# Precision Architecture RISC Diagnostics Manual, Volume 1

## HP Apollo 9000 Series 700



HEWLETT PACKARD

HP Part No. 09740-90041
Printed in USA August 1991

Edition 2
FOR HP INTERNAL USE ONLY

## Notice

**Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.** Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.



UNIX is a trademark of AT&T Laboratories in the USA and other countries.

**System Technology Division**
**19483 Pruneridge Avenue**
**Cupertino, CA 95014**

## Printing History

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The dates on the title page change only when a new edition or a new update is published. No information is incorporated into a reprinting unless it appears as a prior update; the edition does not change when an update is incorporated.

The software code printed alongside the date indicates the version level of the software product at the time the manual or update was issued. Many product updates and fixes do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.


Edition 1 . . . . . . . . . . . . . . . . . . . . . . . . . . February 1991

Update 1 . . . . . . . . . . . . . . . . . . . . . . . . . . May 1991

Update 2 . . . . . . . . . . . . . . . . . . . . . . . . . . July 1991

Edition 2 . . . . . . . . . . . . . . . . . . . . . . . . . . August 1991

## List of Effective Pages

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. No information is incorporated into a reprinting unless it appears as a prior update.

| | |
|---|---|
| All | February 1991 |
| Title Page | May 1991 |
| ii to vii | May 1991 |
| 3a-1 to 3a-62 | May 1991 |
| 4a-1 to 4a-50 | May 1991 |
| 9-1 to 9-11 | May 1991 |
| Title Page | July 1991 |
| iii to iv | July 1991 |
| Contents-1 to Contents-8 | July 1991 |
| 6a-1 to 6a-39 | July 1991 |
| All | August 1991 |

## Safety and Regulatory Information

For your protection this product has been tested to various national and international regulations and standards. The scope of this regulatory testing includes electrical/mechanical safety, radio frequency interference, ergonomics, acoustics, and hazardous materials. Where required, approvals obtained from third-party test agencies are shown on the product label. In addition, various regulatory bodies require some of the information under the following headings.

### USA Radio Frequency Interference

The United States Federal Communications Commission (in 47CFR Subpart J, of Part 15) has specified that the following notice be brought to the attention of the users of this product:

**Warning**

**This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested for compliance with the limits for Class A computing devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.**

### Japanese Radio Frequency Interference

The following notice is for users of this product in Japan:

この装置は，第一種情報装置（商工業地域において使用されるべき情報装置）で商工業地域での電波障害防止を目的とした情報処理装置等電波障害自主規制協議会（VCCI）基準に適合しております。

従って，住宅地域またはその隣接した地域で使用すると，ラジオ，テレビジョン受信機等に受信障害を与えることがあります。

取扱説明書に従って正しい取り扱いをして下さい。

**Japanese Radio Frequency Notice**

## Safety Considerations

This product and related documentation must be reviewed for familiarization with safety markings and instructions before operation. The following figure shows some of the safety symbols used on the product to indicate various safety considerations.

**SAFETY SYMBOLS**

| | |
|---|---|
| ⚠ | Instruction manual symbol: the product will be marked with this symbol when it is necessary for the user to refer to the instruction manual in order to protect the product against damage. |
| ϟ | Indicates hazardous voltages. |
| ⏚ | Indicates earth (ground) terminal (sometimes used in manual to indicate circuit common connected to grounded chassis). |

**Warning**

**The WARNING sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not done correctly or adhered to, could result in injury. Do not proceed beyond a WARNING sign until the indicated conditions are fully understood and met.**

**Caution**

The CAUTION sign denotes a hazard. It calls attention to an operating procedure, practice, or the like, which, if not done correctly or adhered to, could damage or destroy part or all of the product. Do not proceed beyond a CAUTION sign until the indicated conditions are fully understood and met.

# Preface

This manual contains information about the online diagnostics and ISL diagnostics/utilities for the HP Apollo Series 700 Precision Architecture RISC Workstation. It is intended to be used as technical support hardware documentation for Hewlett-Packard CEs, CEC Engineers, SEs, and other qualified support personnel. The procedures and software focus primarily on the hardware troubleshooting environment, and require specific training for correct and safe usage.

# Contents

# Contents

# 1

# Online Diagnostics Overview

## Introduction

The online diagnostics subsystem provides a means of testing hardware modules and devices attached to the HP Precision Architecture RISC computer system. The PA-RISC system and the diagnostic subsystem are intimately tied together for error logging, and for restricting access to other users during diagnostic testing.

The subsystem provides a common, standard user interface to all the diagnostic programs and utilities, as well as a controlling mechanism for diagnostic access to I/O devices on the system. Finally, the subsystem can control the normal I/O error logging process, allowing dynamic display of errors as they occur.

Each major hardware component or aspect of the PA-RISC system can be tested by a diagnostic. Each diagnostic is described separately in the following chapters. All of the diagnostic/utility chapters share the same format.

| | |
|---|---|
| Introduction | Brief explanation of the purpose and nature of the diagnostic. |
| Defects and Enhancements | STARS Database product number for comments about the diagnostic. |
| Minimum Configuration | Necessary hardware and software to run the diagnostic. |
| Operating Instructions | Information about how to start the diagnostic. |
| Default Tests | Lists the tests that are automatically executed if no test sections are specified in the RUN command. |
| RUN Command | Sample RUN command using the typical loaded system configuration described in this chapter. |
| Test Execution | What happens after the diagnostic is started. |
| Test Section Descriptions | What is being tested by each section of the diagnostic. |
| Commands | Commands available during the diagnostic (if any). |
| Error and Warning Messages | Lists all error and warning messages displayed by the diagnostic along with a probable cause and suggested action statements. |

## Operating Requirements

In order to support the online diagnostics subsystem, a PA-RISC computer system must be up and running. User access to at least one functioning terminal is also required.

## Online Diagnostic Subsystem Components

The online diagnostics subsystem is composed of the the Diagnostic User Interface (DUI) and the diagnostic programs which can be run using the operating system.

### Diagnostic User Interface

The DUI provides the communication link between the user and the diagnostic system. The DUI provides such functions as sending messages to the user from diagnostic programs, and returning replies to the appropriate section(s) of the online diagnostic software.

### Diagnostic Programs

The diagnostic programs are a comprehensive set of software to test the devices and components supported on the HP Precision Architecture RISC computer family.

Diagnostic programs are divided into three groups: diagnostics and verifiers; utilities and tools; and system exercisers. Diagnostics are programs that can determine which field replaceable units (FRUs) are malfunctioning.

Verifiers cannot isolate defective FRUs, but can verify which functions of the device are operating properly. Verifiers can determine probable cause of device failures or aid the user in making such determinations. Some diagnostics and verifiers provide thorough tests of the internal I/O modules as well as complete functional tests and system type tests for peripheral devices.

Utilities and tools provide a means for obtaining system information or performing specific I/O operations. System exercisers provide a means of using (loading) a particular part of the system. These programs provide a way of using system resources under stress conditions that equal or exceed those expected under maximum load.

External excercisers are interactive programs provided for some diagnostics to provide the user with access to the set of internal diagnostics and utilities within a particular device.

The next section describes the diagnostics available on the system.

### Available Diagnostics

The following diagnostic programs and utilities, along with the Diagnostic User Interface (DUI) and any background processes, are currently supported on the system:

**Diagnostics**

- Memory Diagnostic (MEMDIAG)
- LAN Diagnostic (LANDAD)
- SCSI CD Diagnostic (SCSICD)
- SCSI Digital Data Storage Tape Drive Diagnostic (SCSIDDS)

**Online Diagnostics Subsystem Operating Software**

- Diagnostic User Interface (DUI)

## DUI Modes

The diagnostic system provides three modes of operation for each diagnostic program: disruptive mode, destructive mode, and normal mode. The diagnostic system determines the mode in which each diagnostic program is allowed to run by considering such things as the device being tested, and the user mode in which the system is running. When the diagnostic program requests access to a device, either at program initiation or at some other time, it is told which mode in which to run via a device control procedure.

In general, the diagnostic is usually granted destructive mode unless the selected device is a system disk or exclusive access to the device cannot be obtained for the diagnostic. The diagnostic program must decide which tests can be run in the mode it was given.

Disruptive Mode
: In disruptive mode, the program can run tests of a "disruptive" nature on the selected device. A disruptive test does not destroy any data on the device, but could cause errors for other users on the system. For example, the internal selftest on a system disk is disruptive, since the disk temporarily goes offline to perform the test, causing errors for others who try to access the disk at the same time.

Destructive Mode
: In destructive mode, the program may run any test on the selected device. This mode is required for tests that have the potential for corrupting data on the device being tested. There are virtually no restrictions on tests run in this mode and, therefore, this mode is handled with extreme care by the diagnostic program. An example of a destructive test is one that reformats the media on a system disk, thus destroying all of the data on it.

Normal Mode
: In normal mode, the diagnostic program cannot run any tests on the selected device that are potentially destructive or disruptive in nature.

## User Modes

There are three user modes available: Single User Mode, Multi-User Mode (normal state), and Single Disk Mode.

| | |
|---|---|
| Single User | Can be selected by a user with the required capability. The primary purpose of Single User Mode is for testing that may cause data integrity problems. Typically, it is used only in the event of a major problem with the system hardware. |
| Multi-User Mode | Can be selected by a user with the required capability. |
| Single Disk Mode | Is selectable only on system boot-up and is system specific. |

## Security

Four levels of security are available for users in the online diagnostics subsystem. Access to the various programs is restricted by security level. In addition, each program may restrict certain functionality to users of varions security levels.

| | |
|---|---|
| Level 0 | The highest security; the user may install, remove, or update programs through a utility program and may do anything that a user at level 1 may do. |
| Level 1 | The user may perform destructive tests, read or modify data on any device, may enter SUM or MUM modes, and do anything that a user at level 2 may do. |
| Level 2 | The user may perform disruptive tests, but may not display or modify user data, and may do anything that a user at level 3 may do. |
| Level 3 | The user may run non-disruptive tests only. |

The following table lists the user capabilities required for each security level:

| Security | |
|---|---|
| Level 0 | Superuser; Configurable |
| Level 1 | Configurable |
| Level 2 | Configurable |
| Level 3 | Users not in /usr/diag/security |

# Contents

2

# Diagnostic User Interface

## Introduction

The Diagnostic User Interface (DUI) enables users to run on-line diagnostic programs on PA-RISC systems through a common, consistent interface. Users need only learn one set of commands to run, process, and manipulate all on-line diagnostic programs.

Numerous commands are available to start, stop, monitor, add, and delete diagnostic programs.

Various utility functions are also provided through the DUI - the output of a diagnostic may be redirected to a file and/or echoed to a hardcopy device; the inputs to a diagnostic program may be obtained from a file rather than from the user's terminal; part of a diagnostic session may be controlled from a command file. The complete set of all such functions will be found in the "COMMANDS" section of this document.

All of the commands accessible through the DUI are invoked identically on all systems on which the online diagnostics subsystem runs. Each individual command has the identical effect on every system on which the online diagnostics subsystem runs.

Diagnostic programs which run under the DUI include:

- programs which test peripheral devices and individual circuit boards and attempt to isolate a failure to a FRU (Field Replaceable Unit)
- programs which load (or stress) a system in order to recreate the conditions under which a system might exhibit failures
- programs which can isolate a "functional" failure, such as a read or a write failure, even if they cannot pinpoint the hardware involved in the failure
- programs which perform utility functions useful to a diagnostician such as creating a map of the system configuration or displaying the contents of log files.

The diagnostic programs available through the DUI may vary from system to system, but all diagnostic programs and associated commands are invoked identically no matter what system they run on.

**Note**

This document discusses the commands

- DEFAULT
- FOREGROUND
- REPLY
- SET
- SHOWDEFAULT
- SHOWPARMS

and refers to a programmatic interface and the ability to run multiple diagnostics using just one `run` command. None of these features is implemented in the first release. If one of the commands listed above is invoked, nothing will be done. The `DUI` prompt will reappear immediately after the command is given.

If more than one diagnostic is named within the `run` command, an error will be printed, and the `DUI` prompt will reappear. The user may then re-enter the command.

## Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database referencing product number 30600-10023.

## Invoking the DUI

To enter the online diagnostics subsystem, one enters the command "SYSDIAG" at the system's prompt. Upon entering the DUI, the following banner will be displayed:

```
********************************************************************
******                                                        ******
******              ONLINE DIAGNOSTIC SYSTEM                  ******
******                                                        ******
****** (C) Copyright Hewlett Packard Co. 1987, 1989           ******
******                All Rights Reserved                     ******
******                                                        ******
******               DUI Version: A.02.03                     ******
******          Diagnostic Monitor Version   A.02.02          ******
******                                                        ******
********************************************************************

     Type "HELP" for assistance.

WARNING:  This Diagnostic System has been designed as a tool for
Hewlett-Packard service personnel only.  Incorrect use of this
System could result in loss of data and additional Hewlett-Packard
support charges.  Hewlett-Packard will not be responsible for
consequences resulting from unauthorized use of the Diagnostic System.

DUI >
```

The user is now free to enter diagnostic system commands. To exit the DUI, the user simply types EXIT.

## Version Identification

The DUI's version number is displayed in the banner. The format of the version is:

VERSION.UPDATE.FIX

Where *version* refers to major changes in large groups of system software, each *update* indicates a major change or addition to the Diagnostic User Interface, and a *fix* indicates bug fixes within the DUI.

## Entering Commands

### Replies and Responses

The user may be prompted from time to time for certain data. If one of several specific responses, such as **yes** or **no**, is required, the valid responses will be indicated in parenthesis. The default response - the assumed response if nothing is given but a carriage return - will be indicated in square brackets ([]). For example, if the user wants to exit and there are still running processes, he might be asked the question:

```
Do you wish to abort the currently running processes (Y/N) [N] ?
```

Valid responses include `Y`, `N`, **yes**, and **no**. Simply hitting the <RETURN> key selects the default response of `N`.

### Continuation Lines

If a command is too long to fit on one line, the user may continue the command on subsequent lines by using the continuation character (\). This character causes the `DUI` to delay interpretation of the command. Usually, interpretation would begin immediately after the receipt of the carriage return. When the continuation character is given at the end of a command line (followed by a carriage return), the `DUI` prompt will appear on the next line - the user may then continue to type the command. A command may consist of numerous continuation lines provided that the total length of the text entered does not exceed 255 characters. For example:

```
DUI > run wizbang errcount=9 \
DUI >> ldev3  erronly
```

Note that the `DUI` prompt is slightly different to indicate that a continuation of the previous line is expected.

### Command Comments

Command lines may include comments if desired. The beginning of a comment is indicated by the # character; the end-of-line automatically terminates the comment.

```
DUI > list             # This is a comment

DUI> # This is a
DUI> #multi-line comment
```

## Installation Instructions

The DUI will be shipped to customers already installed in all releases; no user activity will be required. The set of diagnostic programs will also be installed in all releases prior to being shipped to customers. If a diagnostic program must be installed into the On-Line Diagnostic system at a customer's site the **install** command may be used at the DUI prompt.

## Input and Output Files

The DUI works with *usefile* files, *infile* files, and *outfile* files. (See the USEFILE and OUTFILE commands and the INFILE and OUTFILE run command modifiers.) *Usefile* files and *infile* files are assumed to be comprised of lines 80 (ASCII) characters in length. *Outfile* files, which are made by the DUI, will also consist of 80 character length lines.

*Usefiles*, *infiles*, and *outfiles* may reside in any directory as long as the user has the appropriate capability to access them. The complete path name may be specified for all files and must be specified if the file resides somewhere other than in the directory the user is running the DUI from.

The form of complete path names differ among operating systems. For example, on MPE XL a complete path name for **somefile** might be **somefile.mygroup.official** or just **somefile.mygroup**, while on HP-UX **/user/mydir/somefile** might be appropriate. It is assumed that the user is familiar enough with the file system organization in the operating system being used to be able to specify the appropriate path.

If the file name provided for an *outfile* already exists, it will not be overwritten, rather an error message will be printed.

## Interactive Interface

The normal mode of operation of the DUI is interactive; however, interaction may be mimicked through a *usefile* (a file containing commands to be executed along with information the user would normally give interactively during the execution of those commands). Data which would usually be sent to a diagnostic by typing the information in at the user's terminal may also be entered automatically if an *infile* is specified. Conversely, information which would usually be sent directly from the diagnostic system to the user's terminal may be redirected by using an *outfile* or by giving the **hardcopy** command or modifier.

*Usefile*, *infile*, *outfile*, and **hardcopy** are more fully explained in their own sections of this document.

## Programmatic Interface

A simple programmatic interface to the DUI exists. A program may invoke the DUI by using whatever construct is provided in the operating system being used to execute a command interpreter or shell command from within a program. The command to be executed is SYSDIAG. The program may send a one line command, in the form of an information string, to the DUI.

The one line may be any command the DUI would normally accept, including the **usefile** command. The DUI will terminate immediately after executing the one line command.

For example, on MPE XL the intrinsic **create** or **createprocess** could be used while on HP-UX one of the **exec** system calls would be used.

## Interrupts

The effects of typing a user interrupt while in the DUI depend upon what action is taking place. Interrupts will only affect diagnostic programs and processes which are running in the foreground. Background processes cannot be interrupted.

- If no program is running in the foreground and no USEFILE file is being used
  - "**** INTERRUPT ***" will be printed on the standard output device
  - The DUI prompt will reappear
  - Nothing else will be done.
- If executing a USEFILE
  - "**** INTERRUPT ***" will be printed on the standard output device
  - Reading of the *usefile* is stopped and the *usefile* is closed. All *usefiles* nested with the current *usefile* are also closed.
  - If a program is running in the foreground it will either receive a message that an interrupt has occurred or will be suspended as explained below.
- If a program has requested control of its own interrupts
  - The program will be informed of the interrupt. What then occurs is program dependent.
- If a program is running in the foreground and has not requested control of its own interrupts
  - "**** INTERRUPT ***" will be printed on the standard output device
  - The program will be SUSPENDED.
  - The DUI prompt will reappear

The control characters used to generate a user interrupt are operating system dependent. On MPE XL (CTRL-Y) works; on HP-UX (CTRL-C).

## Security

Many diagnostic tests have the indirect effect of corrupting data (e.g., selftests which write into onboard registers). To ensure system and user data integrity, the online diagnostics subsystem has its own security mechanisms which augment the normal operating system security. The online diagnostics subsystem also uses its own security mechanisms to ensure diagnostic system integrity.

Commands given to the `DUI` will not be executed if the caller has too low a security level. Security level is sometimes called capability level - the two are synonymous.

The online diagnostics subsystem maps the user's operating system security into one of four diagnostic system security levels:

1. diagnostic security level 0
2. diagnostic security level 1
3. diagnostic security level 2
4. diagnostic security level 3

Diagnostic security level 0 is the highest; level 3 is the lowest.

The correspondence between the user's operating system security level and the user's diagnostic security level is operating system dependent. This is necessary because the different operating systems vary in the way different classes of security are defined and in the kinds of security granted different users. However, some general statements may be made as to the diagnostic security levels various users would be given.

An HP-UX superuser (root) or an MPE XL system manager would be assigned the highest diagnostic security level, level 0, and could perform any action the diagnostic system was capable of performing. A user who had only the lowest level of security available on an operating system would be assigned the lowest diagnostic security level, level 3, and could perform only those actions which were non-destructive and which did not require exclusive access to a device. This would include, in most cases, such actions as running diagnostic sections which copied and decoded the *identify* block from a hardware board.

Users whose operating system security fell somewhere between the lowest and highest would be assigned diagnostic security level 1 or level 2 and could perform such actions as sending loopbacks to devices. But again, the diagnostic security level assigned to a user and hence the actual tests and actions which a particular user could perform are operating system dependent.

Users on MPE XL with operating system capability `sm` are given diagnostic security level 0. Those with `di` capability are given diagnostic security level 1. Those with `op` capability are given diagnostic security level 2. All other users are assigned diagnostic security level 3.

An HP-UX user is assigned a diagnostic security level based on whether or not the user has an entry in the diagnostic SECURITY file. Each entry in this file consists of a user's login name, along with a number. Those users whose names are followed by the number "0", such as root, are given diagnostic security level 0. Those users whose names are followed by "1" are given diagnostic security level 1. Those whose names are followed by "2" or "3" are given diagnostic security level 2 or 3, respectively. Thus, a SECURITY file might look like this:

```
root:0
jdoe:2
jroe:0
tsmith:1
```

If the SECURITY file cannot be found, or the user's login name cannot be found in it, the user is assigned diagnostic security level 3.

The HP-UX diagnostic SECURITY file is maintained by the system administrator, and can only be modified by a superuser. The SECURITY file is located in the same directory as the other major diagnostic files, normally /usr/diag/bin. Please see the system administrator if a new entry is needed in this file.

Every action which might be performed through the online diagnostics subsystem is associated with one of four diagnostic security states.

These states are:

1. non-destructive/non-exclusive
2. non-destructive/exclusive
3. destructive/non-exclusive
4. destructive/exclusive

*Non-destructive* and *destructive* (the two possible test modes) indicate the possibility that an action could destroy data. *Non-exclusive* and *exclusive* (the two possible access modes) refer to whether or not a device must be accessed exclusively (locking all other processes out during the time of use) to perform an action.

Many actions do not directly involve devices so, for those actions, the access mode is moot. For example, purging a diagnostic using the PURGE command does not directly involve a device but has a drastic effect on the diagnostic system; the DUI would only allow a user whose diagnostic system security level was such that he could run destructive tests to execute the PURGE command.

The online diagnostics subsystem checks the user's diagnostic security level, whether the device being tested is being accessed exclusively or non-exclusively (access mode), and the level of potential destructiveness (test mode) of the requested tests or the given commands before allowing any particular set of tests to run or commands to execute. The state of the operating system also affects which commands and tests a user can run through the DUI.

For example, a user whose diagnostic security level was such that he would normally not be allowed to perform a *destructive* action on a *non-exclusive* device would be allowed to do so if the operating system had been booted in single-user mode.

If a user attempts to run a test or execute a command which requires a higher security level than he possesses, a message will be printed explaining why the action could not be performed.

# COMMANDS - GENERAL INFORMATION

## Notation and Special Symbols

The following notation is used in the command and command modifier syntax diagrams. The notation is also used in messages printed in response to the **help** command.

[] An element inside brackets is *optional*.
Several elements stacked inside brackets means the user may select any one or none of these elements. For example:

```
[A]
[B]
[C]  User may select A or B or C or none.
```

When brackets are nested, parameters in inner brackets may be specified only if parameters in outer brackets are specified.

For example:
*[parm1 [parm2 [parm3]]]*

may be entered as
*parm1 parm2 parm3*
or
*parm1*
or
*parm1 parm2*

Optional parameters which are not positional are shown as follows:

*[parm1][parm2]*

[,...]
or
[...] Means that the immediately preceding item in the syntax diagram may be repeated any number of times. The [,...] form means that each instance of the repeated item must be preceded by a comma.

{} An element inside curly braces is *required*.
When several elements are stacked within braces

in a syntax statement, the user must select one of those elements. For example:

```
{A}
{B}
{C}  User must select either A or B or C.
```

| | Vertical parallel lines indicate that any or none of the options may be used in any sequence. However, the elements may not be repeated. For example:

```
|A|
|B|
|C|  The user may choose A, B, and C; or C and A,
     or B alone, etc.
```

UPPERCASE Represent literals which are to be entered exactly as shown except that they may be entered in lower case. Also, if an abbreviation or alternate token is listed for the item, a substitution is allowed. Lastly, only the first *n* characters which establish uniquely what the literal is need be entered.

Special Character Literals

The special characters + - / ( ) " * are literals to be entered exactly as shown in the syntax diagrams.

*italics* Items printed in *italics* are to be replaced with user supplied information.

## Delimiters and Abbreviations

All keywords and options may be abbreviated to the shortest number of characters which make the token unique. Delimiters for keywords and options include spaces and/or semicolons (;). Other delimiters may be specified in the descriptions and syntax diagrams for individual commands and command modifiers.

UPPER and lower case command lines are equivalent.

## User Set Default Values

The user may specify default values which differ from the normal diagnostic system default values for command modifiers using the **set** command.

The user may set a "universal" default value for a modifier - that is, a value which will have effect at all times - or a default value which will take effect only when a particular diagnostic is run. In either case, if another value for the modifier is given in a **run** command, that value will have effect only until the diagnostic named in the **run** command completes execution.

Modifier default values which have been changed, either universally or for individual diagnostics, using the **set** command can be reset to their diagnostic system default values using the **default** command.

For more information, please see the **set** command and the **default** command descriptions in this document.

## Designating Devices to be Tested

Usually, when a diagnostic program is invoked the device to be tested must be named. This can be done in one of two ways: either by using the logical name of the device or giving the physical path address of the device. That is, one of the two command modifiers **ldev** or **pdev** must be given with the **run** command, or must be set using the **set** command.

The form these strings take varies among the operating systems and even from one HPPA machine to another.

For example, under the MPE XL operating system the logical name (**ldev**) of a device is a number (e.g., 3) while in the HP-UX operating system the logical name of a device is that of a special device file (e.g., dsk/c0d0).

The physical path address (**pdev**) of a device is a series of numbers separated by various punctuation marks. Each number in the series corresponds to a physical connection along the electrical pathway to the device. For example, an HPIB device adapter might have an address of 4.2 on an HP3000 series 930, but an address of 2/8.0 on an HP9000 series 850.

It is assumed that the user of the diagnostic system knows enough about the operating system and machine being worked with to determine the correct **ldev** or **pdev** for the device being tested. A utility program, SYSMAP, is provided in some diagnostic installations and may be run through the DUI (SYSMAP may NOT be available in all installations) to help determine the desired **ldev** or **pdev**.

## Running Multiple Diagnostics

Several diagnostic programs may be invoked simultaneously using the **run** command. To do this, the user simply names all of the diagnostics to be run along with the command modifiers for each in a **run** command.

Examples:

```
DUI> run xdiag ldev 0 loop 5 erronly ydiag pdev 4.3.2 \
DUI>> errpause background outfile yout

DUI> qdiag ldev 3 infile qinf & rdiag ldev 0 \
DUI>> sc 4/5(20,23)
```

All but one of the programs must be run in the background; only one program may run in the foreground at any one time.

