiple of eight, the upper bits of the most significant byte will be meaningless and should be ignored.

Example: Assume window1 is 32 bits long and window2 is 20 bits long, and there are 5 vectors.
pcf order is phase1, phase2, phase3, window1, window2

#
2 (Number of remaining digits in header)
2
0 (20 bytes of binary data to follow)
B41 (Most significant byte, byte 4, of the first vector window1)
B31
B21
B11
B42
B32
B22
B12
B43
B33
B23
B13
B44
B34
B24
B14
B45
B35
B25
B15 (Least significant byte of vector5 of window 1)
2
1
5 (15 bytes of binary data to follow)
B31 (Most significant byte of vector1 of window 2)
B21
B11
B32
B22
B12
B33
B23
B13
B34
B24
B14
B35
B25
B15 (Least significant byte of vector5 of window 2

Table 4. HP 75000 Result File

HEWLETT PACKARD

For more information, call your local HP sales office listed in your telephone directory or an HP regional office listed at right for the location of your nearest sales office.United States:

United States
Hewlett-Packard Company
4 Choke Cherry Road
Rockville, MD 20850
(301) 670 4300

Hewlett-Packard Company
5201 Tollview Drive
Rolling Meadows, IL 60008
(708) 255 9800

Hewlett-Packard Company
5651 W. Manchester
Los Angeles, CA 90045
(213) 337 8035

Hewlett-Packard Company
2000 South Park Place
Atlanta, GA 30339
(404) 980 7351

Canada:
Hewlett-Packard Ltd.
6877 Goreway Drive
Mississauga, Ontario L4V 1M8
(416) 678 9430

Japan:
Yokogawa-Hewlett-Packard Ltd.
15-7, Nishi Shinjuku 4 Chome
Shinjuku-ku
Tokyo 160, Japan
(03) 5371 1351

Mexico:
Hewlett-Packard
Latin American Region Headquarters
Monte Pelvoux No. 111
Lomas de Chapultepec
11000 Mexico, D.F.
(525) 202 0155

Brazil:
(55 11) 709 - 1444

For other Latin American cities consult local telephone directory or call:
(525) 202 0155

Australia/New Zealand:
Hewlett-Packard Australia Ltd.
31-41 Joseph Street
Blackburn, Victoria 3130
Australia (A.C.N. 004 394 763)
(03) 895 2895

Far East:
Hewlett-Packard Asia Ltd.
22/F Bond Centre, West Tower
89 Queensway
Central, Hong Kong
(852) 848 7777

Korea:
(82 2) 769 - 0114

Taiwan:
(886 2) 712 - 0404

Peoples' Republic of China:
(86 1) 505 - 3888

Singapore:
(65) 271 - 9444

Malaysa:
(60 3) 298 - 6555

For other Asian cities consult local telephone directory or call:
(852) 848 7777

In Europe, please call your local HP sales office or representative:

Austria:
(0222) 2500-0

Central Europe, USSR, and Yugoslavia:
Vienna – Austria
(0222) 2500-0

Belgium and Luxembourg:
(02) 761 31 11

Denmark:
(42) 81 66 40

Finland:
(90) 88 721

France:
(1) 69 82 65 00

Germany:
(06172) 16 0

Greece:
(01) 68 28 811

Iceland:
(91) 67 10 00

Ireland:
(01) 88 33 99

Israel:
Computation and
Measurement Systems (CMS) Ltd.
(03) 5380 333

Italy:
(02) 95 300 134

Netherlands:
(020) 547 6669

Norway:
(02) 87 97 00

Portugal:
(11) 301 73 30

Spain:
900 123 123

Sweden:
(08) 750 20 00

Switzerland:
(057) 31 21 11 (Headoffice)
(022) 780 41 11 (Suisse Romande)
(046) 05 15 05 (Customer Information Center)

South Africa:
HiPerformance Systems
(011) 802 5111

Turkey:
175 29 70

U.K.:
(0344) 369 369

Europe/Africa/Middle East:
Hewlett-Packard S.A.
Marcom Operations Europe
P.O. Box 529
1180 AM Amstelveen
The Netherlands

Printed in USA 4/92

5091-4018E