It is suggested that the output of programs run in the background be redirected to an *outfile*. The `DUI` will not impose any order on the messages received from multiple diagnostic programs but will output each message as it is received.

When multiple diagnostic programs are invoked, the `DUI` will do all the internal checks and initial set up it would usually do for each of the diagnostics. THEN it will launch the diagnostics one immediately after the other using the mechanisms provided by the operating system. The `DUI` will not wait for one of the diagnostics to complete before launching the next.

The number of diagnostics which may be run simultaneously is dependent on the number of processes the operating system will allow any one user to run simultaneously - the diagnostic system itself does not impose a limit.

## Command Summary

The following is a list and brief description of each of the commands available in the DUI.

| Command Name | Description |
|---|---|
| ABORT | Terminates active diagnostic program or utility. |
| CI | Provides access to the operating system command interpreter. |
| CODETEST | For HP internal use only. |
| DEFAULT | Resets command modifiers to their original diagnostic system default values. |
| DIAGSYSTEM | Used to access and manipulate internal diagnostic system processes which are usually not accessible to the user. |
| DO | Allows the user to re-execute any command in the command line history stack. It also permits the user to edit the command before re-executing it. |
| EXIT | Terminates the DUI and returns control to the operating system. |
| FOREGROUND | Moves a diagnostic which has been running in the background into the foreground. |
| HARDCOPY | Causes all terminal input/output to be echoed to the system printer. |
| HELP | Accesses HELP facility for information about the DUI and its commands or for information about any of the diagnostic programs. |
| INSTALL | Allows the user to install diagnostic programs. |
| LIST | Provides information about any or all of the programs in the diagnostic system. |
| LISTREDO | Displays the user's command history stack. |
| MODE | Displays and/or alters the current operating system mode. |
| MODIFY | Allows a user to change information about a diagnostic program without needing to reinstall the diagnostic. |
| OUTFILE | Causes all diagnostic system input and output to be written into the specified file. |
| PURGE | Removes programs from the diagnostic library. |
| REDO | Allows user to display (for command editing) and re-execute any command in the command line history stack. |
| REDOLOAD | Replaces the user's command history stack with the history stack which was saved by the command REDOSAVE. |
| REDOSAVE | Causes all or part of the user's command history stack to be saved into a file. |

| | |
|---|---|
| REDOSIZE | Allows the user to set the maximum number of commands which will be saved in the user's command history stack. |
| REPLY | Used to send a reply to the prompt of a diagnostic program which is running in the background. |
| RESUME | Resumes processing of a suspended program. |
| RUN | Loads and executes a specified program. |
| SETVAR | Allows the user to explicitly set the values of environmental variables which control various features of the user interface. |
| SET | Allows the user to explicitly reset the system default values for modifiers which may be given on a run command line. |
| SHOWACTIVE | Lists programs which are currently active for the user. |
| SHOWDEFAULT | Causes a list of command modifiers and their default values to be displayed. |
| SHOWPARMS | Causes a list of user settable modifiers and their current values to be displayed. |
| SHOWSTATE | Will cause the current system mode (single-user or multi-user) and the user's security capability to be displayed. |
| SUSPEND | Suspends processing of a specified program. |
| UNLOCK | Releases specified device from lock status. |
| USEFILE | Causes input to the diagnostic system to be input from the specified file rather than from a user's terminal. |

The following pages provide a detailed description of each of the available DUI commands. The description includes information about syntax, options available for each command, limitations of each command, and examples.

# ABORT

The abort command terminates a program.

The chosen diagnostic program will be aborted regardless of what it is doing. The diagnostic system may perform clean up actions on any devices allocated to the program but this is not guaranteed. If more than one program is running or in a suspended state, the *program-id* modifier or ALL modifier should be given. If only one program is running or suspended when the ABORT command is given no modifier is necessary.

**Syntax:**

```
Abort [ALL                ] [UNCONDITIONAL]
      [program-id [[,]... ]] [UNCONDITIONAL]
```

**Modifiers:**

| | |
|---|---|
| *program-id* | The process identifier number of the diagnostic program to be aborted. If there is more than one active program and the *program-id* is not specified, a list of *program-ids* from which to choose will appear. |
| ALL | Abort all programs running under the diagnostic system. This will cause all programs running under the DUI through which the abort command was received to terminate. |
| UNCONDITIONAL | Has no effect in online diagnostics subsystem; all aborts are unconditional. |

**Command Examples:**

```
DUI> abort 23

DUI> abort

DUI> abort 4 32
```

# CI

CI (command interpreter) invokes the system's command interpreter or shell so that one or more operating system commands may be executed.

If the CI command is given without a modifier, the command interpreter prompt, which is system specific, will appear. Command interpreter commands can then be given until the user specifically exits back to the DUI. The command used to exit back to the DUI is system specific - **exit** on MPE XL and on HP-UX systems.

If the CI command is given with a *command* that one command will be executed and the DUI prompt will reappear.

**Syntax:**

CI [*command*]

**Abbreviations and Alternative Tokens:**

!
:

**Modifiers:**

*command* The command (and its arguments) which is to be executed by the system's command interpreter or shell.

**Command Examples:**

```
DUI> ci
%                          {the command interpreter's prompt}

DUI> : listf

DUI> !ls
```

**Limitations:**

**Warning**

**If the operating system command to exit a session or to begin a new session is given (e.g.,** logout **on HP-UX or** bye **on MPE XL), the** DUI **will be terminated. No clean up will be done. This is especially dangerous if the system was placed into single-user mode through the** DUI **(see** MODE **command). The system may need to be rebooted before anyone, including the console operator, will be able to log on.**

## CODETEST

This command is provided for Hewlett-Packard internal use only.

## DEFAULT

DEFAULT resets command modifiers to their original diagnostic system default values. (The modifiers could have had their values reset by the set command.)

DEFAULT will have no effect if it is used with a modifier which still has its original diagnostic system default value. No error message or warning will be printed.

If the command is given without an argument the effect is the same as if the user typed DEFAULT ALL.

**Syntax:**

```
DEFAULT  [command modifier [[,]...]]
         [ALL                      ]
```

**Modifiers:**

*command modifier* One of the following

```
BACKGROUND
DEBUG
ERRCOUNT
ERRONLY
ERRPAUSE
ERRPRINT
HARDCOPY
INFILE
LDEV
LOOP
OUTFILE
PDEV
SECTIONS
TRACE
Diagnostic Specific Parameters
```

Please note that this list includes all run command modifiers except SECURITY.

The "Diagnostic Specific Parameters" can be reset to null - the diagnostic system default - by giving a pair of double quotes ("") as a modifier to DEFAULT.

ALL — Resets all modifiers, including the "Diagnostic Specific Parameters" modifier to their diagnostic system default values.

**Command Examples:**

```
DUI> default loop,"",trace,hardcopy

DUI> default all
```

**Related Commands:**

```
SET
SHOWDEFAULT
SHOWPARMS
```

# DIAGSYSTEM

DIAGSYSTEM is used to access and manipulate internal diagnostic system processes which are usually not accessible to the user. DIAGSYSTEM may be used to get a listing of running diagnostic system processes along with their program identifiers; abort diagnostic system processes; and launch diagnostic system processes. These processes run outside of the user interface and their execution cannot be altered by the user. For example, some diagnostic system software is sold by HP as an add-on product - DIAGSYSTEM allows this software to be added to a customer's system without needing to do an `xinstall` and `reboot`. This command also obviates the need to reboot the system to launch diagnostic system processes. Examples of such processes would be diagnostic logging processes and diagnostic statistical analysis processes.

**Syntax:**

```
DIAGSYSTEM {SHOWACTIVE   }
           {ABORT prog_id}
           {RUN prog_name}
```

**Abbreviatons and Alternate Tokens:**

DS

**Modifiers:**

| | |
|---|---|
| SHOWACTIVE | Display a list of running diagnostic system processes along with their program identifiers. |
| ABORT *prog_id* | Terminate the diagnostic system process specified by the given program identifier. |
| RUN *prog_name* | Launch the diagnostic system process specified by *prog_name*. |

**Command Examples:**

```
DUI> diagsystem showactive

DUI> ds run memlogp

DUI> diagsystem abort 33
```

**Limitations:**

Only a user with the highest diagnostic security level (diagnostic security level 0) will be allowed to abort a diagnostic system process.

# DO

DO allows the user to re-execute any command in the command line history stack. It also permits the user to edit the command before re-executing it. The (edited) command is executed immediately after the carriage-return; no interactive editing may occur. (The REDO command must be used if interactive editing is desired.)

**Syntax:**

DO [[CMD=]*command-id*] [, *edit-directives* ]
[;EDIT= *edit-directives*]

**Modifiers:**

*command-id* — Specifies the command to re-execute. The command may be specified by its relative or absolute order in the command line history stack, or by name (as a string) in whole or in part. The default *command-id* is -1, the most recent command. An error is detected if the *command-id* does not exist in the command line history stack.

| *COMMAND-ID* | EXECUTES |
|---|---|
| (omitted) | The most recent command (same as DO -1). |
| -n | nth command before the most recent one. N is a number in the command line stack relative to the most recent command, which is -1. |
| m | Command number m in the command line stack. The number m is absolute (not relative). |
| string | The most recent command beginning with this string. |

*edit-directives* — String specifying the changes to be made in the command represented by the *command-id* before its re-execution. If no *edit-directives* are given, the command is re-executed immediately - no editing is performed.

**Note**

Both *command-id* and *edit-directives* must be surrounded by quote marks ("or ') if they contain any delimiters such as: , ; " ' [ ] = or a space.

When an editing command such as replace, insert, delete, or change is to take effect anywhere other than in column 1 of the original command, the position (in the original command string) where the edit should begin can be designated by preceding the edit command with spaces.

Example:

```
: diuspend     mistyped command
: do ,drs      delete the 'd' in col. 1 and replace
                 the 'i' with an 's'
: suspend      result of edit


: suspllnd          mistyped command
: do ,"     ddie"   delete the fifth and sixth characters
                      and insert an 'e'
: suspend           result of edit
```

---

The editing directives which may be used as *edit-directives* are:

| DIRECTIVE | EFFECT | |
|---|---|---|
| i | INSERT. | If text follows the i, the text following i is inserted in the current line at the position after the i. |
| r | REPLACE. | If text follows the r, the text following r replaces the same number of characters in the current line, beginning at the position of r. |
| d | DELETE. | Deletes a character from the current line for each d specified in the edit line. Note that "d d" does not specify a range, but simply deletes one character from the position above each d. Multiple d's may be followed by an insert or replace operation. |
| d> | DELETE. | Deletes to the end of the current line from the position specified by d>. It may be followed by an INSERT or REPLACE operation. |
| > | APPEND. | > followed by text appends the text to the end of the current line. If > is positioned beyond the end of the current line, then a replacement is performed instead. |
| >d | DELETE. | Deletes from the end of the current line, right-to-left. Multiple d's may be specified after >, as well as INSERT and REPLACE strings. |

| | | |
|---|---|---|
| >r | REPLACE. | Replaces characters at the end of the command line. The replacement is done so that the last (rightmost) character of the replacement string is at the end of the line. |
| c | CHANGE. | Changes all occurrences of one string into another in the current line when the searched for string and the replacement string are properly delimited. A proper delimiter is a non-alphabetic character: ', ", /, etc.<br><br>The substitution is specified as:<br><br>c<delim>*search-string*<delim><br>[*replace-string*[<delim>]]<br><br>Omitting the *replace-string* causes occurrences of the *search-string* to be deleted. |
| *other* | | Simple replacement. Any other character (not i, r, d, d>, >, >d, c, or u) causes that character to be replaced in the current line at the position indicated by the character. In fact, simple replacement also occurs for the editing characters i, r, c, or > if they are not followed by text; or if > appears at or beyond the current end of line. |

---

**Command Examples:**

EDITING SAMPLES

Practical uses of the editing commands listed above are shown here:

| EDIT | ACTION |
|---|---|
| rxyz | Replaces the current text with xyz starting at the position of r. |
| xyz | Replaces the current text with xyz starting at the position of x. |
| ixyz | Inserts xyz into the current line, starting at the position immediately before the i. |
| ddd | Deletes three characters, one above each d. |
| "d xyz" | Deletes a single character above the d, skips one space, then replaces the current text with xyz starting at the position of x. |
| ddixyz | Deletes two characters, then inserts xyz in the current line in the position before the i. |
| 'd  d' | Deletes one character above the first d, skips two spaces and deletes a second character above the second d. It does not delete a range of characters. |
| 'd  d>xyz' | Deletes a single character above the first d, skips two spaces and deletes to the end of the line beginning at the second d, and then appends xyz to the end of line. |
| >xyz | Appends xyz to the end of the current line. |
| >ddxyz | Deletes the last two characters from the end of the current line and then appends xyz to the end of the line. |
| >rxyz | Replaces the last three characters in the current line with xyz. |
| >ixyz | Appends xyz to the end of the line. In this case, the i command is superfluous, because > accomplishes the same result. Using >xyz would be sufficient. |

| | |
|---|---|
| c/ab/def | Changes all occurrences of ab to def, starting at c. |
| c"ab" | Deletes all occurrences of "ab" starting at c. |
| cxyz | Replace the current text with cxyz, starting at c. Because delimiters have not been specified (as they were in the previous two examples), this is a simple replacement. |

---

EXAMPLES

| | |
|---|---|
| DO pas | Re-executes the the most recent command beginning with the string pas. |
| DO 10 | Re-executes command number 10 (absolute) on the command history stack. |
| DO -2 | Re-executes the second-to-last command on the stack (one command before the most recent). |
| DO -2, c/5a/5b | Change all occurrences of 5a to 5b in the command preceding the most recent one before re-executing it. The default is -1. |
| do ,c/5a/5b | Change all occurrences of "5a" to "5b" in the most recent command before re-executing it. |
| DO run, ">;debug" | Append ;debug to the the most recent :RUN command and then re-execute it. |
| DO 'RUN MYP', '>;LIB=G' | Find the most recent command beginning with RUN MYP and append ;LIB=G before re-executing it. |

**Related Commands:**

REDO
LISTREDO
REDOLOAD
REDOSAVE
REDOSIZE

**Limitations:**

- DO is based on an MPE XL command.

# EXIT

EXIT causes the DUI to terminate. The system prompt will then appear. If any diagnostic programs are running or suspended when the command is received the user will be asked if he wishes to abort them - if the answer is no the EXIT will not be processed; the user will remain in the DUI.

An exception to this will occur if either the QUIET or the UNCONDITIONAL modifiers are used. If either or both of these modifiers are used all diagnostic programs existing under the DUI (suspended, running in background, etc.) will be aborted automatically - the user will not be queried.

**Syntax:**

```
EXIT |QUIET        |
     |UNCONDITIONAL|
```

**Abbreviations and Alternate Tokens:**

```
E
```

**Modifiers:**

```
QUIET           Causes all diagnostics running under the DUI
                to be aborted before the DUI is exited. The
                user is not queried to confirm that a process
                should be aborted.

UNCONDITIONAL   Has the same effect as the QUIET modifier
                except that all programs running under the
                DUI will be aborted even if it means leaving
                the system in an unusual state; normal
                clean up will not necessarily be done. For
                example: devices under test might remain in
                a locked state; tables within the diagnostic
                system might not be updated to reflect the
                actual state of the system.
```

**Command Examples:**

```
DUI> exit

DUI> exit unconditional

DUI> exit quiet

DUI> exit unconditional quiet
```

*naming both modifiers is not an error although only the "unconditional" will really have an effect*

**Limitations:**

- The UNCONDITIONAL modifier may leave the diagnostic system and the I/O system in an unknown state: there will not necessarily be any graceful way to recover from this. The effects of this modifier may not be consistent across different operating systems nor even consistent from one session to another on the same machine.

## FOREGROUND

The `FOREGROUND` command moves a diagnostic which has been running in the background to the foreground. If a program identifier (a process id number) is not specified and only one diagnostic program is running in the background, that program will automatically be placed in the foreground.

If more than one process is running in the background, but the user does not specify which is to be brought to the foreground, the user will be given a list of the processes running in the background from which to choose.



**Syntax:**

`FOREGROUND [`*program-id*`]`

**Abbreviations and Alternate Tokens:**

`FG`

**Modifiers:**

*program-id*    The process identifier of the diagnostic program which is to be brought into the foreground.

**Command Examples:**

```
DUI> fg

DUI> foreground 17
```

**Related Commands:**

`RUN (with BACKGROUND modifier)`

**Limitations:**

- Only one diagnostic program may run in the foreground at any time.
- Once a program is running in the foreground it cannot be placed in the background again: it must run to completion in the foreground.

# HARDCOPY

The **HARDCOPY** command causes all further input to and output from the diagnostic system to be printed by a hardcopy device. Input and output will also continue to appear on the standard output device.

This will continue until **HARDCOPY** is turned off or the diagnostic system is exited.

**Syntax:**

```
HARDCOPY [LDEV [=] logical device name [ENV [=] environment]]
         [PDEV [=] physical path       [ENV [=] environment]]
         [OFF                                               ]
         [ON                                                ]
```



**Abbreviations and Alternate Tokens:**

HC

**Modifiers:**

LDEV *logical device name*

The logical name of the hardcopy device which is to be used.

PDEV *physical path* — The physical path address of the hardcopy device to be used

ENV *environment* — Information which may be used by the hardcopy device to control printing in some way. What this information is and how it must be stated varies from hardcopy device to hardcopy device and from operating system to to operating system. This argument is provided as a convenience for users who wish to control their printing environment and who are knowledgeable about the "environments" used by the chosen hardcopy device.

The diagnostic system will use some printing environment by default if the user does not specify one by using the ENV argument.

OFF — Cease to echo input and output to the hardcopy device.

**Command Examples:**

```
DUI> hc

DUI> hardcopy ldev 23
```

**Related Commands:**

```
OUTFILE
RUN (with OUTFILE modifier)
```

**Limitations:**

- Output may not go to the printer immediately; it may be spooled first.

# HELP

The **HELP** command is used to get information about the diagnostic system. Using the **HELP** command without stating a topic will cause general information about the on-Line diagnostic subsystem to appear. If a topic is specified, information about that topic will appear. HELP messages are available for every diagnostic program and every section and step in every diagnostic. HELP messages also exist for every command listed in this document.

**Syntax:**

```
HELP [HELP                           ]
     [command [SYNTAX]               ]
     [run command modifier  [SYNTAX] ]
     [MNEMONICS                      ]
     [program name                   ]
     [program name SECTIONS          ]
     [program name SC                ]
     [program name SECTION n         ]
     [program name SC n              ]
     [program name (n)               ]
     [LDEV [=] logical device        ]
     [LDEV [=] logical device ACCESS ]
     [LDEV [=] logical device ID     ]
     [PDEV [=] physical path         ]
     [PDEV [=] physical path ACCESS  ]
     [PDEV [=] physical path ID      ]
```

**Abbreviations and Alternate Tokens:**

H
?

**Modifiers:**

omitted — General information about the online diagnostics subsystem will be printed.

HELP — HELP used as a modifier to the HELP command will cause a complete list of the topics about which HELP can give information to be printed.

*command* — Causes a brief description of the purpose of the *command* to appear along with a syntax diagram. A help message exits for every command in this document.

*command* SYNTAX
Causes only the command syntax to appear, not a description of the command.

*run command modifier*

Causes a brief description of the purpose of the *run command modifier* to appear along with a syntax diagram. A help message exits for every run command modifier in this document.

*run command modifier* SYNTAX

Causes only the command syntax to appear, not a description of the run command modifier.

MNEMONICS

Causes a list of all currently recognized mnemonics to be output.

*program-name*

Causes a general message about the diagnostic to appear. Although the specific contents of this message will vary from diagnostic to diagnostic, typically, the message will state what the diagnostic tests and what the diagnostic is capable of doing.

*program-name* SECTIONS
*program-name* SC

Causes a list of all the sections available in the named diagnostic to appear along with a brief description of what each does. A list of mnemonics which may be used in place of section numbers will also appear. Each mnemonic will be annotated with a list of the section numbers it replaces.

*program-name* SECTIONS *n*
*program-name* SC *n*

Causes a detailed description of section *n* of the named diagnostic to appear. If the section is divided into steps, a list of these will appear. If any of the steps may be replaced with mnemonics, a list of those mnemonics annotated with the steps they may replace will also appear.

*program-name* (*n*)

Causes a detailed description of step *n* of the named diagnostic to appear.

LDEV *logical device*
PDEV *physical path*

    Results in a message stating the access mode (exclusive or non-exclusive) and the test mode (destructive or non-destructive) of the device. The user will be told what must be done to get the diagnostic security level necessary to access the device. The product name of the device (e.g., HP1234B) will also be given.

LDEV *logical device* ACCESS
PDEV *physical path* ACCESS

    The access mode (exclusive or non-exclusive) and the test mode (destructive or non-destructive) of the device will appear. A message will also be output stating what the user must do to get the diagnostic security level needed to access the device.

LDEV *logical device* ID
PDEV *physical path* ID

    The product name of the device will appear. This will be some number such as HP1234A.

**Command Examples:**

```
DUI> h

DUI> ? foodiag

DUI> help foodiag sections

DUI> h foodiag sc 5

DUI> h foodiag (155)

DUI> help pdev 4/2.3

DUI> ? mnemonics

DUI> ? h

DUI> ? resume

DUI> h h syntax
```

# INSTALL

The `INSTALL` command is used to add programs to the diagnostic system.

The `INSTALL` command causes the `DUI` to update diagnostic system files with information about the program to be installed. It also places the program's message catalog and the executable program file in their correct places in the diagnostic system. Special files (such as downloadable code files) needed by the program are also taken into the diagnostic system using the `INSTALL` command.

The user is expected to have ready an executable copy of the program and a GENCAT formatted copy of the message catalog. The user is also expected to have ready any special files peculiar to the program. These files may be formatted in any way needed by the program.

Information about the program, which is used each time the program is run, is gathered from the program's message catalog when the `INSTALL` command is given.

The `INSTALL` command is used only to add new programs to the diagnostic system; the user will not be allowed to install (add) a program which already exists in the diagnostic system. Instead, the user may

1. revise information about an already installed program by using the `MODIFY` command, or
2. reinstall a program by first removing it from the diagnostic system using the `PURGE` command, or
3. install the program under a different name.

Information about the diagnostic is gathered from messages in the installation set of the diagnostic's message catalog. The content of these messages is as follows:

```
1 - Program version number in the v.uu.ff format
    (e.g., A.01.09).

2 - Catalog version number in the v.uu.ff format
    (e.g., A.01.09).

3 - Type of program being installed. Options are:

              DIAGNOSTIC

              UTILITY

              EXERCISER

              VERIFIER
```

```
4 - Devices diagnosed by the program.

    ID         [AUTO] [DECODE]

     :            :       :        %

     :            :       :        %

     :            :       :        %

{Where:                                              }
{ ID      = Product number for diagnosed device      }
{           (e.g., HP7978B)                          }
{ AUTO    = If specified, this indicates that the    }
{           program serves as the designated auto-   }
{           diagnostic for this device.              }
{ DECODE  = If specified, this indicates that the    }
{           program serves as the designated hardware }
{           status decoder for this device.          }
```

Please note the percent signs (%) in the above example. These tell the message formatter, GENCAT, that the following line is part of the message. The percent signs **must** appear at the end of every line except the last in message 4.

If any of the above messages contains an invalid value, an error message descriptive of the exact problem will be issued and the installation will be aborted. It will then be up to the user to fix the message catalog and reinstall.

The user will be prompted for four additional pieces of information which are needed to complete the installation process. These are:

- The location (file path) of the diagnostic system into which the installation is to be done. This allows experimental/working diagnostic subsystems to be built and maintained for development and integration purposes. The default will always be the location of the diagnostic system originally issued with the operating system.
- The name of the file containing the executable program. This name must be specified as fully as is necessary to locate it in the file system.
- The names of the files containing the formatted message catalogs. These names must be specified as fully as is necessary to find them in the file system. The message catalog files must be the formatted output of the Native Language Support catalog formatter, GENCAT.

  The user will also be asked to specify the language of each message catalog. The language of the message catalog must be specified in the same language in which the DUI is currently printing messages. For example, if the DUI is currently using English and a German language catalog is being installed, the language of the catalog would be specified as being "German." If the DUI is currently using German, the language of a German language catalog would be specified as "Deutsche."

**Caution** MOST IMPORTANTLY, THE LANGUAGE OF THE FIRST MESSAGE CATALOG INSTALLED FOR EACH DIAGNOSTIC PROGRAM MUST BE THE SAME LANGUAGE THE DUI IS CURRENTLY USING TO PRINT MESSAGES.

- The DUI reads the installation information for a diagnostic program from the installation set of the first message catalog installed for that program. If this information is in a different language than that of the DUI, the DUI will not be able to process it and the installation will fail.
- The names of any additional files used by the program. Such files might, for example, contain downloadable code or environmental scripts used by the program.

The information gathered from the message catalog will be displayed (in the same format as for the LIST command). This assumes that no error occurred while getting the information from the message catalog.

If the installation is successful, a message indicating so will be generated and the user will be returned to the DUI prompt. If the installation fails, messages explaining the exact nature of the problem will be displayed before the DUI prompt reappears.

**Syntax:**

```
INSTALL program-name
```

**Abbreviations and Alternate Tokens:**

```
none
```

**Modifiers:**

*program-name* The name by which the program will be referred when issuing the RUN command.

**Data Prompts:**

```
Specify the file path of the diagnostic system into which the
program will be installed:
FILE PATH [<cr>= cccccccccccc] >
```

*{Where* cccccccccccccc *will be the file path to the group or directory where the "normal" or "default" diagnostic system resides}*

```
Specify name (qualify as necessary) of executable program file:
FILE NAME [<cr>= cancel install]  >


Specify name (qualify as necessary) and language of message
 catalog file:
FILE NAME (current language) [<cr>= cancel install]  >
FILE NAME [<cr>= no more message catalogs]  >
   LANGUAGE [<cr>= cancel this file]  >
```

```
FILE NAME [<cr>= no more message catalogs]  >
   LANGUAGE [<cr>= cancel this file]  >
     :
     :

Specify name (qualify as necessary) of another associated file:
FILE NAME [<cr>= no more files]  >
FILE NAME [<cr>= no more files]  >
    :           :
    :           :
```

**Output:**

The following program is being added to *name of diagnostic subsystem*

| Name | Program Version | Program Type | Catalog Languages | Devices | Associated Files |
|---|---|---|---|---|---|
| *cccccccc* | *c.cc.cc* | *cccc* | *cccccccccc* | *ccccccc* ad* | *cccccccc* |
| | | | | *ccccccc* | *cccccccc* |
| | | | : | : | : |
| | | | : | : | : |
| | | | : | : | : |

* a  program is the auto-diagnostic for this device
   d  program is the hardware status decoder for this device

**Command Examples:**

```
DUI> install foodiag
```

**Related Commands:**

MODIFY
PURGE

**Limitations:**

- The installation will be rejected if the named program already exists in the diagnostic system.

# LIST

The LIST command causes a list of programs installed in the diagnostic system to appear. The modifiers used determine which programs are listed and what additional information is given for each.

**Syntax:**

```
LIST [program-name [[,]... ]] [LONG ]
                              [SHORT]
     [PRODUCT product-name   ] [LONG ]
                              [SHORT]
     [TYPE {DIAGNOSTIC}      ] [LONG ]
           {EXERCISER}         [SHORT]
           {VERIFIER}
           {UTILITY}
```

**Abbreviations and Alternate Tokens:**

L

**Modifiers:**

```
omitted        A list of all programs installed in the diagnostic
               system will be printed.

LONG           The program-name information for all installed
               diagnostic programs is printed.
               (See program-name LONG modifier, below).

SHORT          A list of all programs installed in the diagnostic
               system will be printed.
               (Same effect as giving the LIST command without any
                modifiers).

program-name
program-name SHORT
                The program name is printed.

program-name LONG
               The program name is printed along with its

               * executable program (code) version number
               * message catalog languages
               * type (diagnostic, exerciser, verifier, or utility)
               * auto-diagnostic responsibilities
                 (i.e., is it an auto-diagnostic)
               * decoding responsibilities
                 (i.e., may it be used to decode hardware status)
               * products which the program tests
               * associated files (such as downloadable code files
```

used by the program)

PRODUCT *product-name*
PRODUCT *product-name* SHORT
The programs which run with the specified product are listed.

PRODUCT *product-name* LONG
The programs which run with the specified product are listed along with complete information about each program.
(See *program-name* LONG modifier, above).

TYPE DIAGNOSTIC
TYPE DIAGNOSTIC SHORT
Lists the 'diagnostic' programs.

TYPE DIAGNOSTIC LONG
Lists the 'diagnostic' programs with complete program information.
(See *program-name* LONG modifier, above).

TYPE EXERCISER
TYPE EXERCISER SHORT
Lists the 'exerciser' programs.

TYPE EXERCISER LONG
Lists the 'exerciser' programs with complete program information.
(See *program-name* LONG modifier, above).

TYPE VERIFIER
TYPE VERIFIER SHORT
Lists the 'verifier' programs.

TYPE VERIFIER LONG
Lists the 'verifier' programs with complete program information.
(See *program-name* LONG modifier, above).

TYPE UTILITY
TYPE UTILITY SHORT
Lists the 'utility' programs.

TYPE UTILITY LONG
Lists the 'utility' programs with complete program information.

(See *program-name* LONG modifier, above).

**Command Examples:**

```
DUI> list

DUI> list short

DUI> list type utility

DUI> list xdiag long

DUI> list product HP1234

DUI> list product HP1234 short

DUI> list long
```



**Output:**

{For default or SHORT modifier:}

| *program name* | *program name* | *program name* | *program name* |
|---|---|---|---|
| : | : | : | : |
| : | : | : | : |
| : | : | : | : |

{For complete program name:}

| Name | Program Version | Program Type | Catalog Languages | Devices | Associated Files |
|---|---|---|---|---|---|
| *cccccccc* | *c.cc.cc* | *cccc* | *cccccccccc* | *ccccccc* ad* | *cccccccc* |
| | | | | *ccccccc* | *cccccccc* |
| | | | : | : | : |
| | | | : | : | : |
| | | | : | : | : |

* a  program is the auto-diagnostic for this device
  d  program is the hardware status decoder for this device

# LISTREDO

The `LISTREDO` command displays the user's command history stack. Each command in the display will be numbered - these numbers may be used with the `DO` and `REDO` commands. If a number *n* is given as an argument to the command, only the *n* most recent commands in the history stack will be displayed. Otherwise, the entire history stack will be displayed.

**Syntax:**

`LISTREDO [n]`

**Abbreviations and Alternate Tokens:**

`LR`

**Modifiers:**

*n* The number of commands in the history stack to be displayed. The *n* most recent commands will appear. If *n* is greater than the current size of the user's history stack the entire history stack will be displayed

**Command Examples:**

```
DUI> listredo

DUI> lr 3
```

**Related Commands:**

`DO`
`REDO`

# MODE

The MODE command, used without a modifier, will tell the user the current mode of the operating system; either *single user* or *multi-user*. If one of the arguments is used, the operating system will be placed in the mode specified by the argument - assuming the user has the security capability necessary to place the system into that mode.

**Syntax:**

```
MODE [[=] SINGLE]
     [[=] MULTI ]
```

**Abbreviations and Alternate Tokens:**

none

**Modifiers:**

| | |
|---|---|
| omitted | The current operating system mode will be displayed. |
| SINGLE | Place the operating system into single user mode. If the system is already in single user mode this modifier will have no effect. When an operating system is placed in single-user mode ALL other users are logged off. |
| MULTI | Place the operating system into multi-user mode. If the operating system is already in multi-user mode this modifier will have no effect. |

**Command Examples:**

```
DUI> mode multi

DUI> mode

DUI> mode single
```

**Related Commands:**

SHOWSTATE

**Limitations:**

- The operating system mode will only be changed if the command is given by a user with diagnostic system security level 0.
- Placing an operating system in single-user mode causes all other users to be kicked out of the system - this could have serious, adverse affects especially on the popularity of the kicker.

# MODIFY

The MODIFY command allows a user to change information about a diagnostic program without needing to reinstall the diagnostic. The changes specified will be permanent, but the message catalog, which contains the original information, will not be updated. That is, the original installation information in the message catalog will not be overwritten.

The user will be prompted for the location (file path) of the diagnostic system to be modified. This allows experimental/working subsystems to be built and maintained for development and integration purposes. The default will always be the location of the diagnostic system issued with the operating system.

The user will then be presented with a menu of modification tasks. Any or all of the tasks may be selected in any order. Any task may be selected more than once to correct the previous correction (except, of course, that CANCEL will not undo a previous cancel). Corrections will not actually be made final until the user selects DONE, at which point a list of the program information (same format as the LIST command) will be displayed.

If the changes are successfully finalized, a message indicating so will be generated and the user will be returned to the DUI prompt. If the changes could not be finalized, messages explaining the exact nature of the problem will be displayed before the DUI prompt reappears. The CANCEL task cancels **all** pending corrections that have not already been finalized with the DONE task. If DONE is selected and no corrections are pending, the MODIFY command simply terminates without modifying the program information.

The menu is only displayed once. However, the user may recall the menu by typing **REFRESH** at the MODIFY TASK prompt.

Default values for MODIFY prompts correspond to the pre-existing values for the program being modified.

The CATALOG task allows the user to add to and delete from the list of message catalogs for the specified program. A list of the languages of all the message catalogs associated with the diagnostic program will be displayed. The user will then be prompted to delete from this list. Next the user will be prompted for the locations and names of any message catalog files to be added to the list. The user will also be prompted for the name of the language the message catalog is written in.

The language of the message catalog must be specified in the language that the DUI is currently printing messages in. For example, if the DUI is currently using English and a German language catalog is being installed, the language of the catalog would be specified as being "German." If the DUI is currently using German, the language of a German language catalog would be specified as "Deutsche."

Installation data in the catalogs is **not** examined (as it would be if the INSTALL command were used); therefore, the current information for the program may not correspond to that in the catalogs. All catalog files must be the formatted output of the Native Language Support catalog formatter, GENCAT.

The `CODE` task allows replacement of the code for the specified program. The user will be prompted for the location of the file containing the executable code for the program as well as for the new version number. Note that a code change strictly requires a version change. Likewise, to modify the version number, the user must supply a new program file.

The `DEVICES` task allows the user to add to, delete from, and correct the list of devices which the program diagnoses (if applicable). A list of currently diagnosed devices will be displayed. Then the user will be prompted to add to, delete from, and correct the list. In addition to supplying the device name, the user will be able to specify if the program is the designated auto-diagnostic and/or hardware status decoder for that device. If any other program already serves as the designated auto-diagnostic/status decoder for the device, the user will be prompted to confirm an override of the previous designation. Prompts for additional devices will continue to appear until a simple carriage return (<cr>) is entered to terminate the prompt. When this happens, the user will be prompted for the names of any devices to be removed from the list. The user will be warned if deleting a device from the list of devices diagnosed by the program would result in that device no longer having a designated auto-diagnostic and/or hardware status decoder in the system. As with new devices, the user will continue to be prompted for additional devices until <cr> is entered.

The `TYPE` task allows the user to change the function type of the program. Valid options include `DIAGNOSTIC`, `EXERCISER`, `UTILITY`, and `VERIFIER`. These options may be abbreviated to a minimum number of characters.

The `FILES` task allows the user to add and delete extraneous files needed by the program such as files of downloadable code. A list of the files associated with the program will be displayed. Then the user will be prompted to delete from and add to the list.

The `CANCEL` task effectively destroys all changes made since entering `MODIFY` command.

The `DONE` task exits the `MODIFY` command. If any changes are pending, they are finalized.

The `SHOW` task displays the values for the data items as modified by any pending modifications. This command will produce results identical to the `LIST` command for the program being modified.

**Syntax:**

`MODIFY` *program-name*

**Abbreviations and Alternate Tokens:**

`MOD`

**Modifiers:**

*program-name* The name of the diagnostic program for which information is to be changed.

**Command Examples:**

```
DUI> modify xdiag

DUI> mod ydiag
```

**Data Prompts:**

```
Specify diagnostic system in which program will be modified:
[<cr>= supported system] >

   Modification Options:
          CATALOG  - Add or delete program message catalogs
          CODE     - Replace program code
          DEVICES  - Change list of diagnosed devices
          TYPE     - Change type of program
          FILES    - Add or delete special files

          CANCEL   - Cancel all pending changes and return
                     to modify menu

          DONE     - Implement changes and return to DUI
                     prompt

          EXIT     - Cancel all pending changes and return
                     to DUI prompt

          LIST     - Display pending changes

   MODIFY TASK >


   {for CANCEL task: }

        OK to cancel pending corrections? (YES/NO) [NO]  >


   {for CATALOG task: }

         Catalogs installed for this program are:
              - American
              - English
              - French
              - Canadian-French
              - Kanada
              - Katakana
              - German
```

```
Enter languages of catalogs which should be deleted:
   SYNTAX: language-name (as listed above)

    LANGUAGE [<cr>= no more languages] >
    LANGUAGE [<cr>= no more languages] >
    LANGUAGE [<cr>= no more languages] >
        :          :
        :          :


Need file containing program MESSAGE CATALOG and
  the language the file is written in ...
FILE NAME [<cr>= no more files] >
 LANGUAGE [<cr>= cancel this file] >
FILE NAME [<cr>= no more files] >
 LANGUAGE [<cr>= cancel this file] >
    :
    :
```

```
{for CODE task: }

     Need file containing executable PROGRAM code....
     FILE NAME [<cr>= cancel] >
     PROGRAM VERSION (v.uu.ff) >


{for DEVICES task: }
     Current devices installed for this program:
          - device v  AUTO
          - device x
          - device y  AUTO  DECODE
          - device z        DECODE

     Enter devices NO LONGER SERVICED by this program:
                        SYNTAX : device-name
               DEVICE  [<cr>= no more devices] >
               DEVICE  [<cr>= no more devices] >
               DEVICE  [<cr>= no more devices] >
                  :          :
                  :          :

     Enter NEW devices now serviced by this program
        OR CORRECTIONS to current devices
                        SYNTAX : device-name [Auto] [Decode]
               DEVICE  [<cr>= no more devices] >
               DEVICE  [<cr>= no more devices] >
```

```
                DEVICE  [<cr>= no more devices] >
                   :          :
                   :          :

{for TYPE task: }

     Program Type Options:
                DIAGNOSTIC
                EXERCISER
                UTILITY
                VERIFIER

     Specify program TYPE [cccccccccc] >


{for FILES task: }

     Current files:
                fileA
                fileB
                  :
                  :
                  :

     Enter files to be deleted
        FILE NAME [<cr>= no more files] >
        FILE NAME [<cr>= no more files] >
        FILE NAME [<cr>= no more files] >
               :         :
               :         :

     Enter files to be added
        FILE NAME [<cr>= no more files] >
        FILE NAME [<cr>= no more files] >
        FILE NAME [<cr>= no more files] >
               :         :
               :         :
```

**Output:**

```
    {for DONE task: }

The following MODIFICATION is about to be made to the
On-line Diagnostic Subsystem:


          Program   Program  Catalog                      Associated
  Name    Version   Type     Languages    Devices         Files
========  ========  =======  ===========  ============    ==========
cccccccc   c.cc.cc   cccc    cccccccccc    ccccccc ad*    cccccccc
                                          ccccccc         cccccccc
                                  :           :              :
                                  :           :              :
                                  :           :              :

* a  program is the auto-diagnostic for this device
  d  program is the hardware status decoder for this device


    {for SHOW task: }


          Program   Program  Catalog                      Associated
  Name    Version   Type     Languages    Devices         Files
========  ========  =======  ===========  ============    ==========
cccccccc   c.cc.cc   cccc    cccccccccc    ccccccc ad*    cccccccc
                                          ccccccc         cccccccc
                                  :           :              :
                                  :           :              :
                                  :           :              :

* a  program is the auto-diagnostic for this device
  d  program is the hardware status decoder for this device
```

**Related Commands:**

INSTALL
LIST
PURGE

**Limitations:**

- The installation information in the existing message catalog will not be updated.
- The installation information contained in a new message catalog will be ignored.

# OUTFILE

The `OUTFILE` command causes all diagnostic system input and output to be written into the specified file. This is in addition to having the I/O appear on the user's terminal.

*Outfiles* are opened and processed so that each "line" in the file will consist of 80 characters. That is, each "line" has some combination of 80 characters (including spaces), followed by a carriage return, or by a line feed, or by a carriage return and a line feed (the line termination character(s) is operating system dependent). This file organization would be thought of as "80 byte fixed length record ASCII" on MPE XL.

A user needn't do anything to cause an *outfile* to be organized in this manner. Conversely, a user cannot do anything to force the diagnostic system to write to *outfiles* in any other way.

**Syntax:**

```
OUTFILE [=] {filename}
            {OFF     }
```

**Abbreviations and Alternate Tokens:**

```
OF
```

**Modifiers:**

*filename* The name of the file into which all I/O should be placed.

OFF Stop copying I/O into the file.

**Command Examples:**

```
DUI> outfile yfoo

DUI> of xfoo

DUI> of off
```

**Limitations:**

- I/O will not be copied into a file if the name given is that of a pre-existing file.

# PURGE

The PURGE command causes the named program to be removed from the diagnostic system. The catalog and program files for the named program are deleted from the location of the selected diagnostic system and all references to the named program are removed from diagnostic system files. All special files (such as downloadable code files) associated with the program will also be deleted.

The user will be prompted for the location of the diagnostic system from which the program is to be purged. Information about the program to be purged will be displayed (LIST command format) and the user will be prompted for a verification of the purge. If the diagnostic cannot be purged, a detailed explanation will be provided before returning to the DUI prompt. Otherwise, the user will be immediately returned to the DUI prompt.

**Syntax:**

PURGE *program name*

**Abbreviations and Alternate Tokens:**

none

**Modifiers:**

*program name* The name of the diagnostic program which is to be removed from the specified diagnostic system.

**Data Prompts:**

```
Specify diagnostic system from which program will be purged:
FILE PATH [<cr>= ccccccccccccc] >
```

{*Where* ccccccccccccc *will be the file path to the group or directory where the "normal" or "default" diagnostic system resides*}

```
The following program, along with all of its associated files and
message catalogs, will be removed from the diagnostic system:

          Program  Program  Catalog                     Associated
  Name    Version  Type     Languages    Devices        Files
========  ======== =======  ===========  ============   ==========
cccccccc  c.cc.cc  cccc     cccccccccc   ccccccc ad*    cccccccc
                                         ccccccc        cccccccc
                                  :           :            :
                                  :           :            :
                                  :           :            :

* a  program is the auto-diagnostic for this device
  d  program is the hardware status decoder for this device
```

```
Do you still wish to remove this program? [yes/no]
```

**Command Examples:**

```
DUI> purge xdiag
```

**Related Commands:**

```
INSTALL
MODIFY
```

# REDO

REDO allows the user to edit and re-execute any command in the command line history stack. REDO is interactive. The DUI will display the command so that the user may edit it. After the user edits the command the DUI will display the modified command. This version of the command may also be edited by the user. This will continue until the user enters a carriage return without having done any more editing on the command line displayed by the DUI - the command will then be executed.

The first edit of the command may be specified with *edit-directives* attached to the REDO command. Subsequent edits of the command are done by typing editing directives under the displayed command (see examples).

**Syntax:**

REDO [[CMD=] *command-id*] [, *edit-directives* ]
[;EDIT= *edit-directives*]

**Abbreviations and Alternate Tokens:**

none

**Modifiers:**

*command-id*

Specifies the command to re-execute. The command may be specified by its relative or absolute order in the command line history stack, or by name (as a string) in whole or in part. The default *command-id* is -1, the most recent command. An error is detected if the *command-id* does not exist in the command line history stack.

The command represented by *command-id* will be displayed to the user (in a modified form if the command was given with *edit-directives*). The user may then modify the command again.

Each time the user modifies the command it will be re-displayed in its modified form. Command execution will not occur until the user hits a carriage-return without first having re-edited the command.

| *COMMAND-ID* | EXECUTES |
|---|---|
| (omitted) | Previous command (same as REDO -1) |
| -n | The nth command before the most recent one. N is a number in the command line stack relative to the most recent command. The most recent command is -1. |
| m | Command number m in the command line stack. The number m is absolute. |
| string | The most recent command beginning with this string. |

*edit-directives*

A string specifying changes to be made in the command represented by the *command-id* before it is displayed to the user. When the (edited) command line is displayed, the user may hit carriage-return to execute the command or may edit the command further by using the editing directives. Editing directives are placed under that part of the command string where the user wishes them to take effect.

If no *edit-directives* are given, the command represented by the *command-id* is displayed so the user may edit it as described above.

The *edit-directives* must be surrounded by quotation marks (" ") if they contain any scanner/parser delimiters such as: , ; " ' [ ] = or a space.

---

The editing directives which may be used as *edit-directives* are:

| DIRECTIVE | EFFECT |
|---|---|
| i | INSERT. If text follows the i, the text following i is inserted in the current line at the position after the i. |
| r | REPLACE. If text follows the r, the text following r replaces the same number of characters in the |

current line, beginning at the position of r.

d — DELETE. Deletes a character from the current line for each d specified in the edit line. Note that "d d" does not specify a range but simply deletes one character from the position above each d. Multiple d's may be followed by an insert or replace operation.

d> — DELETE. Deletes to the end of the current line from the position specified by d>. It may be followed by an INSERT or REPLACE operation.

> — APPEND. > followed by text appends the text to the end of the current line. If > is positioned beyond the end of the current line, then a replacement is performed instead.

>d — DELETE. Deletes from the end of the current line, right-to-left. Multiple d's may be specified after >, as well as INSERT and REPLACE strings.

>r — REPLACE. Replaces characters at the end of the command line. The replacement is done so that the last (rightmost) character of the replacement string is at the end of the line.

c — CHANGE. Changes all occurrences of one string to another in the current line when the search string and replace string are properly delimited. A proper delimiter is a non-alphabetic character: ', ", /, etc. The substitution is specified as:

**c<delim> search-string <delim> [replace-string [<delim>]]**

Omitting the replace-string causes occurrences of search-string to be deleted, with no substitution.

u — UNDO. A single u in column one cancels the most recent edit of the current line. Using the UNDO command twice in a row cancels all edits for the current line and re-establishes the original, unedited line. If u is placed anywhere other than column one of the current line, then a simple replacement is performed. UNDO makes sense only if you have a line on which you have performed some editing that can be "undone."

| | |
|---|---|
| other | Simple replacement. Any other character (not i, r, d, d>, >, >d, c, or u) causes that character to be replaced in the current line at t position indicated by the character. In fact, simple replacement also occurs for the editing characters i, r, c, or > if they are not followed by text; or if > appears at or beyond the current end of line. |

---

**Command Examples:**

EDITING SAMPLES

Practical uses of the editing commands listed above are shown here:

| EDIT | ACTION |
|---|---|
| u | First occurrence undoes the previous edits. The u must be in column one. |
| u | Second occurrence undoes all edits on the current line. The u must be in column one. |
| rxyz | Replaces the current text with xyz starting at the position of r. |
| xyz | Replaces the current text with xyz starting at the position of x. |
| ixyz | Inserts xyz into the current line, starting at the position immediately before the i. |
| ddd | Deletes three characters, one above each d. |
| 'd xyz' | Deletes a single character above the d, skips one space, then replaces the current text with xyz starting at the position of x. |
| ddixyz | Deletes two characters, then inserts xyz in the current line in the position before the i. |
| "d  d" | Deletes one character above the first d, skips two spaces and deletes a second character above the second d. It does not delete a range of characters. |
| 'd  d>xyz' | Deletes a single character above the first d, skips two spaces and deletes to the end of the line beginning at the second d, and then appends xyz to the end of line. |

| | |
|---|---|
| >xyz | Appends xyz to the end of the current line. |
| >ddxyz | Deletes the last two characters from the end of the current line and then appends xyz to the end of the line. |
| >rxyz | Replaces the last three characters in the current line with xyz. |
| >ixyz | Appends xyz to the end of the line. In this case, the i command is superfluous, because > accomplishes the same result. Using >xyz would be sufficient. |
| c/ab/def | Changes all occurrences of ab to def, starting at c. |
| c"ab" | Deletes all occurrences of "ab" starting at c. |
| cxyz | Replaces the current text with cxyz, starting at c. Because no delimiters have been specified (as they were in the previous two examples), this is a simple replacement. |

---

EXAMPLES

| | |
|---|---|
| REDO pas | Edits the the most recent command beginning with the string pas. |
| REDO 10 | Edits command number 10 (absolute) on the command history stack. |
| REDO -2 | Edits the second-to-last command on the stack (one command before the most recent). |
| REDO , "c/$null/$STDLIST" | Change all occurrences of $null to $STDLIST in the most recent command before editing it. |
| REDO run, ">;debug" | Append ;debug to the the most recent :RUN command and then edit it. |

**Related Commands:**

DO
LISTREDO
REDOLOAD
REDOSAVE
REDOSIZE

**Limitations:**

- REDO is based on an MPE XL command.

# REDOLOAD

REDOLOAD replaces the user's command history stack with the history stack which was saved by the command REDOSAVE.

**Syntax:**

REDOLOAD [*filename*]

**Abbreviations and Alternate Tokens:**

RDLD

**Modifiers:**

| | |
|---|---|
| *filename* | The name of the history stack which was saved using the REDOSAVE command. If no file is specified, the default saved history file, cmdhist, will be restored. |

**Command Examples:**

```
DUI> redoload

DUI> rdld histfile
```



**Related Commands:**

DO
REDO
REDOSAVE
REDOSIZE

**Limitations:**

- Attempting to restore a file which does not contain a history stack will cause unpredictable results. cmdhist, the default saved history file, will always contain the stack most recently saved - this might have been saved by another user.

# REDOSAVE

REDOSAVE causes all or part of the user's command history stack to be saved into a file. This file may later be restored as the current history stack by using the REDOLOAD command. The command history stack file created by this command is a permanent file and so will continue to exist between operating system sessions.

**Syntax:**

REDOSAVE [FILE [=] *filename*] [*n*]

**Abbreviations and Alternate Tokens:**

RDSV

**Modifiers:**

| | |
|---|---|
| FILE *filename* | The name of the file in which to save the command history stack. If specified, this must be a new file. If not specified, the command history stack will be saved into the default file, cmdhist, whether or not this file already contains a command history stack. |
| *n* | The number of commands in the current history stack to be saved by being written to a file. The *n* most recent commands will be saved. If *n* is greater than the number of commands actually in the current history stack or if *n* is not specified the entire history stack will be saved. |

**Command Examples:**

```
DUI> redosave

DUI> rdsv 3

DUI> rdsv file foohist

DUI> rdsv 18 file fiehist
```

**Related Commands:**

DO
REDO
REDOLOAD
REDOSIZE

**Limitations:**

- If a file is named, it cannot be a pre-existing file. If no file is specified, the history stack will be saved into the default file cmdhist, overwriting the contents of this file.

## REDOSIZE

REDOSIZE allows the user to set the maximum number of commands which will be saved in the user's command history stack. If this command is not used, the history stack will contain a maximum of 25 commands.

**Syntax:**

```
REDOSIZE [=] n
```

**Abbreviations and Alternate Tokens:**

```
RDSZ
```

**Modifiers:**

*n* The number of commands to be held in the user's command history stack. The *n* most recent commands will be retained.

**Command Examples:**

```
DUI> rdsz 15

DUI> redosize 12
```

**Related Commands:**

```
DO
REDO
REDOLOAD
REDOSAVE
```

**Limitations:**

- Setting the command history stack to an arbitrarily large size using this command could cause unpredictable results on some machines. Although no limit is placed on the number of commands which the history stack may hold, it is assumed that the user will limit the size to some "reasonable" number. The maximum "reasonable" size is operating system and machine dependent.

# REPLY

REPLY is used to send a reply to the prompt of a diagnostic program which is running in the background.

If there is only one program running in the background, the *program-id* (a unique process identifier) need not be specified. If more than one program is running in the background and no *program-id* is specified, a list of the background programs waiting for replies will be given. Both the program-id and the programs prompt of every background program waiting for a reply will appear. The user may then repeat the REPLY command.

Not giving a *reply message* has the effect of sending a null string to the program as the reply.

**Syntax:**

REPLY [*program-id*] "*reply message*"

**Abbreviations and Alternate Tokens:**

REP

**Modifiers:**

| | |
|---|---|
| *program-id* | The unique identifying number assigned to a process by the operating system. |
| *reply message* | The message to be sent to the background diagnostic. Could be a carriage return. |

**Command Examples:**

```
DUI> reply yes

DUI> rep 12 no
```

**Related Commands:**

BACKGROUND

**Limitations:**

- The DUI does not check the reply message for validity - the message is sent to the diagnostic program exactly as it is typed.
- If a valid *program-id* is given without a *reply message*, the effect is to send a null string as the reply. A null string is also sent to the program if no *program-id* is given but only one program with a reply pending is running in the background.

# RESUME

RESUME causes the execution of a previously suspended diagnostic program to continue. If the program which is to be resumed is not specified and the user has only one suspended program that one program will be resumed. A list of programs will be printed if more than one program is suspended and the user does not specify a *program-id*. The user should then repeat the command giving one or more of the *program identifiers* from this list.

**Syntax:**

```
RESUME [program identifier [[,]...]] [BACKGROUND]
       [ALL                        ] [BACKGROUND]
```

**Abbreviations and Alternate Tokens:**

RES

**Modifiers:**

| | |
|---|---|
| *program identifier* | The diagnostic's identifier - the unique number assigned to the diagnostic program by the operating system. |
| ALL | Resume running all suspended programs. |
| BACKGROUND | Place the specified programs in the background - if they are not there already - and then resume them. |

**Command Examples:**

```
DUI> resume

DUI> res

DUI> res 24 26 background

DUI> res all background
```

**Limitations:**

- The user may only resume programs which he "owns."
- "Resumed" programs which had been running in the foreground before being suspended will run in the foreground after being resumed. "Resumed" programs which were running in the background when suspended will run in the background when resumed.
- If *i)* more than one *program identifier* is given as an argument to the RESUME command or *ii)* the ALL argument is given but the BACKGROUND argument is not given and *iii)* more than one of the suspended programs to be resumed was running in the foreground when suspended then only the designated foreground process most recently suspended will resume. All other suspended foreground processes will remain suspended.

  All designated background suspended processes will be resumed.

# RUN

The **RUN** command is used to start one or more diagnostic programs. It may be used explicitly or implicitly.

When more than one diagnostic program is named within a run command, all internal setup for each one of the programs will be performed before the DUI calls upon the operating system to launch the programs one after the other. The DUI will not wait for one program to complete before launching the next.

**Syntax:**

[RUN] {*program name* [*command modifier* [...]]} [,...]

**Abbreviations and Alternate Tokens:**

none

**Modifiers:**

*program name* The diagnostic program which is to be run.

*command modifier* The possible modifiers are

BACKGROUND
DEBUG
ERRCOUNT
ERRONLY
ERRPAUSE
ERRPRINT
HARDCOPY
INFILE
LDEV
LOOP
OUTFILE
PDEV
SECTIONS
TRACE
Diagnostic Specific Parameters

Please see the following pages for descriptions of each of these modifiers.

**Command Examples:**

```
DUI> run xdiag

DUI> xdiag

DUI> run xdiag &, ydiag pdev=4.0 sc 3, zdiag ldev 0 bg
```

**Limitations:**

- The number of diagnostic programs which can be launched with one *run* command is limited by the number of processes the operating system will allow a user.
- Only one diagnostic program may run in the foreground at any one time; when running multiple programs all but one must be run in the background.

## RUN COMMAND MODIFIERS

The following pages present detailed functional specifications for the RUN command modifiers.

### BACKGROUND

The BACKGROUND modifier causes the diagnostic program just invoked to be run in the background. Output from a program running in the background will appear on the user's terminal unless it is redirected by the use of the OUTFILE modifier.

**Syntax:**

```
BACKGROUND
```

**Abbreviations and Alternate Tokens:**

```
BG
```

**Default Value:**

```
off
```

**Examples:**

```
DUI> run xdiag &, ydiag bg
DUI> xdiag background outfile xout
```

**Limitations:**

- If the output of the program placed in the background is not redirected, it will appear on the user's terminal. This could cause confusion if more than one program is running in the background or if a program is also running in the foreground.
- When the output of a program running in the background is not redirected, prompts from the program will appear on the user's screen. These prompts CANNOT be replied to directly since there is no interaction possible with a program running in the background. The user must use the *reply* command to send responses to prompts from programs running in the background.

## DEBUG

The DEBUG modifier causes the program to run within a debugger. If the user does not specify which debugger to use, the diagnostic system will invoke a default debugger such as the *nmdebug* debugger on MPE XL or the *xdb* debugger on HP-UX.

Arguments specific to the debugger will not be accepted. This is an exception to the *Support User Interface Standard*.

If the DEBUG modifier has been associated with a diagnostic program by the **set** command, the user may state "debug off" on the run command line in order to run the diagnostic outside of the debugger that one time.

**Syntax:**

```
DEBUG [[=] OFF      ]
      [[=] debugger ]
```

**Abbreviations and Alternate Tokens:**

```
DB
```

**Default Value:**

```
off
```

**Examples:**

```
DUI> xdiag pdev 2/4.3 debug
DUI> run ydiag ldev ydg/0 debug
DUI> run zdiag debug off
```

**Limitations:**

- The DEBUG modifier will, normally, be used only for purposes of diagnostic program development and maintenance. Anyone using this modifier must ensure that the diagnostic program being invoked to run under the debugger has been compiled and linked with whatever options are necessary to get the executable file to run under the chosen debugger. Further, the user must see that any additional files needed to run under the debugger are on the system and in the appropriate places. For example, if the *xdb* debugger is used the source and header files for the program must be on the system.

## ERRCOUNT

ERRCOUNT sets the number of errors to tolerate before aborting the diagnostic program.

**Syntax:**

    ERRCOUNT n

**Abbreviations and Alternate Tokens:**

    EC

**Default Value:**

*infinite - an unlimited number of errors may occur*

**Examples:**

```
DUI> xdiag ldev 4 errcount 12
DUI> run ydiag ldev 2 ec 5
```

**Limitations:**

- The program will be aborted automatically when the specified number of errors have occurred - the user will not have a chance to continue running the diagnostic, even if the modifier ERRPAUSE has also been set.
- If the number of errors to tolerate before terminating is specified (by using the *n* argument), the number MUST be equal to or less than maxint.

## ERRONLY

When ERRONLY is on, only error messages will be printed; most informational messages generated by a diagnostic program will not be displayed.

ERRONLY may be placed "on" by typing ERRONLY or ERRONLY ON.

When ERRONLY is off, all messages will be printed - informational and error.

**Syntax:**

```
ERRONLY [[=] ON ]
        [[=] OFF]
```

**Abbreviations and Alternate Tokens:**

```
EO
```

**Default Value:**

```
off
If ERRONLY is on, the long message form is the default
```

**Examples:**

```
DUI> run xdiag erronly
DUI> run xdiag erronly on
DUI> set ydiag erronly
DUI> run ydiag ldev 3 erronly off
```

## ERRPAUSE

When ERRPAUSE is on and an error occurs, the user will be queried as to whether to continue executing the program. If the user responds yes the program will continue. If the user responds no the program will be aborted.

The modifier ERRPAUSE is equivalent to ERRPAUSE ON.

**Syntax:**

```
ERRPAUSE [[=] ON ]
         [[=] OFF]
```

**Abbreviations and Alternate Tokens:**

```
EPS
```

**Default Value:**

```
off
```

**Examples:**

```
DUI> run xdiag pdev 2/4.2 errpause
DUI> ydiag ldev 4 errpause off
```

**Limitations:**

- If ERRCOUNT has also been set and the maximum number of errors reached, the program will abort without querying the user.

## ERRPRINT

ERRPRINT may be used to control the number of messages printed when an error occurs.

When the LONG modifier is in effect, all error messages associated with an error will be printed - there may be duplicate messages and some of the messages may be obscure. By default, all error messages will be printed when an error occurs - LONG is the online diagnostics subsystem default.

When the SHORT modifier is in effect, only the error message stating what the diagnostic was trying to do, or what the diagnostic believes the problem to be will be printed.

ERRPRINT will always be ON in the online diagnostics subsystem; using the OFF modifier will have no effect.

**Syntax:**

```
ERRPRINT [ON   ]
         [OFF  ]
         [LONG ]
         [SHORT]
```

**Abbreviations and Alternate Tokens:**

```
EPR
```

**Default Value:**

```
on, long
In the online diagnostics subsystem, ERRPRINT cannot be turned off.
```

**Examples:**

```
DUI> run xdiag errprint         has no effect
DUI> run xdiag errprint on      has no effect
DUI> set ydiag epr short
DUI> run ydiag ldev=3 epr off   has no effect
DUI> xdiag errprint long
```

## HARDCOPY

The HARDCOPY modifier causes all of the diagnostic program's input messages and output messages to be printed on a hardcopy device such as a line printer or laser printer.

This modifier does not redirect I/O - it just causes a hardcopy of it to be created. All I/O will also appear on the user's terminal unless a modifier such as INFILE or OUTFILE has also been used.

The user may specify which hardcopy device to use by ldev or pdev but, normally, the diagnostic system will recognize a particular hardcopy device to be used by default.

The user may also specify an "environment" to be used by the hardcopy device. The environment is information which is used by the hardcopy device to control printing in some way. What this information is and how it must be stated varies from hardcopy device to hardcopy device and from operating system to to operating system. This argument is provided as a convenience for users who wish to control their printing environment and who are knowledgeable about the "environments" used by the chosen hardcopy device.

**Syntax:**

```
HARDCOPY [LDEV [=] logical device name [ENV [=] environment]]
         [PDEV [=] physical path       [ENV [=] environment]]
         [OFF                                               ]
         [ON                                                ]
```

**Abbreviations and Alternate Tokens:**

HC

**Default Value:**

off

**Examples:**

```
DUI> ydiag hardcopy
DUI> run xdiag hc ldev 7
DUI> run zdiag hc off
```

### INFILE

The **INFILE** modifier causes all input data expected by a diagnostic program to be read from the specified *infile* rather than from the standard input device.

*Infiles* are opened and processed under the assumption that each "line" in the file will consist of 80 or less characters, followed by a termination character or set of characters. The termination character is operating system dependent. . That is, each "line" has some combination of 80 characters or less (including spaces), followed by a carriage return, or by a line feed, or by a carriage return and a line feed, depending on the operating system. This file organization would be thought of as "80 byte fixed length record ASCII" on MPE XL.

An *infile* with any "line" longer than 80 characters will not be read or processed correctly.

The OFF argument may be used if an *infile* was bound to the diagnostic using the *set* command, but the user does not wish to get input to the program from the file during the current run.

The modifier *INFILE* differs from the command *USEFILE*. A *USEFILE* contains a series of commands (and, possibly, data) and controls the diagnostic session until the end of file is reached. An *INFILE* contains data which a particular diagnostic program would expect the user to give it during the course of its (interactive) execution.

**Syntax:**

```
INFILE [=] {filename}
           {OFF     }
```

**Abbreviations and Alternate Tokens:**

```
IN
```

**Default Value:**

*data is received via the user's terminal rather than a file*

**Examples:**

```
DUI> run wdiag ldev 3 infile winput


The infile winput might contain the following:

       yes
       3
       continue
       yes
       exit

Assuming this is reasonable data for wdiag to receive during
the course of its execution.
```

## LDEV

The LDEV represents the logical name of a device to be tested or used by a diagnostic. This name differs among the various operating systems. For example, on MPE XL systems the LDEV is a number while on HP-UX systems the LDEV is the name of a special device file.

**Syntax:**

LDEV [=] *logical device name*

**Abbreviations and Alternate Tokens:**

```
none
```

**Default Value:**

```
There is no LDEV default value. If a device is needed for a
diagnostic program to run, that device MUST be specified
using either its PDEV or its LDEV.
```

**Examples:**

```
DUI> run xdiag ldev 6
DUI> ydiag sc 5 ldev dsk/c0d0
```

## LOOP

The LOOP modifier specifies the number of times the sections and steps are to be repeated before the diagnostic program terminates. If the modifier is given without a number the sections and steps will be repeated until an interrupt is given.

The OFF argument has the same effect as setting LOOP to 1.

If the ERRPAUSE or ERRCOUNT modifiers are also set, their effect will take precedence over LOOP.

**Syntax:**

```
LOOP [[=] OFF]
     [[=] n  ]
```



**Abbreviations and Alternate Tokens:**

```
none
```

**Default Value:**

```
If LOOP is not set only one iteration of the sections and steps
will be performed before the diagnostic program terminates.
```

**Examples:**

```
DUI> run xdiag ldev 5 sc 4/6 steps 32,46,120/125 loop 6
DUI> xdiag ldev 5 loop
DUI> ydiag pdev 8.4.3 sc 7 loop off
```

**Limitations:**

- If LOOP is given without an argument, the sections and steps will be repeated infinitely. The only way to get out of the infinite loop is to send an interrupt and abort the diagnostic program. The program will terminate abnormally and with unpredictable results.
- If the number of loops is specified (by using the *n* argument), the number MUST be equal to or less than maxint.

## OUTFILE

The `OUTFILE` modifier causes all output from a diagnostic program to be placed into the named file rather than be displayed on the user's terminal.

*Outfiles* are opened and processed so that each "line" in the file will consist of 80 characters. That is, each "line" has some combination of 80 characters (including spaces), followed by a carriage return, or by a line feed, or by a carriage return and a line feed (the line termination character(s) is operating system dependent). This file organization would be thought of as "80 byte fixed length record ASCII" on MPE XL.

A user needn't do anything to cause an *outfile* to be organized in this manner. Conversely, a user cannot do anything to force the diagnostic system to write to *outfiles* in any other way.

The `OFF` argument may be used if an *outfile* was bound to a diagnostic using the *set* command but the user does not wish to have output placed in the file during the current run.

**Syntax:**

```
OUTFILE [=] {filename}
            {OFF      }
```



**Abbreviations and Alternate Tokens:**

```
OUT
```

**Default Value:**

```
Redirection of output does not occur; all output is displayed
on the standard output device which is usually the user's
terminal.
```

**Examples:**

```
DUI> run xdiag outfile fooout
DUI> ydiag out yout
```

**Limitations:**

- The file named cannot already exist.

## PDEV

A PDEV represents the physical path to a device and is composed of numbers corresponding to hardware slot numbers with various sorts of punctuation separating the numbers. The punctuation used may be machine dependent.

**Syntax:**

PDEV [=] *physical path*

**Abbreviations and Alternate Tokens:**

none

**Default Value:**

```
There is no PDEV default value. If a device is needed for a
diagnostic program to run, that device MUST be specified
using either its PDEV or its LDEV.
```

**Examples:**

```
DUI> run xdiag pdev 4/2.3 sc 4
DUI> run ydiag pdev 8.1
```

## SECTIONS

A SECTION is a major operation or set of related operations within a diagnostic which can be explicitly invoked by a user. Some or all of a diagnostic's SECTIONS may be designated by the diagnostic developer to be default SECTIONS which are to be run if the user does not explicitly invoke one or more SECTIONS. A SECTION which performs more than one operation will have a subset of those operations, the steps, designated as the defaults to be run when the section is named but none of its steps are specified. When the SECTION is invoked without explicitly stating which steps are to be run the "default" steps will be run automatically.

SECTIONS and steps are numbered although the mnemonics listed below may be used instead of explicit numbers. When SECTIONS are invoked they are run in numerical order.

Steps are also run in numerical order. Steps may be explicitly named, using numbers or mnemonics, only if their associated SECTIONS are also named.

The security capability of the user determines whether or not an operation represented by a section or step is actually performed. For example, a user would need a high security capability before the diagnostic system would allow him to perform an operation which could cause the loss of user data on a device.

If a SECTION includes more than one separately callable operation (step) the needed capability is determined and checked for each individual operation. If a user tries to run a section or step for which he lacks the appropriate security capability a message will be output stating which security capability is needed and what security capability the user has.

Any sections and steps which the user invoked for which he does have the appropriate security capability will be run. Any which require a higher user security capability will not be.

**Syntax:**

```
SECTIONS { | [=] [+]n[[,]...]  | } [(|[+]n[[,]...]  |)]
             [-]                      [-]
           | [=] [+]n/n[[,]...]|    |[+]n/n[[,]...]|
             [-]                     [-]
```

*Please note that steps are designated using either parenthesis () or square brackets []. The square brackets are not shown in the above syntax diagram.*

*The "+" and "-" before section and step numbers are used when the user has already set up a list of sections and steps to run using the* **set** *command and when the user wishes to run "default" sections and or steps with certain exceptions. The "+" and "-" refer to additions to and subtractions from a list of default sections and steps. Using the "+" will cause the immediately following sections or steps to be executed IN ADDITION TO the previously set sections or steps. Using the "-" will cause previously set sections and steps to be executed with the EXCEPTION of those whose numbers are preceded by the "-".*

*In any case, any number or number range preceded by the "+" will be run - security permitting - while any number or number range preceded by the "-" will not be run. The "+" and "-" may also be used with the* `sections` *modifier in the* `set` *command to change the default sections and steps to be run when a diagnostic is invoked without needing to re-specify the entire list of wanted sections and steps.*

**Abbreviations and Alternate Tokens:**

`sc`

**Default Value:**

The sections and steps which the diagnostic writer named as the defaults for the invoked diagnostic program. This differs for every diagnostic program.

**Examples:**

```
DUI> run xdiag pdev 2/4 sections 5/6,7 (-9,20/50)
```

*Note: in this example, step 9 will not be executed but steps 20 through 50 will be executed.*

```
DUI> ydiag ldev 3 sc 4,6,+9 [10,+12]
```

**Mnemonics:**

The following may be used instead of section and/or step numbers. However, the section or step numbers which will be executed when one of these mnemonics is used varies from diagnostic to diagnostic. Not all diagnostics will have sections and steps associated with these mnemonics.

Each diagnostic writer determines which sections and steps will be run when one of these mnemonics is given in conjunction with a particular diagnostic.

For a list of the sections and/or steps which will be run when one of these is used type

**help** *program name* **sections**.

*These may be substituted for section numbers and step numbers*

```
all
auto
clear
default
errlog
fast
hwstatus
id
loopback
non-destructive
selftest
interactive
```

**Mnemonic Descriptions:**

*Please note that the actual affect of any of these mnemonics may be altered if the capability of the user is too low to allow some operation to be performed. Although the descriptions speak about sections the mnemonics may also be used to substitute for steps.*

**ALL:** Run all sections, default and non-default alike.

**AUTO:** Run all sections which are designated as autodiagnosable. That is, run all sections which might be run automatically by the operating system when it detects a possible hardware defect.

**CLEAR:** Run whatever section or sections "clear" or "reset" the device being diagnosed.

**DEFAULT:** Run the default sections. This mnemonic will be useful when non-default sections have been specified at a more global scope and the user wishes to only run defaults locally.

**ERRLOG:** Run whatever sections read and decode error logs.

**FAST:** Run the sections which the diagnostic developer has designated as "fast." That is, run those sections which will quickly test a large part of the device.

**HWSTATUS:** Run the sections which read and decode the hardware status of a device.

**ID:** Run the sections which "identify" the device.

**LOOPBACK:** Run whichever sections perform a loopback to the device(s). The type of loopback(s) performed will vary from diagnostic to diagnostic - the diagnostic developer determines which loopback(s) to perform when this mnemonic is given.

**NON-DESTRUCTIVE:** Run only non-destructive sections or steps. This will be useful to users possessing the highest security level who wish to avoid inadvertently running any potentially destructive sections.

**SELFTEST:** Run the sections which perform selftests on the device(s).

**INTERACTIVE:** Run the diagnostic's interactive sections.

**Examples of Mnemonic Use:**

*Please note that a user is not expected to know what section or step numbers a mnemonic replaces - the user may name a section or step number and the mnemonic that replaces it on the same line. This is not an error.*

*If there is a conflict among statements on the same line the rightmost statement will take precedence. Nested statements are not allowed.*

**ALL:**

```
sc all

sc all(all)

sc all(+all)
```

*Please note that the "+" and "-" operators may be used with any mnemonic even if, as in this case, they have no effect.*

**AUTO:**

```
sc auto

sc +auto
```

*Run previously named sections AND autodiagnosable sections.*

```
sc 5/7(auto)
```

*Only run those steps of sections 5 through 7 which are autodiagnosable.*

```
sc 8(-auto)
```

*Do not run the autodiagnosable steps of section 8. Do run the default steps* *of section 8 which are not autodiagnosable.*

**CLEAR:**

`sc clear`

`sc clear(4, 10/12)`

*Run steps 4 and 10 through 12 of whatever sections clear the device. If none of the sections which clear the device have any steps 4, 10, 11, or 12 nothing will be run and a message will be output to the user stating this.*

**DEFAULT:**

`sc 9/24(1/300)`

*Run steps 1 through 300 of sections 9 through 24.*

`sc 14(default)`

*Only run the default steps of section 14. These will be either the diagnostic system default steps or the steps which the user has designated to be defaults for section 14 using the* `set` *command. The same effect would be achieved by using* `sc 14`.

`sc 10(+default)`

*Run the default steps of section 10. The "+" has no effect here but is not an error.*

`sc default`

*Run all default steps of all default sections. The same effect can be achieved by invoking the diagnostic on a run command line without mentioning sections.*

**ERRLOG:**

`sc errlog`

```
sc auto,errlog
```

*Run the default steps of both the autodiagnosable and errlog sections.*

**FAST:**

```
sc fast

sc 2/3,fast,6,10(-auto)
```

*Run the default steps of sections 2, 3, and 6 and the default steps of all "fast" sections. Run the default steps of section 10 EXCEPT for those default steps which are also autodiagnosable. If section 10 is a "fast" section still only run its non-autodiagnosable default steps.*

**HWSTATUS:**

```
sc hwstatus

sc auto, hwstatus(2/6)
```

*Run the default steps of all autodiagnosible sections AND steps 2 through 6 of any hardware status sections. If any hardware status sections are autodiagnosible their default steps will be run.*

**ID:**

```
sc id

sc id(-auto)
```

*Run the non-autodiagnosable steps only of the "identify" sections. If the "identify" sections do not have any autodiagnosable steps the directive is ignored. If the "identify" sections are composed completely of autodiagnosable steps or are themselves autodiagnosable sections no operations will be performed.*

**LOOPBACK:**

```
sc loopback

sc 3(-auto,+default), loopback(4/3,+auto)
```

*If section 3 is a loopback section its autodiagnosable steps will be run since the "+auto" directive appears to the right of the "-auto."*

**NON-DESTRUCTIVE:**

```
sc id

sc id(non-destructive)
```

*Run the non-destructive steps only of the "identify" sections. If the "identify" sections are not divided into steps only run the sections if they are non-destructive.*

```
sc non-destructive,5/8,3(+auto)
```

*Although sections and steps will be run in numerical order the diagnostic user interface will accept them in any order. When section 3 is run in this example any steps previously set for it at a global level will be run along with all autodiagnostic steps. If no special steps have previously been set for it, its default and autodiagnostic steps will be run.*

**SELFTEST:**

```
sc selftest

sc selftest(non-destructive), fast
```

**INTERACTIVE:**

```
sc interactive

sc 4/6(-auto), interactive,fast
```

*The user is not expected to know which* *numbers correspond to each mnemonic so any of sections 4, 5, or 6 could be "interactive" and/or "fast" sections. Duplication is acceptable. If any of sections 4, 5, or 6 is either "interactive" or "fast" its autodiagnosable steps will be run if they are also default steps since the "interactive" and "fast" directives appear to the right of "-auto."*

## TRACE

The TRACE modifier enables/disables software tracing messages. The entering and exiting of procedures and functions are noted by the display of messages such as

```
Entering fee
  Entering fie
    Entering foe
      Entering fum
      Exiting fum
    Exiting foe
  Exiting fie
Exiting fee
```

Selective tracing may be done by using arguments corresponding to various types of software modules: diagnostic programs (PROG), diagnostic library routines (LIB), device access routines (DAR), and I/O system modules such as device drivers (SYS). Tracing of error conditions will occur if the ERROR modifier is used. Any of these may be used in combination.

Please note that the SYS argument will not result in trace messages on most systems. The "hooks" needed to make this work are only being placed in the appropriate code on a few systems.

**Syntax:**

```
TRACE [=] { OFF   }
          { ALL   }
          { PROG  }
          { LIB   }
          { DAR   }
          { SYS   }
          { ERROR }
```

**Abbreviations and Alternate Tokens:**

TR

**Default Value:**

off

**Examples:**

```
DUI> xdiag ldev 4 sc auto trace prog
```

```
DUI> ydiag ldev 2 tr off
```

*Trace was probably "bound" to ydiag using the* set *command. This turns tracing off during this one run of* ydiag.

```
DUI> run wdiag tr prog dar
```

**Limitations:**

- TRACE can only be effective when the code modules being traced are properly instrumented. The TRACE modifier cannot display progress through code which has not had the proper calls inserted during development.

## Diagnostic Specific Parameters

Parameters, modifiers, or other information unique to a particular diagnostic program may, in some cases, be specified within the `run` command. Such information is placed within double quotes ("").

The information within the double quotes is passed directly to the invoked program - no checking is done by the DUI to determine the correctness or validity of the information being passed. Many diagnostic programs prompt for any specific information they need. This mechanism for passing information to a diagnostic is provided as a convenience to the diagnostics but is not used by all of them.

The `set` command may also be used to bind program specific parameters to a particular diagnostic. Every time that diagnostic is invoked the "set" information will be passed directly to the diagnostic.

Please see the diagnostic's manual regarding what information a particular diagnostic expects to receive in this manner.

**Syntax:**

"*information*"

**Abbreviations and Alternate Tokens:**

`none`

**Default Value:**

`null`

**Examples:**

`DUI> run xdiag pdev 4.3.2 sc 7 (2/6,10)` "*some information*"
`DUI> set ydiag` "*some information*"

**Limitations:**

- No checking is done before the information is passed to the diagnostic; the information is passed exactly as given.
- The information must be eighty (80) characters or less in length.

# SETVAR

The **SETVAR** command allows the user to explicitly set the values of environmental variables which control various features of the user interface. The **SETVAR** command may also be used to examine the current values of these variables. The only environmental variable currently associated with the **SETVAR** command within the **DUI** is **TRACE**.

If **SETVAR** is given without a variable being specified, a list of all **DUI** environmental variables along with their current values will be displayed.

The variables modified by the **SETVAR** command affect the Diagnostic User Interface itself - NOT the diagnostic programs run from it.

**Syntax:**

SETVAR [*variable* [=] *value* [[,]...]]

**Abbreviations and Alternative Tokens:**

none

**Modifiers:**

| *variable* | *value* |
|---|---|
| TRACE | OFF |
| | ALL |
| | PROG |
| | LIB |
| | SYS |
| | ERROR |

```
The TRACE variable, used with the PROG modifier,
results in the display of software tracing messages.
The entering and exiting of the DUI's procedures and functions
are noted by the display of messages such as

        Entering fee
          Entering fie
            Entering foe
              Entering fum
              Exiting fum
            Exiting foe
          Exiting fie
        Exiting fee

Tracing of error conditions will occur if the ERROR modifier
is used. The errors which will be reported are diagnostic system
internal errors. Some of these "errors" are expected conditions
which are encountered during normal processing. The error messages
which will be displayed as a result of TRACE ERROR are written
for factory personnel troubleshooting the diagnostic system itself
```

- they are not written
for the end user and may be meaningless to many users.

Errors which would normally be reported to a user will continue to be reported whether TRACE ERROR is set or not.

PROG and ERROR may be used in combination.

Tracing will continue until the SETVAR TRACE OFF command is given or the diagnostic session is ended.

**Command Examples:**

```
DUI> setvar

DUI> setvar trace error

DUI> setvar trace prog

DUI> setvar trace error prog

DUI> setvar trace off
```

**Limitations:**

- The ALL, LIB, and SYS values of the SETVAR variable TRACE have no effect within the online diagnostics subsystem.

# SET

The SET command allows the user to explicitly reset the system default values for *modifiers* which may be given on a run command line. Once set, these values are used for every run of every diagnostic during a diagnostic session unless the modifier values for an individual diagnostic are changed. These *modifier* values become the global defaults for the diagnostic session.

The SET command also lets the user bind particular command modifier values to individual diagnostic programs. Modifier values associated with an individual diagnostic through the use of the SET command become the default values for that diagnostic until the end of the diagnostic session.

Individual modifier values may be changed, temporarily, by naming those modifiers and their new values on the run command line of a diagnostic. At the conclusion of that one diagnostic run the modifier values will revert to their defaults.

If the SET command is given with a *program name* but no modifiers, a list of the current values of all possible modifiers to that *program* will be displayed.

If the SET command is given with neither *modifiers* nor a *program name*, the current global default value of each *modifier* will be displayed.

**Syntax:**

SET {[*program name*] *command modifier* [[,]...]}

**Modifiers:**

*program name* — The diagnostic program whose modifier default values are to be set or displayed.

*command modifier* — The modifier whose value is to be set at the global level. Any of the run command modifiers may be set in this way except the SECURITY modifier. The following is a complete list of the command modifiers which may be used with this command.

```
BACKGROUND
DEBUG
ERRCOUNT
ERRONLY
ERRPAUSE
HARDCOPY
INFILE
LDEV
LOOP
OUTFILE
PDEV
SECTIONS
TRACE
Diagnostic Specific Parameters
```

Please see the descriptions of each of these modifiers for an indication of values which may legitimately be assigned.

**Command Examples:**

```
DUI> set xdiag loop 5 errpause

DUI> set loop 20 errcount 4

DUI> set loop

DUI> set xdiag

DUI> set
```

**Related Commands:**

SHOWPARMS

# SHOWACTIVE

The **SHOWACTIVE** command causes a list of all current diagnostic processes to be displayed along with their process identifier numbers. Each displayed diagnostic process is also noted as being "running," "suspended," or "aborting" and either "foreground" or "background."

A "running" program is one that is executing normally.

A "suspended" program is one which was "suspended" by use of the **SUSPEND** command and is waiting for a **RESUME** command to return to a running state.

An "aborting" program is one which is in the transient state occurring between the receipt of an **ABORT** command and the actual termination of the program.

"Foreground" and "background" show where the process is. Although every program will be designated as one or the other, the designation is most important for suspended programs. The behavior of the RESUME command may be affected by where a program was running when it was suspended.

**Syntax:**

```
SHOWACTIVE
```

**Abbreviations and Alternate Tokens:**

```
SA
```



**Command Examples:**

```
DUI> showactive

DUI> sa
```

# SHOWDEFAULT

SHOWDEFAULT will cause a list of command modifiers and their default values to be displayed. The values displayed will be those the modifiers held when the diagnostic system was invoked; i.e., the initial diagnostic system default values for user settable modifiers will be displayed.

If SHOWDEFAULT is given with a list of command modifiers, the diagnostic system default values for those modifiers will be displayed. If SHOWDEFAULT is given without a list of modifiers or with the ALL argument, all user settable modifiers will be displayed along with their default values.

**Syntax:**

```
SHOWDEFAULT [ALL                             ]
            [command modifier [[,]...]]
```

**Abbreviations and Alternate Tokens:**

SD

**Modifiers:**

| | |
|---|---|
| ALL | List all user settable modifiers and their default values. |
| *command modifier* | A user settable diagnostic modifier. The following is a complete list of these modifiers. Any valid run command modifier except SECURITY and the "Diagnostic Specific Parameters" modifier may be used with the SHOWDEFAULT command. |

```
BACKGROUND
DEBUG
ERRCOUNT
ERRONLY
ERRPAUSE
HARDCOPY
INFILE
LDEV
LOOP
OUTFILE
PDEV
SECTIONS
TRACE
```

**Command Examples:**

```
DUI> showdefault

DUI> sd all

DUI> sd loop errpause errcount
```

**Related Commands:**

```
SET
SHOWPARMS
```

## SHOWPARMS

SHOWPARMS will cause a list of user settable modifiers and their current values to be displayed.

The arguments the user gives to this command will determine which set of values for the modifiers will be displayed. If no argument is given the effect is the same as using the ALL argument.

If no argument is given or if the ALL argument is given the global default modifier values will be displayed along with the values the modifiers have for individual diagnostic programs (the values set with the SET command).

**Syntax:**

```
SHOWPARMS [GLOBAL                  ]
          [LOCAL                   ]
          [ALL                     ]
          [program name [[,]... ]]
```

**Abbreviations and Alternate Tokens:**

SP

**Modifiers:**

GLOBAL — Display all modifiers with the values they hold globally. Each of these values will be either the diagnostic system default value for the modifier or the value for the modifier which has been explicitly named using the SET command.

LOCAL — Display only those modifiers and values explicitly set for individual diagnostic programs.

ALL — Display all modifiers with the values they hold globally. Also display those modifiers and values which have been reset for individual diagnostic programs using the SET command.

*program name* — Display all modifiers with the values they hold for the named program(s). These values may be the original diagnostic system default values, the global values set with the SET command, or values set explicitly for the named program using the SET command. In any case, the values displayed will be those which will be used when the named program is invoked unless the values are temporarily reset on the run command line.

**Command Examples:**

```
DUI> showparms

DUI> sp xdiag ydiag

DUI> sp global

DUI> showparms local

DUI> sp all
```

**Related Commands:**

SET
SHOWDEFAULT

# SHOWSTATE

The `SHOWSTATE` command will cause the current system mode (single-user or multi-user) and the user's security capability to be displayed.

**Syntax:**

```
SHOWSTATE
```

**Abbreviations and Alternate Tokens:**

```
SS
```

**Command Examples:**

```
DUI> showstate

DUI> ss
```

**Related Commands:**

```
MODE
```

# SUSPEND

The SUSPEND command causes the execution of a diagnostic program to stop. The program will remain "frozen" until a RESUME command is received at which time program execution will resume.

If the SUSPEND command is given without an argument and only one program is running, that one program will be suspended. If more than one program is running and the SUSPEND command is given without an argument, a list of programs and their program identifier numbers (*program-id*) will be displayed. The user may then repeat the SUSPEND command giving one or more of these *program-ids* as arguments.

**Syntax:**

```
SUSPEND [ALL                  ]
        [program id [[,]... ]]
```

**Abbreviations and Alternate Tokens:**

SUS

**Modifiers:**

| | |
|---|---|
| ALL | Suspend every running program which was invoked by the user. |
| *program id* | The unique number identifying a particular run of a particular program. Used to tell the diagnostic system which program(s) to suspend. |

**Command Examples:**

```
DUI> suspend

DUI> sus all

DUI> sus 23 17
```

**Related Commands:**

RESUME

# UNLOCK

The UNLOCK command is used to explicitly unlock a device. This will release the device back to general access. UNLOCK may be used to release a device back to the system for general usage after the device has been fixed (or replaced) if it had previously been locked because it was defective.

If no LDEV or PDEV is given with the UNLOCK command a list of locked devices will be displayed. The devices listed will be malfunction locked.

**Syntax:**

```
UNLOCK [LDEV [=] logical device name ]
       [PDEV [=] physical path       ]
```

**Modifiers:**

| | |
|---|---|
| omitted | A list of malfunction locked devices will be displayed. |
| LDEV *logical device name* | The logical name of the device. |
| PDEV *physical path* | The physical path to the device. |

**Command Examples:**

```
DUI> unlock pdev 4/2.3

DUI> unlock ldev 12
```

# USEFILE

The USEFILE command causes input to the diagnostic system to be gotten from the specified file rather than from a user's terminal. Reading from the file begins immediately; the file controls the diagnostic session until end of file is reached or the *usefile* is prematurely closed because an interrupt was received.

*Usefiles* are opened and processed under the assumption that each "line" in the file will consist of 80 or less characters, followed by a termination character or set of characters. The termination character is operating system dependent. . That is, each "line" has some combination of 80 characters or less (including spaces), followed by a carriage return, or by a line feed, or by a carriage return and a line feed, depending on the operating system. This file organization would be thought of as "80 byte fixed length record ASCII" on MPE XL.

An *usefile* with any "line" longer than 80 characters will not be read or processed correctly.

*Usefiles* may be nested; that is, a *usefile* may contain the USEFILE command. All open *usefiles* will be closed when an interrupt is received.

**Syntax:**

USEFILE [=] *filename*

**Abbreviations and Alternate Tokens:**

USE

**Modifiers:**

*filename* The file from which the diagnostic system should get its input.

**Command Examples:**

```
DUI> usefile foocmds

DUI> use cmdfile
```

The usefile *foocmds* might contain the following:

```
run xdiag pdev 4.0.12 sc 3/10
run ydiag ldev 3 infile foo outfile fum
list type utility
mode single
run wdiag ldev 0 sc 5 outfile fee
```

**Limitations:**

- The USEFILE command can only be run in the foreground.
- It is assumed that the file contains commands and input sensible to the diagnostic system. If not, errors may occur.

## Error Messages

The following are the error messages generated by the DUI.

**305** `*** COULD NOT READ MESSAGE AT DUI's PORT (DUIERR 305)`

CAUSE: Input, such as a command, was given to the DUI at a user's terminal. The input was queued to the DUI's message port. The attempt to pull the message off of the port so that it could be processed failed.

ACTION: Try again. If the failure occurs a second time get out of the diagnostic system any way possible. It may be necessary to log into another terminal and abort the DUI process - the EXIT command to the DUI is unlikely to work in the present case. Submit an SR against the DUI; give as much information as possible about the circumstances surrounding the error (was the command the first given during the diagnostic session or had previous input been processed correctly, had any unusual messages been printed when the DUI was invoked, etc.).

---

**308** `*** RECEIVED UNEXPECTED MESSAGE AT DUI'S PORT (DUIERR 308)`

CAUSE: Either a program sent an unrecognized request (something other than enable_intr_notify, disable_intr_notify, suspend_prog, or req_user_info) or the message type of a message pulled off of the DUI's port was not one the DUI could recognize.

ACTION: Use the command SETVAR TRACE ERROR. Repeat the command which resulted in the error the first time and note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors. If the command used was RUN, it is possible that the error is in the diagnostic being run rather than in the DUI. However, if in doubt, submit the SR against the DUI rather than against the diagnostic.

---

**309** `*** COULD NOT READ FROM SET# ! MSG# ! (DUIERR 309)`

CAUSE: The attempt to retrieve a message from a diagnostic's message catalog failed. The most likely reason is that the message was never put into the catalog.

ACTION: Submit an SR against the diagnostic being run.

---

**311** `*** UNRECOGNIZED PROCESS - NOT IN LIST OF CURRENT PROCESSES (DUIERR 311)`

CAUSE

The DUI was either processing an output_data request, a program request, or a user reply to a program but could not find the process identifier (pid) in its (the DUI's) table of known processes. The pid is needed to process the request because, without it, the DUI has no way of knowing which catalog to pull the message to be printed from. (The DUI uses the pid as an "index" into a table which contains information about the process - including the file descriptor of the open message catalog belonging to the running program).

It is also possible that the pid was in the DUI's table, but the corresponding message catalog file descriptor was 0.

The pid is also needed when a "program reply" is processed. The reply received was a handshake to an interrupt received message which the DUI sent the program. If the pid associated with the handshaking cannot be found in the DUI's table the DUI cannot reset the "interrupt sent" flag for the process. Until this flag is reset, no more interrupts will be sent. (A moot point, since without the pid the DUI cannot know there is a process needing interrupt notification).

Lastly, the DUI will print this message when it has received a program request to enable_intr_notify, disable_intr_notify, suspend_prog, or request_user_info and cannot find the pid associated with the request (in the ipc message) in its table. Without the pid the DUI cannot process the request since the information needed to process the request is in the table.

The most likely reason for any of these situations to occur is that a timing problem in the underlying operating system caused the request or reply to be received by the DUI long after the DUI was informed that the diagnostic terminated. When the DUI is informed that a diagnostic is to terminate it always searches for and processes any messages associated with the diagnostic before processing the termination.

ACTION

If the problem is reproducible, submit an SR against the DUI giving as much information as possible about what seems to be happening when the error occurs. If the problem ever occurs it is likely to be transient and difficult to reproduce - the SR should explain in as much detail as possible the context in which it happens.

**312** `*** COULD NOT CREATE A PORT FOR THE DUI (DUIERR 312)`

CAUSE
Call to build_port failed. Build_port returned an error status. (i.e. some status other than successful or dipc_dup_pname). If dipc_dup_pname had been returned, the DUI would just try again to build a port using a different port name.

ACTION
There is something seriously wrong with either the inter-process communication system or with the diagnostic monitor. If the user can kill the monitor and its associated processes and remove the diagnostic ports and then restart the monitor the problem might go away. However, in most cases this will not be possible. Since the normal tracing and printing mechanisms have not been set-up this early in the code very little additional information can be gotten. Please log an SR against the DUI giving as much information as possible about what is happening. Include the operating system build version, whether the system had been recently rebooted or not, whether the diagnostic system had been run successfully before the problem appeared, etc.

---

**319** `*** COULD NOT DO INITIALIZATION NEEDED FOR PRINTING (DUIERR 319)`

CAUSE
Two possible causes.

1) could not form the file path for the DUI's message catalog

2) a message catalog for the DUI written in the system default language could not be found and the DUI was unable to determine if it was all right with the user to just use English language messages. (get_user_input returned with an unsuccessful status of some sort).

ACTION
Check to see if a message catalog for the DUI is on the system. The catalog will be named CDUIFXXX where XXX is three digits corresponding to native language localization language codes. The default catalog is CDUIF000. If the catalog cannot be found or is not in the correct directory, have the system administrator put the file on the system with the correct access permissions. If the catalog is on the system, file an SR against the DUI.

---

**320** `*** COULD NOT PRINT - PRINTING INSTRUCTIONS INCOMPLETE (DUIERR 320)`

CAUSE
Neither a "print literal" nor a "print message from set" was specified when print_dui_msg was called. The DUI is the only code which uses this procedure.

ACTION
Log an SR against the DUI giving a description of what was being done when the error was printed and a list of any messages which immediately preceded this one.

---

| | |
|---|---|
| **321** | ***** COMMAND RECOGNIZED BY PARSING ROUTINES BUT NOT BY PROCESSING ROUTINES (DUIERR 321)** |
| CAUSE | The command the user gave was in the table of command names recognized by the DUI's parser. However, it was not in the list of commands the DUI recognizes as being able to process. |
| ACTION | Log an SR against the DUI listing this message and the command which was not recognized. |

| | |
|---|---|
| **322** | ***** INVALID LDEV SPECIFIED (DUIERR 322)** |
| CAUSE | The ldev the user gave with the run command could not be converted into a pdev (which is actually used by the diagnostic system). The call to obtain_pdev returned a status of dac_invalid_ldev. |
| ACTION | Determine the correct ldev (sysmap may be able to help if it is on the system) and run the diagnostic again or give the pdev instead of the ldev with the run command. |

| | |
|---|---|
| **323** | ***** COULD NOT CONVERT THE SPECIFIED LDEV INTO A PDEV (DUIERR 323)** |
| CAUSE | The ldev the user gave with the run command could not be converted into a pdev (which is actually used by the diagnostic system). The call to obtain_pdev returned a non-successful status (some status other than dac_invalid_ldev or successful). |
| ACTION | Determine the correct ldev (sysmap may be able to help if it is on the system) and run the diagnostic again or give the pdev instead of the ldev with the run command. |

| | |
|---|---|
| **324** | ***** FAILED TO SEND IPC MESSAGE TO A PORT (DUIERR 324)** |
| CAUSE | Send_to_port failed (returned status something other than successful). The DUI was trying to send either one of the three program initialization messages or a user_interrupt notification to a diagnostic but failed for some unknown reason. |
| ACTION | Re-run the diagnostic. If the failure occurs again, use the SETVAR TRACE ERROR command. Repeat the RUN command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors. |

325 *** COULD NOT LAUNCH THE DIAGNOSTIC (DUIERR 325)

CAUSE The diagnostic system service launch_process failed to launch a diagnostic. The message printed just prior to this one should give some indication why (the message will be the one associated with the error status returned by launch_process).

ACTION Whatever action is associated with the message printed immediately before this one.

---

326 *** UNRECOGNIZED IPC PROGRAM FUNCTION (DUIERR 326)

CAUSE The DUI's procedure send_msg_to_program was asked to process a request other than prog_info1, prog_info2, prog_info3, or user_interrupt. Therefore, the DUI did not recognize the request.

ACTION Re-run the diagnostic. If the failure occurs again, use the SETVAR TRACE ERROR command. Repeat the RUN command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

327 *** CANNOT RUN THE DIAGNOSTIC (DUIERR 327)

CAUSE The DUI tried to split the file path for the diagnostic - just in case the run command included the fully qualified name of the diagnostic (the file name along with the complete path to it in the file system). This needs to be done before the DUI checks the list of installed diagnostics to see if the name is in it. The call to split_file_path failed (non-successful status returned).

ACTION Use the SETVAR TRACE ERROR command. Repeat the RUN command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

328 *** CANNOT OPEN THE CATALOG (DUIERR 328)

CAUSE The DUI could not open the diagnostic's message catalog. (The call to the service cat_open returned a non-successful status). The most likely cause is that there is no message catalog for the diagnostic written in the native language currently being used on the system.

ACTION Make sure a properly generated message catalog for the diagnostic, written in the same language as the current system default language, has been installed.

---

**329** `*** FAILED TO GET USER INPUT (DUIERR 329)`

CAUSE A call to get_user_input made from print_dui_msg failed. The DUI needed user input because the printing function set it received in print_dui_msg included a reply_pending function.

ACTION Use the SETVAR TRACE ERROR command. Repeat the same sequence of commands which led to the error noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**338** `*** ! COULD NOT BE SUSPENDED (DUIERR 338)`

CAUSE The request to the operating system to suspend the program failed. The program probably terminated between the the time the DUI checked to see if the program existed and the time the operating system received the suspension request.

ACTION If the error can be repeated, submit an SR against the DUI. Use the SETVAR TRACE ERROR command. Repeat the RUN command followed by the SUSPEND command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**339** `*** ! COULD NOT BE RESUMED (DUIERR 339)`

CAUSE The request to the operating system to resume the program failed. The program probably terminated between the the time the DUI checked to see if the program existed and the time the operating system received the resumption request.

ACTION If the error can be repeated, submit an SR against the DUI. Use the SETVAR TRACE ERROR command. Repeat the RUN command followed by the RESUME command as before noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**340** `*** COULD NOT SUSPEND ! - IT IS NOT RUNNING (DUIERR 340)`

CAUSE The request to suspend a specified program failed because the program was not in a "running" state. The program is either already suspended or is aborting or terminating.

ACTION Nothing to be done.

---

| | |
|---|---|
| 341 | `*** COULD NOT RESUME ! - IT IS NOT SUSPENDED (DUIERR 341)` |
| CAUSE | The request to resume a specified program failed because the program was not in a "suspended" state. |
| ACTION | Nothing to be done. |

| | |
|---|---|
| 343 | `*** THE SPECIFIED PROCESS IDENTIFIER IS NOT RECOGNIZED (DUIERR 343)` |
| CAUSE | The specified pid is not in the DUI's process table, so it cannot be suspended, resumed, or aborted. |
| ACTION | Use the SHOWACTIVE command to get a list of the pids of processes which may be suspended, resumed, or aborted. |

| | |
|---|---|
| 344 | `*** THERE ARE NO RUNNING PROCESSES TO SUSPEND (DUIERR 344)` |
| CAUSE | The user gave the suspend command with the ALL option but the DUI's process table does not have any "running" processes in it so nothing can be suspended. |
| ACTION | Nothing to be done. |

| | |
|---|---|
| 345 | `*** THERE ARE NO SUSPENDED PROCESSES TO RESUME (DUIERR 345)` |
| CAUSE | The user gave the resume command with the ALL option but the DUI's process table does not have any "suspended " processes in it so nothing can be resumed. |
| ACTION | Nothing to be done. |

| | |
|---|---|
| 346 | `*** THERE IS MORE THAN ONE RUNNING PROCESS. PLEASE SPECIFY BY PROCESS IDENTIFIER WHICH OF THE FOLLOWING SHOULD BE SUSPENDED (DUIERR 346)` |
| CAUSE | The user gave the suspend command without an option, but the DUI's process table has more than one "running" process in it so the user must specify which process is to be suspended. |
| ACTION | Re-enter the SUSPEND command naming by pid the process to be suspended. |

**347** `*** THERE IS MORE THAN ONE SUSPENDED PROCESS. PLEASE SPECIFY BY PROCESS IDENTIFIER WHICH OF THE FOLLOWING SHOULD BE RESUMED (DUIERR 347)`

CAUSE The user gave the resume command without an option but the DUI's process table has more than one "suspended " process in it so the user must specify which process is to be resumed.

ACTION Use the SHOWACTIVE command to get a list of suspended processes and their pids. Re-enter the RESUME command naming the pid of the process to be suspended.

---

**348** `*** AN INTERRUPT WAS RECEIVED FROM THE USER TERMINAL. (DUIERR 348)`

CAUSE A user interrupt was given at the DUI prompt.

ACTION Nothing to do.

---

**349** `*** ! COULD NOT BE ABORTED (DUIERR 349)`

CAUSE The request to the operating system to abort the program failed. The program may have terminated between the time the DUI checked to see if it was running and the time the OS received the abort request.

ACTION Probably nothing to do. However, if this is repeatable (and not just the result of some very odd timing circumstance) submit an SR against the DUI. Use the SETVAR TRACE ERROR command. Repeat the same series of commands up to and including the ABORT command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**350** `*** THERE IS MORE THAN ONE RUNNING OR SUSPENDED PROCESS. PLEASE SPECIFY BY PROCESS IDENTIFIER WHICH OF THE FOLLOWING SHOULD BE ABORTED (DUIERR 350)`

CAUSE The user gave the abort command without an option but the DUI's process table has more than one process in it so the user must specify which process(es) is(are) to be aborted.

ACTION Use the SHOWACTIVE command to get a list of running and suspended processes along with the associated pids. Re-enter the ABORT command naming one or more of the pids.

---

| | | |
|---|---|---|
| **351** | | `*** THERE ARE NO PROCESSES (DUIERR 351)` |
| CAUSE | | The DUI's process table is empty. The user wanted to abort or suspend a process. |
| ACTION | | There is nothing to be done. |

---

**354** `*** RESERVE WORD LIST MESSAGE ! COULD NOT BE OBTAINED (DUIERR 354)`

CAUSE Call to CAT_READ to obtain the message containing a list of reserved words failed. A more specific error message will precede this one.

ACTION Do the action associated with the message printed immediately before this one.

---

**355** `*** UNABLE TO OBTAIN RESERVED WORD FOR ITEM ! FROM MESSAGE ! (DUIERR 355)`

CAUSE The call to GET_TEXT to obtain a reserved word from a reserved word message failed. The DUI's catalog probably has a bug in it.

ACTION Submit an SR against the DUI.

---

**356** `*** MESSAGE ! WAS NOT ACCEPTED BY SCANNER PACKAGE (DUIERR 356)`

CAUSE The procedure reset_scan failed while trying to accept the message given. This can usually only happen when an empty buffer is passed to the reset_scan function.

ACTION Submit an SR against the DUI.

---

**359** `*** UNABLE TO READ DUI PROMPT FROM MESSAGE CATALOG (DUIERR 359)`

CAUSE The procedure was not able to successfully read the DUI prompt string from the message catalog. The fault might be in the procedure trying to read from the catalog or in the catalog itself.

ACTION Submit an SR against the DUI.

---

**363** *** UNABLE TO READ DUI VERSION STRING FROM MESSAGE CATALOG (DUIERR 363)

CAUSE The procedure was not able to successfully read the DUI version string from the message catalog. The fault might be in the procedure trying to read from the catalog or in the catalog itself.

ACTION Submit an SR against the DUI.

---

**364** *** UNABLE TO READ DUI HEADER FROM MESSAGE CATALOG (DUIERR 364)

CAUSE The procedure was not able to successfully read the DUI header string from the message catalog. The fault might be in the procedure trying to read from the catalog or in the catalog itself.

ACTION Submit an SR against the DUI.

---

**365** *** UNABLE TO READ MONITOR VERSION FROM MESSAGE CATALOG (DUIERR 365)

CAUSE The procedure was not able to successfully read the diagnostic monitor's version display string from the message catalog. The fault might be in the procedure trying to read from the catalog or in the catalog itself.

ACTION Submit an SR against the DUI.

---

**366** *** UNABLE TO READ DUI WARNING MESSAGE FROM CATALOG (DUIERR 366)

CAUSE The procedure was not able to successfully read the DUI warning string from the message catalog. The fault might be in the procedure trying to read from the catalog or in the catalog itself.

ACTION Submit an SR against the DUI.

---

**370** *** UNABLE TO RETRIEVE MESSAGE (!) FROM SET (!) OF THE DUI MESSAGE CATALOG. (DUIERR 370)

CAUSE The service used to pull messages from the DUI's catalog failed to do so. The most likely reason (but not the only possible one) is that the message is not in the catalog.

ACTION The specific reason the message could not be gotten from the DUI's catalog will be printed immediately before this message is. The action taken will depend on that message.

---

**371** `*** UNABLE TO OBTAIN HELP INFORMATION FOR LDEV (!). (DUIERR 371)`

CAUSE The ldev named by the user could not be translated into a corresponding pdev. The diagnostic system uses pdevs internally rather than ldevs. No processing can be done on any device nor can information be gotten about any device designated by ldev unless the diagnostic system can find a corresponding pdev.

ACTION Check to make sure that the ldev is correct and configured into the system. If it is, try again using the corresponding pdev rather than the ldev.

---

**372** `*** UNABLE TO DETERMINE THE IDENTITY OF PDEV (!). FURTHER HELP INFORMATION CANNOT BE OBTAINED. (DUIERR 372)`

CAUSE Could not get a help message for the specified pdev.

ACTION The reason the help message could not be gotten will be printed immediately before this message. Appropriate action depends on what that previous message is.

---

**373** `*** UNABLE TO OBTAIN HELP INFORMATION FOR PDEV (!). (DUIERR 373)`

CAUSE The user asked for the "id" of a particular pdev but the diagnostic system failed to find the product number of the device represented by the pdev.

ACTION The actual reason for the failure will be printed immediately before this message. Appropriate action depends on what that message is.

---

**374** `*** UNABLE TO RETRIEVE MESSAGE (!) FROM SET (!) OF THE CATALOG FOR (!). (DUIERR 374)`

CAUSE The service used to pull messages from diagnostic message catalogs failed to do so. The most likely reason (but not the only possible one) is that the message is not in the catalog.

ACTION The specific reason the message could not be gotten from the diagnostic's catalog will be printed immediately before this message is. The action taken will depend on that message.

---

**375** `*** UNABLE TO OPEN THE MESSAGE CATALOG FOR (!).`
`(DUIERR 375)`

CAUSE Was unable to open a diagnostic's message catalog for reading. The most likely reason for this is that the catalog file has been accidentally purged from the diagnostic system. It is also possible that the diagnostic catalog has not been translated into the language currently being used by the diagnostic system.

ACTION Use the MODIFY command to reinstall the appropriate catalog. If the problem is that the catalog does not exist for the native language currently being used, either change the language being used to one for which a catalog exists or use the MODIFY command to install an existing catalog. When asked which language the catalog is written in, lie and give the current system language. This could cause problems for latter users, but will solve the immediate problem.

---

**376** `*** UNABLE TO LOCATE THE MESSAGE CATALOG FOR (!).`
`(DUIERR 376)`

CAUSE Failed to fully qualify the message catalog name. (That is, the attempt to add the group and account or the directory path of the diagnostic system to the file name failed - the actual reason for the failure is unknown).

ACTION Use the command SETVAR TRACE ERROR. Repeat the command which resulted in the error the first time and note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR noting all of the errors.

---

**377** `*** (!) IS NOT CURRENTLY INSTALLED IN THE DIAGNOSTIC`
`SYSTEM. (DUIERR 377)`

CAUSE The diagnostic was not found in the list of currently installed diagnostics on the present system.

ACTION If a typographical error was made, just redo the command correcting the spelling of the diagnostic name. Otherwise, install the diagnostic.

---

**378** `*** YOU HAVE A DIAGNOSTIC SECURITY LEVEL OF !, BUT A`
`MINIMUM DIAGNOSTIC SECURITY LEVEL OF ! IS NEEDED`
`TO PERFORM THE REQUESTED FUNCTION (DUIERR 378)`

CAUSE The user does not have sufficient security to do requested function.

ACTION Log in as another user (one who has the necessary security level) or ask the system administrator to add the appropriate capability to your account.

---

**379** *** THE DIAGNOSTIC PROGRAM'S CATALOG DOES NOT CONTAIN INFORMATION ABOUT SECTIONS NEEDED TO PROCESS THIS REQUEST(DUIERR 379)

CAUSE In the RUN command, the user named sections without explicitly naming steps for every section named. The diagnostic's catalog does not have the default step message (Set 2, msg 3). Or the user named sections for a diagnostic which does not have any - the catalog does not have a section message.

ACTION Use the HELP command to find out if the diagnostic does have sections. If it does, file an SR against the diagnostic; state that the diagnostic's message catalog is missing the required set 2, message 3. If the diagnostic does not have sections, run it again without naming any sections.

---

**380** *** THE TEST SYSTEM IS RUNNING SO NO TRACING OF THE DUI (OTHER THAN ERROR TRACING) CAN BE DONE AT THIS TIME (DUIERR 380)

CAUSE The user requested some trace of the DUI, but the test system is already handling a codetest or a duitest and so cannot be called upon (through start_test_system) to also do a trace without generating an error.

ACTION Try again after the codetest or duitest is finished. The codetest or duitest is being run by a user under a different DUI on the same machine.

---

**381** *** INAPPROPRIATE I/O REQUEST (DUIERR 381)

CAUSE The DUI was sent an info string as its input (implying that the DUI was invoked programmatically). Input from the user is needed, but no infile or usefile is open so input would have to come from the terminal; however, it is an error to try to get input from the terminal if an info string has been received - the DUI should just exit rather than getting more input at this point.

ACTION Create an infile or usefile with the input the DUI needs and correct the programmatic call to the DUI so that the command received in the info string refers to the infile or usefile.

---

**501** *** SYNTAX ERROR (DUISERR 501)

CAUSE Something was syntactically wrong with a command or a sections/steps message in a diagnostic's catalog. The actual error may be printed out before this message is printed.

ACTION Re-enter the command correctly (use HELP <command> SYNTAX to see the correct syntax). If the RUN command was used and the problem appears to be with the sections/steps, submit an SR against the diagnostic stating that there is a syntactical error in one of the sections/steps messages. Give an exact copy of the command as typed in the SR so that the engineer can narrow down the the problem to the most likely catalog message.

---

**505** `*** CANNOT PARSE MESSAGE CATALOG MESSAGE (DUISERR 505)`

CAUSE A sections/steps message in a diagnostic's message catalog has an open parenthesis ( "(" ) without a matching closing parenthesis ( ")" ).

ACTION Submit an SR against the diagnostic explaining the problem. Give an exact copy of the command as originally typed in the SR so that the engineer can narrow down the problem to the most likely catalog message.

---

**509** `*** INVALID RANGE - ENDING VALUE < THAN STARING VALUE (DUISERR 509)`

CAUSE A section or step number range of the form n/n was given in the RUN command or found in one of the diagnostic message catalog section/step messages. The second number in the range was smaller than the first number.

ACTION If the range was given in the RUN command, re-enter the command using correct numerical values in the range. If the RUN command was correct, submit an SR against the diagnostic explaining the problem. Give an exact copy of the command as originally typed in the SR so that the engineer can narrow down the problem to the most likely section/step catalog message.

---

**513** `*** NO INPUT RECEIVED (DUISERR 513)`

CAUSE A blank line rather than a command was given by the user. Not really an error.

ACTION Ignore message and continue.

---

**515** `*** LINKED LIST BEING CHECKED OR MANIPULATED IS EMPTY (DUISERR 513)`

CAUSE One of the DUI procedures which adds to or removes a section or step from the linked list which the DUI uses internally to determine which sections and steps the user wishes to run was passed a null linked section/step list.

ACTION Use the command SETVAR TRACE ERROR. Repeat the command which resulted in the error the first time and note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**522** `*** NEEDED A CLOSING PARENTHESIS ")" (DUISERR 522)`

CAUSE A set of steps in a diagnostic's message catalog section/step message or a step given with the HELP STEP command was preceded by an open-parenthesis ( "(" ) but had no matching closing parenthesis ( ")" ).

ACTION If the HELP STEP command was at fault, re-enter the command with a closing parenthesis. Otherwise, enter the SETVAR TRACE ERROR command. Repeat the command which resulted in the error the first time and note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the diagnostic noting all of the errors.

---

**525** `*** NEEDED A CLOSING SQUARE BRACKET "]" (DUISERR 525)`

CAUSE The RUN command was given with an open square bracket ( "[" ) used to signify the beginning of a list of steps. The step list was not terminated with a matching closing square bracket ( "]" ).

ACTION Re-enter the RUN command correctly.

---

**530** `*** PRINTING ENVIRONMENT EXPECTED (DUISERR 530)`

CAUSE The "env" keyword was given with either the HARDCOPY command or the RUN command with a HARDCOPY parameter, but no printing environment was specified.

ACTION Redo the command either specifying a printing environment or without the "env" keyword.

---

**533** `*** EXPECTED THE NAME OF A FILE (DUISERR 533)`

CAUSE A command which requires a file name or a diagnostic name was given without naming a file or diagnostic.

ACTION Redo the command, giving the appropriate file name or diagnostic name. Enter the HELP <command> or HELP <command> SYNTAX for more information.

---

**536** `*** LDEV SPECIFICATION EXPECTED (DUISERR 536)`

CAUSE The "ldev" keyword was given with a command, but no ldev was named.

ACTION Redo the command specifying an appropriate ldev.

---

**538** `*** PDEV SPECIFICATION EXPECTED (DUISERR 538)`

CAUSE The "pdev" keyword was given with a command, but no pdev was named.

ACTION Redo the command specifying an appropriate pdev.

---

**541** `*** FILE NAME OR "OFF" EXPECTED HERE (DUISERR 541)`

CAUSE The OUTFILE command was given or the RUN command was given with an OUTFILE parameter but without naming an outfile or specifying "off."

ACTION Redo the command either specifying an outfile or "off."

---

**545** `*** "OFF" OR A NUMBER EXPECTED HERE (DUISERR 545)`

CAUSE The RUN command was given with the "loop" parameter, but neither a number nor the word "off" was specified.

ACTION Re-enter the command specifying either a number or the option "off."

---

**546** `*** "ON" OR "OFF" EXPECTED (DUISERR 546)`

CAUSE The RUN command was given with either the "erronly" or "errpause" parameter, but the parameter was not followed by either the word "on" or "off."

ACTION Re-enter the command specifying either "on" or "off" after the parameter.

---

**553** `*** TRACING OPTION EXPECTED HERE (DUISERR 553)`

CAUSE The "trace" parameter was given with the SETVAR or RUN command but no tracing option was specified.

ACTION Re-enter the command specifying a tracing option. Use the HELP TRACE command to get a list of tracing options.

---

**555** `*** QUOTED TEXT STRING EXPECTED HERE (DUISERR 555)`

CAUSE Diagnostic specific parameters were assumed to be given with the RUN command since a double-quote ( " ) was found in the command. However, there was something wrong with the quoted string - it might not have had a terminating quote or might have been null or might have had some other problem.

ACTION Re-enter the command, correcting the string parameter.

---

**560** `*** APPROPRIATE MNEMONIC MESSAGE COULD NOT BE FOUND IN THE DIAGNOSTIC'S CATALOG (DUISERR 560)`

CAUSE The user gave mnemonics in place of some section(s) or step(s) in the RUN command, but no message could be found in the diagnostic's message catalog which would allow the DUI to translate the mnemonic into the corresponding numbers.

ACTION Re-enter the RUN command using section/step numbers instead of mnemonics. A list of the sections and steps in the diagnostic may be gotten by using the HELP <diagnostic name> SECTIONS command.

---

**563** `*** COULD NOT FIND THE NUMBER BEING SEARCHED FOR IN THE CATALOG MESSAGE (DUISERR 563)`

CAUSE The DUI tried to remove a section or step number from the linked list of sections/steps it maintains while determining which sections and steps the user wishes to run with a diagnostic. The section/step number was not in the list. This is not a real error condition.

This happens when the DUI's list has been made using default sections and/or steps. The number not found was not found because it was not in the diagnostic's catalog message containing default sections or steps.

The user should never see this message.

ACTION Ignore the message if the diagnostic runs after the message appears. If the diagnostic does not run, submit an SR against the DUI stating that this error message appeared and giving an exact copy of the RUN command which resulted in the message appearing.

---

**565** `*** NEITHER A NUMBER NOR A MNEMONIC WAS FOUND (DUISERR 565)`

CAUSE A number or mnemonic was searched for but not found in either a diagnostic catalog section/step message or in a command given by the user. This is not necessarily an error; for instance, it might just signify that the end of a list of sections or steps had been reached. However, this message will only be printed when the condition is an error - as when the user gives the RUN command with the "sections" parameter but fails to name any sections.

ACTION Re-enter the command, specifying a number where needed.

---

**570** `*** EXPECTED A NUMBER OR MNEMONIC AFTER THE COMMA (DUISERR 570)`

CAUSE A number list was given with a command for which such a list is appropriate. One of the numbers was followed by a comma but no number appeared after the comma. If the RUN command was given, either a number or a mnemonic would have been acceptable after the comma, but neither appeared.

ACTION Re-enter the command. Either specify a number after the comma or do not type the last comma.

---

**573** `*** EXPECTED A NUMBER AFTER THE RANGE SIGN (DUISERR 573)`

CAUSE A command was given with a number range ( "n/n" ), but the last number in the range was missing. No number followed the range sign so what was read was of the form "n/".

ACTION Re-enter the command specifying the last number in the range.

---

**574** *** EXPECTED A NUMBER OR MNEMONIC AFTER THE SIGN (DUISERR 574)

CAUSE A "+" or "-" sign was found while trying to parse a command (most likely the RUN command), but the sign was not followed by a number or mnemonic.

ACTION Re-enter the command either specifying a number or mnemonic after the sign or eliding the sign.

---

**583** *** "OFF" OR A DEBUGGER NAME EXPECTED HERE (DUISERR 583)

CAUSE The RUN command was given with the "debug" parameter, followed by an equals sign ( "debug=" ), but neither a debugger name nor the word "off" was found after the sign.

ACTION Re-enter the command either specifying a debugger or giving the "off" keyword. The "debug" parameter may also be given without the equals sign - signifying that the default debugger should be used.

---

**585** *** "OFF," "ON," "LONG," OR "SHORT EXPECTED HERE (DUISERR 585)

CAUSE The RUN command was given with the "errprint" parameter, followed by an equals sign ( "errprint =" ). But, none of the possible options to "errprint" was found after the equals sign.

ACTION Use the HELP ERRPRINT command to see a list of all possible options and their use. Re-enter the command either naming an option after the "errprint" parameter or omitting the equals sign.

---

**592** *** PROGRAM IS NOT INSTALLED (DUISERR 592)

CAUSE What was assumed to be a diagnostic name given with the RUN or HELP command was not found in the diagnostic system's list of installed diagnostics.

ACTION If a typographical error was made re-enter the command. If the diagnostic is not installed, install it. A list of installed diagnostics may be gotten by using the LIST command.

---

**593** *** OPTION IS INVALID FOR THIS COMMAND (DUISERR 593)

CAUSE A command was given containing a keyword which is valid for another command, but not for the given command.

ACTION Re-enter the command using the correct syntax. The valid form for the command may be seen by using HELP <command> or HELP <command> SYNTAX.

---

**595** `*** THE SECTION NUMBER WAS NOT FOUND IN THE MESSAGE CATALOG MESSAGE (DUISERR 595)`

CAUSE: The DUI was unable to find a particular section number (probably given in the RUN command) in the diagnostic's message catalog message of valid sections.

ACTION: Use the HELP <diagnostic> SECTIONS command to see a list of valid sections for the diagnostic. Re-enter the command.

---

**598** `*** UNEXPECTED TOKEN FOUND (DUISERR 598)`

CAUSE: While trying to parse a diagnostic catalog section/step message an unexpected punctuation mark was found (something other than a slash, open-parenthesis, close-parenthesis, or comma).

ACTION: Use the SETVAR TRACE ERROR command. Repeat the RUN command noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the diagnostic noting all of the errors.

The RUN command may also be redone naming the sections in some alternate way. That is, if the original command used number ranges try listing the numbers explicitly. If the original command used mnemonics, try the numbers instead. If the original command relied on the defaults (i.e. didn't specify numbers or mnemonics), explicitly name the sections/steps to be run. The diagnostic catalog section/step messages which the DUI must parse in order to determine which sections/steps to run are dependent on the form of the input used in the RUN command. By changing the form of input, it is possible that the message with the error can be avoided. An SR should still be submitted though even if this works.

---

**599** `*** UNRECOGNIZED INPUT (DUISERR 599)`

CAUSE: A command was given which contained unrecognized garbage. Usually something extra and unexpected was found at the end of the command line. This could also appear if a valid parameter which can only be used once with the command is repeated.

ACTION: Use the HELP <command> or HELP <command> SYNTAX to see the valid form of the command. Re-enter the command.

---

**600** `*** PROGRAM IDENTIFIERS ALREADY GIVEN - "ALL" NOT VALID (DUISERR 600)`

CAUSE: The ABORT command was given with the program identifiers of the programs to be aborted. Somewhere after the identifier list, the word "all" appeared. "All" is not valid if a program identifier has already been named.

ACTION: Redo the command, eliding the "all."

---

**601** *** "ALL" ALREADY GIVEN - PROGRAM IDENTIFIERS ARE NOT VALID (DUISERR 601)

CAUSE The ABORT command was given with the "all" parameter. Somewhere after the "all" a program identifier was named. Program identifiers are not valid if the "all" parameter has already been used.

ACTION Redo the command, deleting the program identifier(s).

---

**602** *** A BLANK IS NOT VALID AFTER THE ASTERISK (DUISERR 602)

CAUSE A hardcopy environment was given beginning with an asterisk, but a space or tab appeared after the asterisk - this is not allowed.

ACTION Redo the command, removing the blank or tab.

---

**603** *** A TYPE (DIAGNOSTIC, EXERCISER, VERIFIER, OR UTILITY) IS NEEDED (DUISERR 603)

CAUSE The user gave the LIST command with the "type" parameter but did not specify which type he wanted information about.

ACTION Use the HELP LIST or HELP LIST SYNTAX command to see a list of valid list types. Re-enter the LIST command.

---

**604** *** A PRODUCT NAME IS NEEDED (DUISERR 604)

CAUSE The user gave the LIST command with the "product" parameter but did not specify which product he wanted information about.

ACTION Use the HELP LIST or HELP LIST SYNTAX command to see a list of valid products. Re-enter the LIST command.

---

**605** *** A PROGRAM NAME WAS EXPECTED AFTER THE COMMA (DUISERR 605)

CAUSE The LIST command was given with program names. A comma followed one of the program names but was not itself followed by a program name in turn.

ACTION Redo the LIST command, either adding a diagnostic program name after the offending comma or omitting the comma.

---

**606** *** A PARAMETER WAS EXPECTED AFTER THE EQUALS SIGN (DUISERR 606)

CAUSE An equals sign appearing after a command was not followed by a parameter.

ACTION Use HELP <command> or HELP <command> SYNTAX to see the valid form for the command. Re-enter the command.

---

607 *** A COMMAND IDENTIFIER WAS EXPECTED AFTER THE EQUALS SIGN (DUISERR 607)

CAUSE The DO or REDO command was given with the "cmd=" parameter. However, no command identifier appeared after "cmd="

ACTION Redo the command, specifying a command identifier. Use the HELP DO or HELP REDO commands for a complete explanation of valid input.

---

608 *** AN EQUALS SIGN IS EXPECTED AFTER "CMD" AND "EDIT" (DUISERR 608)

CAUSE The DO or REDO command was given with the "cmd=" and/or the "edit=" parameter. However, the "=" was missing. The equals sign is not optional in this case.

ACTION Redo the command, inserting the equals sign.

---

609 *** EDITING DIRECTIVES WERE EXPECTED HERE (DUISERR 609)

CAUSE The DO or REDO command was given without specifying editing directives after naming one of the parameters which must be followed by editing directives (";edit" or ",").

ACTION Re-enter the command, specifying editing directives. Use the HELP DO or HELP REDO command for information about editing directives.

---

610 *** A STRING WAS EXPECTED HERE (DUISERR 610 )

CAUSE The DO or REDO command was given with the "cmd=" and/or ";edit" or "," parameter tokens. A legitimate command identifier/edit directive was not found after the parameter.

ACTION Use the HELP DO or HELP REDO command to see a discussion of command identifiers and editing directives. Redo the DO or REDO command.

---

611 *** THIS STRING DOES NOT HAVE A TERMINATOR. (DUISERR 611)

CAUSE The DO or REDO command was given with a quoted string as a command identifier or editing directive. However, no terminating single or double quote was found to match the quote with which the string began.

ACTION Redo the command, adding the terminating quotation character.

---

| | | |
|---|---|---|
| 612 | | *** EXPECTED A NUMBER AFTER THE EQUALS SIGN. (DUISERR 612) |
| CAUSE | | The HELP <diagnostic> SECTION or HELP <diagnostic> SC command was given with an equals sign immediately following the SECTION or SC. No number appeared after the equals sign. |
| ACTION | | Redo the command either inserting a number after the equals sign or deleting the equals sign. |

| | |
|---|---|
| 613 | *** EXPECTED A NUMBER HERE. (DUISERR 613) |
| CAUSE | A number was not found in a command when one was required. |
| ACTION | Use HELP <command> or HELP <command> SYNTAX for information about the correct form of the command. Redo the command. |

| | |
|---|---|
| 614 | *** NEED A PROGRAM NAME (DUISERR 614) |
| CAUSE | The RUN command was given, but no program was named. |
| ACTION | Redo the command, inserting the name of an installed diagnostic program. A list of installed programs may be gotten by using the LIST command. The proper form for the RUN command may be gotten by using the HELP RUN command. |

| | |
|---|---|
| 616 | *** THIS CANNOT BE A PROGRAM IDENTIFIER (DUISERR 616) |
| CAUSE | A command was given with a program identifier as a parameter. But the program identifier given was too large to be a legitimate pid. |
| ACTION | Use the SHOWACTIVE command to get a list of active diagnostic processes and their identifiers. Re-enter the original command giving the correct pid. |

| | |
|---|---|
| 802 | *** CAN'T READ THE TEMPORARY DIAGNOSTIC LIST FILE. (DUIIERR 802) |
| CAUSE | This error means that the temporary diagnostic list file could not be read from. The system may be corrupted at this point. |
| ACTION | Submit an SR explaining the problem. |

| | |
|---|---|
| 803 | *** THE FILE PURGE LIST FILE COULD NOT BE OPENED. (DUIIERR 803) |
| CAUSE | This means the temporary purge file could not be opened. The file should exist. |
| ACTION | Submit an SR explaining the problem. |

**805** `*** THE FILE PURGE LIST FILE COULD NOT BE CLOSED. (DUIIERR 805)`

CAUSE The error means that the temporary purge file could not be closed.

ACTION Submit an SR explaining the problem.

---

**806** `*** CAN'T DELETE THE FILE !. (DUIIERR 806)`

CAUSE This means that a file that was involved with a deleted diagnostic could not be deleted. File protected may have been altered or the file actually doesn't exist which means the directory has been corrupted.

ACTION You can delete the file by hand if it still exists but there is no reason that the file should have a problem being deleted. If you can't, submit an SR explaining the problem.

---

**807** `*** CAN'T DELETE THE TEMPORARY PURGE LIST FILE. (DUIIERR 807)`

CAUSE This means that the temporary purge file could not be deleted.

ACTION Submit an SR explaining the problem.

---

**808** `*** THE FILE PURGE LIST FILE COULD NOT BE READ. (DUIIERR 808)`

CAUSE This means the temporary purge file was opened but can't be read from.

ACTION Submit an SR explaining the problem.

---

**902** `*** THE TEMPORARY DIAGNOSTIC LIST FILE HAS BEEN CORRUPTED. (DUIIERR 902)`

CAUSE This means that a copy of the temporary diagnostic list file was found to be corrupted during a modify catalog or modify file command.

ACTION Submit an SR explaining the problem.

---

| | |
|---|---|
| **903** | `*** THE VERSION ! IS INVALID.  TRY AGAIN. (DUIIERR 903)` |
| CAUSE | The program version in a catalog or one input from a modify code command has an invalid syntax. The syntax should be a.bb.cc. If the modify code command was chosen the version may be lower than the previous version. |
| ACTION | A higher version should be entered if in a modify code command otherwise the version in the catalog should be corrected. |

| | |
|---|---|
| **904** | `*** CAN'T DELETE THE TEMPORARY MODIFY DIAGNOSTIC LIST FILE.  (DUIIERR 904)` |
| CAUSE | This means that a temporary diagnostic list file could not be deleted during a modify command. |
| ACTION | Submit an SR explaining the problem. |

| | |
|---|---|
| **908** | `*** ! IS ALREADY TESTED BY !. (DUIIERR 908)` |
| CAUSE | This means that a device which is already being tested by the program is being added to the list of devices the program tests. If this occurs during an install, then a duplicate device name exists in the device list. |
| ACTION | Please re-enter a device that is not tested by this program. |

| | |
|---|---|
| **911** | `*** THE ! CATALOG ALREADY EXISTS. (DUIIERR 911)` |
| CAUSE | This means that a catalog is being added to the list of catalogs belonging to the program but the program already has a catalog with that language. If this is during an install then the catalog in the DUI's language needs to be edited. |
| ACTION | Please re-enter the language. |

| | |
|---|---|
| **1001** | `*** THE TEMPORARY DIAGNOSTIC DIRECTORY FILE HAS BEEN CORRUPTED. (DUIIERR 1001)` |
| CAUSE | This means the copy of the diagnostic directory has been corrupted. It could also mean the real diagnostic directory is corrupted. |
| ACTION | Submit an SR explaining the problem. |

1201 `*** THE CATALOG ROOT COULD NOT BE FORMED (DUIIERR 1201)`

CAUSE This means the catalog root in the diagnostic directory could not be converted to a string.

ACTION Submit an SR explaining the problem.

---

1202 `*** THE CATALOG NAME COULD NOT BE FORMED. (DUIIERR 1202)`

CAUSE This means the destination name of a catalog file could not be formed.

ACTION Submit an SR explaining the problem.

---

1204 `*** THE DUI CATALOG COULD NOT BE ACCESSED. (DUIIERR 1204)`

CAUSE This means a message cannot be read from the DUI's catalog.

ACTION Submit an SR explaining the problem.

---

1205 `*** THE FILE ! COULD BE OPENED BUT NOT PROPERLY CLOSED. (DUIIERR 1205)`

CAUSE This means a file input by the user exists and was opened but the file header was corrupted or the file could not be closed properly.

ACTION Submit an SR explaining the problem.

---

1206 `*** THE FILE ! EXISTS BUT IT COULD NOT BE OPENED PROPERLY. (DUIIERR 1206)`

CAUSE The file exists but it could not be opened properly. The protection on the file may be bad.

ACTION Submit an SR explaining the problem.

---

1209 `*** THE FILE PATH ! COULD NOT BE PARSED OR IS INCOMPLETE. (DUIIERR 1209)`

CAUSE This means the path of the file is invalid for this operating system or it is incomplete.

ACTION Correct the path if it is incorrect or invalid or incomplete. If not, submit an SR explaining the problem.

---

**1212** `*** ! COULD NOT BE COPIED TO !. (DUIIERR 1212)`

CAUSE This is caused when the source cannot be copied to the destination. The source may not exist or can't be accessed. The destination may already exist or the disk space is used up.

ACTION Submit an SR explaining the problem.

---

**1213** `*** THE DIAGNOSTIC SYSTEM FILE ! IS INVALID. (DUIIERR 1213)`

CAUSE This means one of the system files to be added to the diagnostic list file is invalid.

ACTION Submit an SR explaining the problem.

---

**1215** `*** THE DEVICE NAME ! IS TOO LONG. (DUIIERR 1215)`

CAUSE This is caused when a device entered is too long. If it is during an installation then the catalog has a problem.

ACTION Re-enter the device name or fix the catalog.

---

**1218** `*** THE FILE ! DOES NOT EXIST. (DUIIERR 1218)`

CAUSE This means the input file does not exist.

ACTION Input a valid existing file name that follows diagnostic naming rules discussed in the diagnostic development guide. If problems continue, submit an SR explaining the problem.

---

**1219** `*** THE DUI COULD NOT SCAN FOR USER COMMANDS.`
`(DUIIERR 1219)`

CAUSE This means the list of valid program types has been corrupted.

ACTION Submit an SR explaining the problem.

---

**1223** `*** THE FILE PATH ! IS TOO LONG. (DUIIERR 1223)`

CAUSE This means the file path of the input file is too long.

ACTION Input a file with a valid shorter path. If the problems continues, submit an SR explaining the problem.

---

**1224** `*** THE TEMPORARY DIAGNOSTIC LIST FILE CANNOT BE APPENDED TO. (DUIIERR 1224)`

CAUSE This means an internal temporary file cannot be appended to.

ACTION Submit an SR explaining the problem.

---

**1225** `*** THE DIAGNOSTIC DIRECTORY FILE COULD NOT BE CLOSED PROPERLY. (DUIIERR 1225)`

CAUSE This means the actual diagnostic directory file cannot be closed.

ACTION Submit an SR explaining the problem.

---

**1226** `*** THE DIAGNOSTIC LIST FILE COULD NOT BE CLOSED PROPERLY. (DUIIERR 1226)`

CAUSE This means the real diagnostic list file or a temporary one cannot be closed properly.

ACTION Submit an SR explaining the problem.

---

**1227** `*** THE TEMPORARY DIAGNOSTIC DIRECTORY FILE COULD NOT BE CLOSED PROPERLY. (DUIIERR 1227)`

CAUSE This means a temporary diagnostic directory file could not be closed properly.

ACTION Submit an SR explaining the problem.

---

1228 *** THE TEMPORARY DIAGNOSTIC LIST FILE COULD NOT BE CLOSED PROPERLY. (DUIIERR 1228)

CAUSE This means a temporary diagnostic list file could not be closed properly.

ACTION Submit an SR explaining the problem.

---

1229 *** THE USER CATALOG COULD NOT BE CLOSED PROPERLY. (DUIIERR 1229)

CAUSE This means the catalog the user input as the one with the same language as the DUI could not be closed properly.

ACTION Submit an SR explaining the problem.

---

1231 *** THE DIAGNOSTIC DIRECTORY FILE COULD NOT BE CREATED. (DUIIERR 1231)

CAUSE This means the actual diagnostic directory file could not be created.

ACTION Check if there is any disk space left.

---

1232 *** THE DIAGNOSTIC LIST FILE ! COULD NOT BE CREATED. (DUIIERR 1232)

CAUSE This means the actual diagnostic list file could not be created.

ACTION Check to see if there is any disk space left.

---

1233 *** THE DIAGNOSTIC DIRECTORY FILE COULD NOT BE DELETED. (DUIIERR 1233)

CAUSE This means the old diagnostic directory file could not be deleted.

ACTION Submit an SR explaining the problem.

---

1234 `*** THE DIAGNOSTIC LIST FILE COULD NOT BE DELETED.`

`(DUIIERR 1234)`

CAUSE This means the old diagnostic list file could not be deleted.

ACTION Submit an SR explaining the problem.

---

1235 `*** THERE WAS NO DEVICE GIVEN IN THE MESSAGE CATALOG.`

`(DUIIERR 1235)`

CAUSE This means the user catalog had a device message which was empty. It can also be a problem parsing a device name.

ACTION Correct the device name by editing the catalog. If the problem continues, submit an SR explaining the problem.

---

1236 `*** THE DESTINATION FILE PATH COULD NOT BE CREATED.`

`(DUIIERR 1236)`

CAUSE This means the destination file path of a catalog, program, or downloadable file could not be parsed.

ACTION Submit an SR explaining the problem.

---

1237 `*** THE DIAGNOSTIC DIRECTORY FILE COULD NOT BE OPENED.`

`(DUIIERR 1237)`

CAUSE The actual diagnostic directory file could not be opened.

ACTION Submit an SR explaining the problem.

---

1238 `*** THE DIAGNOSTIC LIST FILE COULD NOT BE OPENED.`

`(DUIIERR 1238)`

CAUSE The actual diagnostic list file could not be opened.

ACTION Submit an SR explaining the problem.

---

| | | |
|---|---|---|
| 1239 | | `*** THE TEMPORARY DIAGNOSTIC DIRECTORY COULD NOT BE OPENED. (DUIIERR 1239)` |
| CAUSE | | A temporary diagnostic directory file could not be opened. |
| ACTION | | Submit an SR explaining the problem. |

---

| | | |
|---|---|---|
| 1240 | | `*** THE TEMPORARY DIAGNOSTIC LIST FILE COULD NOT BE OPENED. (DUIIERR 1240)` |
| CAUSE | | A temporary diagnostic list file could not be opened. |
| ACTION | | Submit an SR explaining the problem. |

---

| | | |
|---|---|---|
| 1241 | | `*** THE USER CATALOG COULD NOT BE OPENED. (DUIIERR 1241)` |
| CAUSE | | This means the user catalog matching the language of the DUI could not be opened. |
| ACTION | | Submit an SR explaining the problem. |

---

| | | |
|---|---|---|
| 1242 | | `*** THE TEMPORARY DIAGNOSTIC DIRECTORY FILE COULD NOT BE READ. (DUIIERR 1242)` |
| CAUSE | | A temporary diagnostic directory file could not be read from. |
| ACTION | | Submit an SR explaining the problem. |

---

| | | |
|---|---|---|
| 1243 | | `*** THE USER CATALOG COULD NOT BE READ. (DUIIERR 1243)` |
| CAUSE | | This means the user catalog matching the language of the DUI could not be read. |
| ACTION | | Submit an SR explaining the problem. |

---

**1245** `*** THE TEMPORARY DIAGNOSTIC DIRECTORY COULD NOT BE WRITTEN TO. (DUIIERR 1245)`

CAUSE The temporary diagnostic directory file could not be written to.

ACTION Submit an SR explaining the problem.

---

**1246** `*** ONE OF !'S DEVICE OPTIONS IS INVALID. (DUIIERR 1246)`

CAUSE This means that a device had a option that was invalid or a duplicate of one of its existing options. Valid options are AUTO and DECODE.

ACTION If this happened interactively re-enter the device and its correct options. If this happened during an install then the catalog needs to be fixed.

---

**1247** `*** THE FILE PATH ! IS TOO LONG. (DUIIERR 1247)`

CAUSE This means the path to this file is too long.

ACTION Re-enter a shorter path.

---

**1250** `*** THE DUI AND DIAGNOSTIC DIRECTORY FILE ARE INCOMPATIBLE. (DUIIERR 1250)`

CAUSE This means the version of the DUI used to create the current on-line diagnostic system is not the version of the current DUI.

ACTION Re-install the on-line diagnostic system from the install tape and add in any changes you have made.

---

**1251** `*** THE TEMPORARY DIAGNOSTIC DIRECTORY COULD NOT BE DELETED. (DUIIERR 1251)`

CAUSE The temporary diagnostic directory file could not be deleted.

ACTION Submit an SR explaining the problem.

---

**1252** *** THE TEMPORARY DIAGNOSTIC LIST FILE COULD NOT BE DELETED. (DUIIERR 1252)

CAUSE A temporary diagnostic list file could not be deleted.

ACTION Submit an SR explaining the problem.

---

**1253** *** THE DEFAULT SYSTEM PATH COULD NOT BE FORMED. (DUIIERR 1253)

CAUSE This means the default on-line diagnostic system location could not be parsed correctly.

ACTION Submit an SR explaining the problem.

---

**1254** *** THE FILE ADDITION LIST FILE COULD NOT BE OPENED. (DUIIERR 1254)

CAUSE A temporary diagnostic add file could not be opened.

ACTION Submit an SR explaining the problem.

---

**1255** *** THE FILE ADDITION LIST FILE COULD NOT BE READ. (DUIIERR 1255)

CAUSE A temporary diagnostic add file could not be read from.

ACTION Submit an SR explaining the problem.

---

**1256** *** THE FILE ADDITION LIST FILE COULD NOT BE WRITTEN TO. (DUIIERR 1256)

CAUSE A temporary diagnostic add file could not be written to.

ACTION Submit an SR explaining the problem.

---

**1257** `*** THE FILE ADDITION LIST FILE COULD NOT BE CLOSED. (DUIIERR 1257)`

CAUSE A temporary diagnostic add file could not be closed.

ACTION Submit an SR explaining the problem.

---

**1258** `*** CAN'T FORM THE TEMPORARY DIAGNOSTIC DIRECTORY FILE PATH. (DUIIERR 1258)`

CAUSE This means the file path for a temporary on-line diagnostic system file could not be parsed correctly.

ACTION Submit an SR explaining the problem.

---

**1259** `*** CAN'T FORM THE TEMPORARY DIAGNOSTIC LIST FILE PATH. (DUIIERR 1259)`

CAUSE This means the file path for a temporary on-line diagnostic system file could not be parsed correctly.

ACTION Submit an SR explaining the problem.

---

**1260** `*** CAN'T FORM THE TEMPORARY DIAGNOSTIC PURGELIST FILE PATH (DUIIERR 1260)`

CAUSE This means the file path for a temporary on-line diagnostic system file could not be parsed correctly.

ACTION Submit an SR explaining the problem.

---

**1261** `*** CAN'T FORM THE TEMPORARY DIAGNOSTIC ADDLIST FILE PATH. (DUIIERR 1261)`

CAUSE This means the file path for a temporary on-line diagnostic system file could not be parsed correctly.

ACTION Submit an SR explaining the problem.

---

1262 `*** THE FILE ADDITION LIST FILE COULD NOT BE DELETED.`

`(DUIIERR 1262)`

CAUSE A temporary diagnostic add file could not be deleted.

ACTION Submit an SR explaining the problem.

---

1263 `*** THE FILE ADDITION LIST FILE COULD NOT BE CREATED.`

`(DUIIERR 1263)`

CAUSE A temporary diagnostic add file could not be created.

ACTION Check to see if there is disk space available.

---

1264 `*** A FATAL DUI ERROR HAS OCCURRED ... ABORTING.`

`(DUIIERR 1264)`

CAUSE This means an internal error occurred in print_dui_message while attempting to get a reply from the user.

ACTION Submit an SR explaining the problem.

---

1265 `COULD NOT RECEIVE MONITOR REPLY FOR A SEMAPHORE REQUEST (DUIERR 1265)`

CAUSE The `DUI` could not receive an ipc message from the diagnostic monitor.

ACTION Verify the diagnostic system is running correctly. If so, submit an SR explaining the problem.

---

1266 `COULD NOT SEND A SEMAPHORE REQUEST TO THE MONITOR (DUIERR 1266)`

CAUSE The `DUI` could not send a message to the diagnostic monitor

ACTION Verify the diagnostic system is running correctly. If so, submit an SR explaining the problem.

---

1267 `COULD NOT GET THE MONITOR PORT DURING A SEMAPHORE REQUEST (DUIERR 1267)`

CAUSE The `DUI` could not get the diagnostic monitor's port number.

ACTION Verify the diagnostic system is running correctly. If so, submit an SR explaining the problem.

---

**4001** `*** COULD NOT OPEN FILE ! (DUIERR 4001)`

CAUSE The DUI could not open a usefile for reading or an outfile for writing. The specific cause of the failure is not known.

ACTION Use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error and note the additional errors which will appear. An easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Unfortunately, OUTFILE and HARDCOPY might not work in this case. Submit an SR against the DUI noting all of the errors.

---

**4002** `*** COULD NOT READ FILE ! (DUIERR 4002)`

CAUSE Could not read from a usefile or infile which had previously been opened successfully for reading. The specific cause of the error is not known.

ACTION First, check the file to see if there is anything obviously wrong with it (such as being in binary rather than ASCII). If the file seems to be correct use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error appearing noting the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**4003** `*** COULD NOT CLOSE FILE ! (DUIERR 4003)`

CAUSE Failed to close an infile, outfile, or usefile which had previously been opened successfully. The specific reason for the failure is not known.

ACTION Check to see if the file still exists - it might have somehow been purged from the system after being opened.

If the file still exists then use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error appearing using a different infile, outfile, or usefile if necessary. Note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called - OUTFILE might not work if the original problem involved an outfile). Submit an SR against the DUI noting all of the errors.

---

**4004** `*** FILE ! DOES NOT EXIST (DUIERR 4004)`

CAUSE Could not open a file because it did not exist.

ACTION Create the file.

---

**4005** `*** SECURITY VIOLATION ON FILE ! (DUIERR 4005)`

CAUSE Could not open an outfile or usefile because the user does not have permission to read or write to it.

ACTION Either change the permissions on the file, log in as a user with a higher security level, or have the system administrator increase your security capabilities.

---

| | | |
|---|---|---|
| **4009** | `*** ACCESS TO THE FILE CANNOT BE GRANTED BECAUSE IT IS CURRENTLY OPENED EXCLUSIVELY. (DUIERR 4009)` | |

CAUSE Either could not open or could not read a file because another user or process already has opened the file exclusively.

ACTION Retry the command after the file has been closed.

---

**4010** `*** THE FILE IS NOT CURRENTLY OPEN (DUIERR 4010)`

CAUSE Could not read from, write to, or close a particular file because the file descriptor the procedure was trying to use was not recognized by the file system.

ACTION See if the file has already been closed or if it still exists. If the file is open then use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error appearing using a different infile, outfile, usefile, or hardcopy device if necessary. Note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called - these might not work if the original problem involved an outfile or hardcopy device). Submit an SR against the DUI noting all of the errors.

---

**4012** `*** HARDCOPY ENVIRONMENT ! IS INVALID (DUIERR 4012)`

CAUSE There is something wrong with the environment specified with the HARDCOPY command or hardcopy parameter to the RUN command.

ACTION Re-enter the command either specifying a correct environment or not specifying any environment (i.e. use the default environment). The "correct" environment is operating system and device dependent; the diagnostic system has no control over this feature.

---

**4015** `*** USEFILE IS NOT OPEN (DUIERR 4015)`

CAUSE Either could not read from or could not close a usefile because none were open.

ACTION See if the usefile has already been closed or has been purged from the system. If neither of these is the case then use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error appearing using a different usefile if necessary. Note the additional errors which will appear (an easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called). Submit an SR against the DUI noting all of the errors.

---

**4017** `*** INVALID COMMAND, ONLY RUN COMMAND AVAILABLE FOR CODETEST (DUIERR 4017)`

CAUSE Successfully parsed a command line from a codetest file, however, the command was not the run command. The only DUI command allowed within a codetest file is the run command.

ACTION Remove the command from the codetest file.

---

**4031** `*** FILE ! ALREADY EXISTS (DUIERR 4031)`

CAUSE The OUTFILE command or the RUN command with the outfile parameter was given. But, an outfile could not be created using the name specified for it because a file with that name already exits.

ACTION Redo the command, giving a different name for the outfile which is to be created.

---

**4032** `*** UNABLE TO CREATE THE SPECIFIED OUTFILE (DUIERR 4032)`

CAUSE Could not create the specified outfile for some reason. The actual reason for the failure will be printed immediately before this message.

ACTION Action taken will depend on the reason for the failure; see the cause/action statement for the message which will immediately precede this one.

---

**5009** `*** THE DUI RECEIVED A NEGATIVE OR ZERO VALUE FOR REDO STACK SIZE (DUIDOREDO 5009)`

CAUSE The REDOSAVE or REDOSIZE command was given with a negative redo stack size specified. The redo stack size must be a value greater than zero.

ACTION Repeat the command, giving a positive integer value for the size of the history stack.

---

**5021** `*** THE DUI TRIED TO PLACE A COMMAND STRING INTO THE SCANNER INPUT BUFFER, BUT ENCOUNTERED AN ERROR. (DUIDOREDO 5021)`

CAUSE The DO or REDO command was given and the command to be repeated was successfully gotten from the history stack. However, the attempt to place the command in a buffer and point to the first character in the command so that it could be parsed failed. This is a particularly strange error which should never occur.

ACTION Use the SETVAR TRACE ERROR command. Repeat the command which resulted in the error and note the additional errors which will appear. An easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit an SR against the DUI noting all of the errors.

---

**5022** *** THE DUI TRIED TO LOCATE THE NUMERIC COMMAND ID ON THE COMMAND HISTORY STACK, BUT THE NUMBER WAS NOT THERE. (DUIDOREDO 5022)

CAUSE The DO or REDO command was given with a numeric command identifier, but no matching command could be found in the history stack.

ACTION Use the LISTREDO command to see the diagnostic system history stack. Then re-enter the command, giving a correct command identifier. Information about command identifiers may be found by using the HELP DO or HELP REDO commands.

---

**5023** *** THE DUI TRIED TO LOCATE THE STRING COMMAND ID ON THE COMMAND HISTORY STACK, BUT THE STRING WAS NOT THERE. (DUIDOREDO 5023)

CAUSE The DO or REDO command was given with a command identifier string, but no matching command could be found in the history stack.

ACTION Use the LISTREDO command to see the diagnostic system history stack. Then re-enter the command, giving a correct command identifier. Information about command identifiers may be found by using the HELP DO or HELP REDO commands.

---

**5024** *** THE DUI TRIED TO SAVE THE COMMAND HISTORY STACK, BUT THERE WAS NOTHING ON THE STACK. (DUIDOREDO 5024)

CAUSE The REDOSAVE command was given, but the diagnostic history stack could not be saved because it was empty.

ACTION If the REDOSAVE command is the first command given during the diagnostic session the history stack should be empty - nothing can or should be done to save the stack. If some commands have preceded the REDOSAVE command, then an SR should be submitted against the DUI.

Use the SETVAR TRACE ERROR command. Repeat the REDOSAVE command noting the additional errors which will appear. An easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit the SR against the DUI noting all of the errors.

---

**6006** *** INVALID PARAMETERS WERE PASSED. (DUIERR 6006)

CAUSE The MODE command was given with both the "multi" and the "single" parameters.

ACTION Repeat the MODE command using either the "multi" or the "single" parameters, but not both.

---

**6008** `*** LEVEL 0 PRIVILEGE IS NEEDED TO CHANGE THE MODE. (DUIERR 6008)`

CAUSE The MODE SINGLE command was given while the system was in multi-user mode. The user has too low a security capability (less than diag_level_0) for the request to be honored.

ACTION Log in as a user with a higher security level, or have the system administrator increase your security capabilities.

---

**6015** `*** THE SECURITY LEVEL NUMBER WAS INVALID. (DUIERR 6015)`

CAUSE The MODE SINGLE command was given by a user with too low a security level for the request to be honored. The DUI was prevented from telling the user what his/her security level was because the security level could not be translated into an ASCII printable string.

ACTION The actual cause of the failure will be printed immediately before this message; see the cause/action statement for that message. However, if that proves unsatisfactory then invoke the SETVAR TRACE ERROR command. Repeat the MODE SINGLE command noting the additional errors which will appear. An easy way to get a copy of these would be by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit the SR against the DUI noting all of the errors.

---

**801** `*** COULD NOT LAUNCH COMMAND INTERPRETER (DUIERR 801)`

CAUSE The CI command was given, but the DUI was unable to invoke the command interpreter.

ACTION Invoke the SETVAR TRACE ERROR command. Repeat the CI command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit the SR against the DUI noting all of the errors.

---

**9001** `*** ONLY TEST NUMBERS GREATER THAN ZERO ARE ALLOWED (DUICTERR 9001)`

CAUSE The CODETEST command was given with a "0" or a negative number explicitly specified as a test number to be processed. Only positive integer values greater than 0 may be used as test numbers.

ACTION Repeat the CODETEST command using correct test numbers. If a CODETEST test script contains tests whose numbers are not integer values greater than 0, correct the script.

---

**9002** `*** INVALID TEST RANGE - THE LAST NUMBER IN A RANGE MUST BE GREATER THAN THE FIRST (DUICTERR 9002)`

CAUSE — The CODETEST command was given with a test number range ( "n/n" ), but the last number in the range was smaller than the first.

ACTION — Repeat the command correcting the test range so that the second number in the range is greater than the first.

---

**9501** `*** UNABLE TO CONVERT THE SPECIFIED LDEV (!) TO ITS ASSOCIATED PDEV. (DUIERR 9501)`

CAUSE — The UNLOCK command was given with an ldev parameter, but the diagnostic system could not convert the specified ldev into its corresponding pdev. Pdevs rather than ldevs are used internally in the diagnostic system so the failure to translate the given ldev into a pdev precludes unlocking the device.

ACTION — Repeat the command using a pdev parameter rather than the ldev parameter.

---

**9502** `*** UNABLE TO UNLOCK PDEV (!). (DUIERR 9502)`

CAUSE — The UNLOCK command was given. The DUI received a garbled handshaking reply from the underlying process which should have performed the "unlock." Because the handshake was garbled, the DUI is assuming that the device was not unlocked.

ACTION — Run something which uses the device (such as its diagnostic) to see if it is really locked or not. Submit an SR against the DUI. If the device is actually still locked the SR is more urgently needed than if it is not.

Invoke the SETVAR TRACE ERROR command. Repeat the UNLOCK command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit the SR against the DUI noting all of the errors.

---

**10000** `*** THE FOLLOWING MESSAGE FROM THE DIAGNOSTIC PROGRAM'S CATALOG CONTAINS A SYNTAX ERROR (DUIERR 10000)`

CAUSE When the user enters the RUN command, the DUI obtains information concerning the sections and steps of the diagnostic from the diagnostic's message catalog file. The DUI detected a syntax error during the parsing of these section/step messages. This error message will be followed by a copy of the line from the catalog in which the syntax error was detected along with a detailed message describing the problem.

ACTION Submit an SR against the diagnostic including all the information given in this message and the two following.

The RUN command may also be redone naming the sections in some alternate way. That is, if the original command used number ranges try listing the numbers explicitly. If the original command used mnemonics, try the numbers instead. If the original command relied on the defaults (i.e. didn't specify numbers or mnemonics), explicitly name the sections/steps to be run. The diagnostic catalog section/step messages which the DUI must parse in order to determine which sections/steps to run are dependent on the form of the input used in the RUN command. By changing the form of input, it is possible that the message with the error can be avoided. An SR should still be submitted though even if this works.

---

**10001** `*** DUI WAS NOT ABLE TO READ MESSAGE 3 OF SET 2 FROM THE DIAGNOSTIC PROGRAM'S CATALOG FILE (DUIERR 10001)`

CAUSE The RUN command was invoked. The DUI was not able to read the above mentioned message from the diagnostic program's message catalog. This message contains a list of the default sections for the diagnostic and is needed to determine which sections to run if none were named with the RUN command. An error message indicating the specific problem will follow this message.

ACTION Repeat the RUN command explicitly naming the sections to be run. A list of these may be seen by using the HELP <diagnostic> SECTIONS command.

Additional action may be needed depending on the error message which will follow this one. For example, if that message indicates that the set 2, message 3 message does not exist then an SR should be entered against the diagnostic.

---

**10002** `*** DUI WAS NOT ABLE TO READ MESSAGE 2 OF SET 2 FROM THE DIAGNOSTIC PROGRAM'S CATALOG FILE (DUIERR 10002)`

CAUSE The RUN command was invoked. The DUI was not able to read the above mentioned message from the diagnostic program's message catalog. This message contains a list of the default steps for every section in the diagnostic and is needed to determine which steps to run if some steps were not explicitly named within the RUN command. An error message indicating the specific problem will follow this message.

ACTION Repeat the RUN command explicitly naming all steps to be executed. A list of these may be seen by using the HELP <diagnostic> SECTIONS command.

Additional action may be needed depending on the error message which will follow this one. For example, if that message indicates that the set 2, message 2 message does not exist, then an SR should be entered against the diagnostic.

---

**10003** `*** MESSAGE 3 OF SET 2 FROM THE DIAGNOSTIC PROGRAM'S CATALOG IS A BLANK LINE (DUIERR 10003)`

CAUSE The RUN command was invoked on a diagnostic which contains sections, but no sections were named. The DUI found that the diagnostic catalog message which should have had a list of the default sections for the diagnostic was a blank line. This is not valid. The DUI cannot determine which sections to run in this case.

ACTION Repeat the RUN command explicitly naming all the sections to be executed. A list of these may be seen by using the HELP <diagnostic> SECTIONS command.

An SR should also be entered against the diagnostic.

---

**10004** `*** MESSAGE 2 OF SET 2 FROM THE DIAGNOSTIC PROGRAM'S CATALOG IS A BLANK LINE (DUIERR 10004)`

CAUSE The DUI found that the above mentioned message was a blank line. This is not valid. The DUI is not able to validate the section and step numbers entered by the user without this information.

The RUN command was invoked on a diagnostic which contains sections and steps, but either no sections were named or some section was named without specifying steps. The DUI found that the diagnostic catalog message which should have had a list of the default steps for each section in the diagnostic was a blank line. This is not valid. The DUI cannot determine which steps to run in this case.

ACTION Repeat the RUN command explicitly naming all the sections and steps to be executed. A list of these may be seen by using the HELP <diagnostic> SECTIONS command.

An SR should also be entered against the diagnostic.

---

**11010** `*** ERROR - COULD NOT PERFORM REQUESTED OPERATION ON THE SPECIFIED DIAGNOSTIC SYSTEM PROCESS (DUISYSERR 11010)`

CAUSE 1) The DIAGSYSTEM SHOWACTIVE command was given. The proper response should have been for the DUI to print out a list of active diagnostic system processes and their program identifiers. However, one of the identifiers could not be printed because it could not be converted into an ASCII printable string.

2) Either the DIAGSYSTEM RUN <program name> or the DIAGSYSTEM ABORT <program identifier> command was given. The DUI received a handshaking reply from the underlying process which should have performed the "run" or "abort" stating that the action was not completed successfully. The actual reason for the failure will be printed immediately before this message.

ACTION If the DIAGSYSTEM SHOWACTIVE command was given invoke the SETVAR TRACE ERROR command. Repeat the DIAGSYSTEM SHOWACTIVE command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit the SR against the DUI noting all of the errors.

If the DIAGSYSTEM RUN <program name> or DIAGSYSTEM ABORT <program identifier> command was used then follow the action specified for the message which will immediately precede this one.

---

**11011** `*** ERROR - COULD NOT ABORT THE SPECIFIED DIAGNOSTIC SYSTEM PROCESS (DUISYSERR 11011)`

CAUSE The DIAGSYSTEM ABORT <program identifier> command was given. The DUI received a garbled handshaking reply from the underlying process which should have performed the "abort." Because the handshake was garbled, the DUI assumes that the process was not aborted.

ACTION Use the CI command and whatever process status command works on the current operating system to see if the process has actual been aborted. The DIAGSYSTEM SHOWACTIVE command might also work, but its results could be suspect in the current case.

Submit an SR against the DUI. If the process was not actually aborted the SR is more urgently needed than if it was.

Invoke the SETVAR TRACE ERROR command. Repeat the DIAGSYSTEM ABORT <program identifier> command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Give a complete list of the resultant errors in the SR submitted against the DUI.

---

**11012** `*** ERROR - COULD NOT RUN THE SPECIFIED DIAGNOSTIC SYSTEM PROCESS (DUISYSERR 11012)`

CAUSE: The DIAGSYSTEM RUN <program name> command was given. The DUI received a garbled handshaking reply from the underlying process which should have performed the "run." Because the handshake was garbled, the DUI assumes that the process was not launched.

ACTION: Use the CI command and whatever process status command works on the current operating system to see if the process is running. The DIAGSYSTEM SHOWACTIVE command might also work.

Submit an SR against the DUI. If the process was not actually launched the SR is more urgently needed than if it was.

Invoke the SETVAR TRACE ERROR command. Repeat the DIAGSYSTEM RUN <program name> command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Give a complete list of the resultant errors in the SR submitted against the DUI. It might be necessary to abort the diagnostic system process before repeating the DIAGSYSTEM RUN <program name> command.

---

**11013** `*** ERROR - CANNOT LOCATE THE LIST OF DIAGNOSTIC SYSTEM PROCESSES (DUISYSERR 11013)`

CAUSE: The DIAGSYSTEM SHOWACTIVE command was given. The DUI could not find the diagnostic system table which contains all the information about diagnostic system processes.

ACTION: Invoke the SETVAR TRACE ERROR command. Repeat the DIAGSYSTEM SHOWACTIVE command and note the additional errors which will appear. An easy way to get a copy of these is by using the OUTFILE and/or HARDCOPY commands - preferably before the SETVAR command is called. Submit an SR against the DUI giving a complete list of the errors.

---

# Contents

# 3

# Memory Diagnostic

## Introduction

The `MEMDIAG` diagnostic is designed to detect general memory device failures in PA-RISC Memory Array cards, PA-RISC Memory Controller cards, and in PA-RISC Memory Controller/Memory Array combination cards. The diagnostic offers a selection of high level tests that will call lower level routines to access the hardware. If an error is encountered, the Field Replaceable Unit (FRU) will be pinpointed and the test will continue to execute, if possible.

The diagnostic is divided into sections that the user can decide whether or not to run. The user can enter the numbers for the test sections to be run as parameters for the **run** command, used to invoke the diagnostic.

After the diagnostic has cycled through all the memory pages it was able to access, `MEMDIAG` will terminate, and the user will be returned to the Diagnostic User Interface (DUI).

## Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database referencing product number 30600-10009.

## Minimum Configuration

The hardware required to run `MEMDIAG` is the PA-RISC system to be tested, consisting of the memory controller card and the memory array card, or of the memory controller/memory array combination card, plus any other equipment required to get the operating system up and running.

## Operating Instructions

There is no security level check mechanism within MEMDIAG. The DUI checks the user's security level before initiating MEMDIAG. Refer to the section on the DUI for a detailed description of user capabilities.

### Default Tests

If the user does not specify any sections or steps to be run, the default sections are executed, based on the diagnostic mode which has been selected by the Online Diagnostic subsystem. (See the Online Diagnostics Overview discussion of diagnostics modes for details.) The default test is:

Section 9 Trouble Tree

### RUN Command

To bring up the Online Diagnostic subsystem or DUI, enter the following command to the HP-UX system prompt:

```
%/usr/diag/bin/sysdiag
```

The system responds with the following Diagnostic User Interface prompt:

```
DUI >
```

Typing HELP causes a summary of the DUI and its commands to be printed. Refer to the DUI Section of this manual for details.

To run the diagnostic, enter:

```
DUI >RUN MEMDIAG
```

*no parameters required to load test suite (though parameters can be given)*

The user can enter `help memdiag section` and see a menu that briefly describes the section and steps available in the diagnostic.

#### Parameters to the RUN command

At the invocation of the online diagnostics subsystem diagnostic, the user can define several run time parameters. The `loop` parameter is used to control the number of times the selected sections and steps are executed.

For further information on parameters to the run command, refer to the online diagnostics subsystem DUI ERS. All parameters available in the run command are accepted as parameters when executing this diagnostic.

## Test Execution

After the user enters the DUI run command, the diagnostic MEMDIAG will be invoked. Once the program is started, the following header and welcome message will be displayed on the screen.

```
*********************************************************************************
*****                                                                      *****
*****                                                                      *****
*****                       Memory Array Diagnostic                        *****
*****                                                                      *****
*****             (c) Copyright Hewlett-Packard Company 1987               *****
*****                        All Rights Reserved.                          *****
*****                                                                      *****
*****                        Version A.00.00                               *****
*****                                                                      *****
*****                                                                      *****
*********************************************************************************
Welcome, Today is Thursday, January 5, 1987 at 10:50:30
```

Messages will be sent to the screen informing the user of the current test being executed and, upon its completion, the results. If the user did not specify any sections or steps, the default sections and steps will be executed. In the case of MEMDIAG, the default section is the Trouble Tree (section 9). Each section has default steps set that will be run if the section is selected without specifying any steps. These default steps are listed under their respective sections in this document.

Whenever MEMDIAG prompts the user for input, the user may enter "exit" to terminate the MEMDIAG diagnostic and jump back into the DUI system environment, or enter "suspend" to temporarily suspend MEMDIAG execution. Either the entire word or any number of characters sufficient to uniquely identify the command may be entered.

The user may then perform tasks within the Diagnostic User Interface (DUI) and later resume/abort execution of MEMDIAG.

## Early Termination

MEMDIAG runs extensive pattern tests that usually take more than a half hour to complete. Because of this, several options are available for exiting the testing sequence before it terminates in a normal fashion.

1. If time is a critical factor in your testing, go into Section 8, User Interactive, and select specific memory ranges to test. This is a more direct approach, and is much more time-efficient: you can test memory pages mapping to a specific FRU.
2. Before any test is invoked, the following prompt will be displayed to ensure that the user is aware of the time these tests take to complete. The user can then choose to continue testing or not:

```
***
***      WARNING
***
***  The section you have selected will run memory tests that
***  take more than half hour to complete (on average).
***
***  If you don't have the time now, go into USER INTERACTIVE (Section 8)
***  and setup your own memory test by selecting parameters that will
***  specifically test certain ranges and, therefore, shorten test duration.
***
***
***  Do you want to continue with this Section now? (y/n) [y]:
***
```

3. The user may enter **exit** or **suspend** at any MEMDIAG prompt; this will either terminate or temporarily suspend execution of MEMDIAG.
4. If a test is in progress, and the user would like to interrupt its execution, a system-dependent sequence of characters can be entered to interrupt the normal execution cycle of MEMDIAG. The interrupt sequences are <CNTL-C> for HP-UX and <CNTL-Y> for MPE XL.

## Detailed Test Descriptions

The following is a detailed discussion of `MEMDIAG` sections and steps. While in the Diagnostic User Interface, the user may view a menu of sections/steps by using the **`help memdiag section`** command. The menu gives a brief summary of each section and step. The following table lists the sections/steps of this diagnostic:

```
                               MEMDIAG Menu

   Section 1 :          Initialize Card

       step 10 :            initialize memory controller(s)
       step 11 :            memory to all ones
       step 12 :            memory to all zeros


   Section 2:          Identify

       step 20 :           Configuration Information
       step 21 :            Identify Information


   Section 3:          Status


   Section 4:          Memory tests (1 Second Wait per Page)

       step 40 :            address uniqueness/complement
       step 41 :            walking ones/zeros pattern
       step 42 :            alternating ones/zeros pattern
       step 43 :            ALL section 4 patterns


   Section 5:          Memory tests (5 Second Wait per Page)

       step 50 :            address uniqueness/complement
       step 51 :            walking ones/zeros pattern
       step 52 :            checkerboard/complement pattern
       step 53 :            ALL section 5 patterns


   Section 6:          Verify/Sweeper\(Not implemented yet)


   Section 7:          EDC Logic test\(Not implemented yet)


   Section 8:          USER INTERACTIVE


   Section 9:          Trouble Tree (Default Test)
```

## Section 1 : INITIALIZE CARD

This section will initialize the memory space to all ones or all zeros and clear the syndrome register(s) of all memory controller(s) in the system. Step 12 will write all zeros to memory then read it back and verify proper storage. Step 11 will write all ones to memory then read it back and verify proper storage. Step 10 will clear the memory controller syndrome registers. This section has three steps, as shown below. The user can select any combination of steps.

```
Step 10 :        Initialize memory controller(s)

Step 11 :        Initialize memory to all 1's

Step 12 :        Initialize memory to all 0's
```

**POSSIBLE OUTPUT MESSAGES:**

```
Section 1  :   INITIALIZE CARD

      STEP 12 - Initialize memory to ZEROS
------------------------------------------------------------------------------
      The following PAGE RANGES are being tested            ALL ZEROS
------------------------------------------------------------------------------
      Start - End       Start - End        Start - End       Start - End

      2598 - 2603       2608 - 2613        2618 - 2622       2625 - 2636

      2638 - 2653       2655 - 2720        2722 - 3015       3018 - 3064

      3066 - 3212       3214 - 3344        3346 - 3425       3430 - 3504

      3506 - 3522       3524 - 3527        3529 - 3555       3557 - 3562

      3564 - 3568       3570 - 3593        3596 - 3625       3627 - 3634

                               ...etc...
```

**DEFAULT:**

Steps 10 and 12.

## Section 2 : IDENTIFY

The Identify section displays configuration and model revision information about the memory system currently being tested. This section has two steps:

| | |
|---|---|
| Step 20 | Configuration Information |
| Step 21 | Model Revision Information (Identify) |

The Memory Controller table shown below is from a system where all memory is located onboard the memory controller cards only (there are no associated memory array cards).

**SAMPLE OUTPUT MESSAGE 1, when memory interleave is off:**

```
Section 2   : IDENTIFY

        STEP 20 - Configuration Information
```

| **Memory Controller #** | | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| Slot Number | = | 3 | 4 | 5 | 7 | 9 |
| Hard Physical Address | = | fff8c000 | fff90000 | fff94000 | fff9c000 | fffa4000 |
| Associated MA Cards | = | 0 | 0 | 0 | 0 | 0 |
| MC chip size (in bits) | = | 1048576 | 1048576 | 1048576 | 1048576 | 1048576 |
| MC memory size (in Mbytes) | = | 8 | 8 | 8 | 8 | 8 |
| Start Page | = | 0 | 4096 | 8192 | 12288 | 16384 |
| End Page | = | 4095 | 8191 | 12287 | 16383 | 20479 |
| Page Range | = | 4096 | 4096 | 4096 | 4096 | 4096 |

**SAMPLE OUTPUT MESSAGE 2, when memory interleave is off:**

```
Section 2  : IDENTIFY

        STEP 20 - Configuration Information
```

| Memory Controller # | | 1 | 2 |
|---|---|---|---|
| Slot Number | = | 0 | 1 |
| Hard Physical Address | = | fff80000 | fff81000 |
| Associated MA cards | = | 3 | 1 |

```
>>>MEMORY ARRAY cards under control of MC in Slot # 0
>>>WARNING: If a memory array is disabled, then the corresponding start page
and the corresponding end page will be -1.

If a memory controller is disabled, then all the memory arrays
under this memory controller are also disabled.
```

| Memory Array # | | 1 | 2 | 3 |
|---|---|---|---|---|
| Slot Number | = | 0 | 1 | 2 |
| MA chip size (in bits) | = | 1048576 | 1048576 | 1048576 |
| MA memory size (in Mbytes) | = | 16 | 16 | 16 |
| Start Page | = | 0 | -1 | 8192 |
| End Page | = | 8191 | -1 | 16383 |
| Page Range | = | 8192 | 0 | 8192 |

```
>>>MEMORY ARRAY cards under control of MC in Slot # 1
>>>WARNING: If a memory array is disabled, then the corresponding start page
and the corresponding end page will be -1.

If a memory controller is disabled, then all the memory arrays
under this memory controller are also disabled.
```

| Memory Array # | | 1 |
|---|---|---|
| Slot Number | = | 0 |
| MA chip size (in bits) | = | 1048576 |
| MA memory size (in Mbytes) | = | 16 |
| Start Page | = | 16384 |
| End Page | = | 24575 |
| Page Range | = | 8192 |

**SAMPLE OUTPUT MESSAGE 3, when memory interleave is on:**

```
Section 2  : IDENTIFY

      STEP 20 - Configuration Information
```

| Memory Controller # | | 1 | 2 |
|---|---|---|---|
| Slot Number | = | 0 | 1 |
| Hard Physical Address | = | fff80000 | fff81000 |
| Associated MA cards | = | 3 | 2 |

```
>>>MEMORY ARRAY cards under control of MC in Slot # 0
>>>WARNING: If a memory array is disabled, then the corresponding start page
and the corresponding end page will be -1.
If a memory controller is disabled, then all the memory arrays
under this memory controller are also disabled.
```

| Memory Array # | | 1 | 2 | 3 |
|---|---|---|---|---|
| Slot Number | = | 0 | 1 | 2 |
| MA chip size (in bits) | = | 1048576 | 1048576 | 1048576 |
| MA memory size (in Mbytes) | = | 16 | 16 | 16 |
| MA status | = | Enabled | Disabled | Enabled |
| Memory Interleave is ON. | | | | |

```
>>>MEMORY ARRAY cards under control of MC in Slot # 1
>>>WARNING: If a memory array is disabled, then the corresponding start page
and the corresponding end page will be -1.
If a memory controller is disabled, then all the memory arrays
under this memory controller are also disabled.
```

| Memory Array # | | 1 | 2 |
|---|---|---|---|
| Slot Number | = | 0 | 2 |
| MA chip size (in bits) | = | 1048576 | 1048576 |
| MA memory size (in Mbytes) | = | 16 | 16 |
| MA status | = | Enabled | Enabled |
| Memory Interleave is ON. | | | |

The Memory Controller table shown below is from a system where the memory controller is on the mother board, and memory array cards, which are installed in pairs in a slot, are also on the mother board.

**SAMPLE OUTPUT MESSAGE 4, MC/MA on the mother board:**

```
Section 2  :     IDENTIFY


STEP 20 - Configuration Information
```

| **Memory Controller # on the mother board slot** | | | |
|---|---|---|---|
| Slot Number | = | 3 | 1 |
| Hard Physical Address | = | 6 | 1 |
| Associated Bank Number | = | 0 | 0 |
| Associated SPA | = | 0 | 16777216 |
| Associated MA cards | = | 2 | 2 |
| Start Page | = | 0 | 4096 |
| End Page | = | 4095 | 8191 |
| Page Range | = | 4096 | 4096 |

```
>>> MEMORY ARRAY cards under control of MC in Slot # 3

>>> WARNING:  If a memory array is disabled, then the corresponding start page
              and the corresponding end page will be -1.
              If a memory controller is disabled, then all the memory arrays
              under this memory controller are also disabled.
```

| **Memory Array #** | | 1 | 2 |
|---|---|---|---|
| Slot Number | = | A | B |
| MA chip size (in bits) | = | 4194304 | 4194304 |
| MA memory size (in Mbytes) | = | 8 | 8 |
| MA status | = | Enabled | Enabled |

```
>>> MEMORY ARRAY cards under control of MC in Slot # 1

>>> WARNING:  If a memory array is disabled, then the corresponding start page
              and the corresponding end page will be -1.
              If a memory controller is disabled, then all the memory arrays
              under this memory controller are also disabled.
```

| **Memory Array #** | | 1 | 2 |
|---|---|---|---|
| Slot Number | = | A | B |
| MA chip size (in bits) | = | 4194304 | 4194304 |
| MA memory size (in Mbytes) | = | 8 | 8 |
| MA status | = | Enabled | Enabled |

Finally, Step 21 will print out an Identify Table for each memory controller in the memory system, this will contain model/revision information for the system, as well as the SPA associated with the memory array card.

**SAMPLE OUTPUT MESSAGE 1:**

```
STEP 21 - Identify Information
```

| **IDENTIFY Information for Memory Controller in Slot #0** | | |
|---|---|---|
| Hardware Model | = | 4 |
| Hardware Revision | = | 4 |
| Software Model | = | 9 |
| Software Revision | = | 0 |
| Software Option | = | 0 |
| Soft Physical Address | = | 0 |

**SAMPLE OUTPUT MESSAGE 2, MC/MA on the mother board:**

```
STEP 21 - Identify Information
```

| **IDENTIFY Information for Memory Controller in Slot # on the mother board** | | |
|---|---|---|
| Hardware Model | = | 0 |
| Hardware Revision | = | 1 |
| Software Model | = | 0 |
| Software Revision | = | 0 |
| Software Option | = | 0 |
| Soft Physical Address | = | 0 |

**DEFAULT:** Steps 20 and 21.

## Section 3 : STATUS

This section will return status information from *all* the Memory Controller status register(s) associated with the memory system to be tested. This means the error flag conditions reported in the syndrome registers will be displayed. An example is shown below of a system with two memory controllers.

**SAMPLE OUTPUT MESSAGE 1:**

```
  SECTION 3 - STATUS


---------------------------------------------------------
           Memory Controller #1            Slot A
---------------------------------------------------------
     >>> NO Memory Errors registered.
---------------------------------------------------------
           Memory Controller #2            Slot 8
---------------------------------------------------------
     >>> SINGLE BIT ERROR due to CHECK bits




  END Section 3
```

**SAMPLE OUTPUT MESSAGE 2, MC/MA on the mother board:**

```
  SECTION 3 - STATUS


---------------------------------------------------------
         Memory Controller #  on the mother board  slot
---------------------------------------------------------
   >>> NO Memory Errors registered.



  END Section 3
```

**DEFAULT:**

This section has no steps.

## Section 4 : MEMORY TEST (1 Second Wait per BUSY Page)

This section is a standard selection of pattern tests to exercise memory, checking for basic functionality. It is broken down into three steps, where each step is actually running two tests: the specified pattern and its complement pattern. There is also a fourth step that will batch the patterns into one test.

If no steps are requested in the run command, this section will default to run all six tests. The patterns in this section will also be run in the Trouble Tree section.

The difference between running all tests by selecting step 43, or running all tests by selecting the section (which defaults to run steps 40, 41, and 42) is as follows. Running the Section will sweep memory from start to end six times, once for each pattern used. Running Step 43 will sweep memory from start to end only once, writing and reading back all six patterns in the same cycle. This means that:

- The time that it takes to run step 43 is less than the time that it takes to run the section, because step 43 allocates a page only once, instead of six times.
- Running step 43 may not cover as many busy pages as running the section.

The diagnostic will cycle through memory requesting memory pages as it goes; if a page is busy, this section will wait 1 second for that page to free up before it proceeds on to request the next page.

The tests are explained below.

Step 40 — **Address Uniqueness/Complement test:** This test will write the address value of a memory location into itself. For example, the value 12 will be stored in memory address location 12. This test will verify that every storage cell exists as a separate and unique entity. Once this is verified, the complement of this test will be performed, where the complement of each address will be written into each address location.

Step 41 — **Walking Ones/Zeros pattern test:** This test writes a walking one pattern into memory, then it is read back and verified. The diagnostic will then write and verify a complementary walking zero pattern. The Walking Ones pattern test will cycle through memory writing 32 patterns (each of 32 bits) into consecutive memory locations.

The patterns cycled through are as follows:

```
1000 0000 0000 0000 0000 0000 0000 0000
0100 0000 0000 0000 0000 0000 0000 0000

0010 0000 0000 0000 0000 0000 0000 0000
0001 0000 0000 0000 0000 0000 0000 0000

              . . . etc . . .

0000 0000 0000 0000 0000 0000 0000 1000
0000 0000 0000 0000 0000 0000 0000 0100

0000 0000 0000 0000 0000 0000 0000 0010
0000 0000 0000 0000 0000 0000 0000 0001
```

Step 42 **Alternating Ones/Zeros pattern test:** This test writes an Alternating Ones pattern into memory and then reads it back and verifies memory contents. Again, the complementary pattern of Alternating Zeros is written and read back upon successful completion of the initial pattern.

The Alternating Ones pattern is as follows :

```
1010 1010 1010 1010 1010 1010 1010 1010
```

Step 43 **All Patterns Section 4:** Starting at Page 0, request pages until end of memory is reached. For each page obtained, write and read back *all* patterns from Section 4 into this page.

**POSSIBLE OUTPUT MESSAGES:**

```
Section 4  :   MEMORY TEST   (1 second wait per BUSY page)

        STEP 40 - Address Uniqueness/Complement Test

------------------------------------------------------------------------------------

     The following PAGE RANGES are being tested          ADDRESS UNIQUENESS

------------------------------------------------------------------------------------

     Start - End       Start - End       Start - End       Start - End

     2598 - 2603       2608 - 2613       2618 - 2622       2625 - 2636

     2638 - 2653       2655 - 2720       2722 - 3015       3018 - 3064

     3066 - 3212       3214 - 3344       3346 - 3425       3430 - 3504

     3506 - 3522       3524 - 3527       3529 - 3555       3557 - 3562

     3564 - 3568       3570 - 3593       3596 - 3625       3627 - 3634

     3636 - 3673       3570 - 3593       3596 - 3625       3627 - 3634

                               ...etc...

  END Section 4
```

**DEFAULT:**

Steps 40, 41, and 42.

## Section 5 : MEMORY TEST (5 Second Wait per BUSY Page)

This section is broken down into three steps, where each step is actually running two tests: the specified pattern and its complement pattern. It will default to run all six tests, if no steps are requested in the `run` command. There is also a fourth step that will batch all the patterns into a single test. This section will *not* be called from the Trouble Tree section.

The difference between running all tests by selecting Step 53, or running all tests by selecting the section (which defaults to run steps 50, 51, and 52) is as follows. Running the Section will sweep memory from start to end six times, once for each pattern used. Running Step 53 will sweep memory from start to end only once, writing and reading back all six patterns in the same cycle. This means:

- The time that it takes to run step 53 is less than the time that it takes to run the section, because step 53 allocates a page only once, instead of six times.
- Running step 53 may not cover as many busy pages as running the section.

The diagnostic will cycle through memory requesting memory pages as it goes; if a page is busy, this section will wait 5 seconds for that page to free up before it proceeds on to request the next page.

Step 50 — **Address Uniqueness/Complement test:** This test will write the address value of a memory location into itself. For example the value 12 will be stored in memory address location 12. This test will verify that every storage cell exists as a separate and unique entity. Once this is verified, the complement of this test will be performed, where the complement of each address will be written into each address location.

Step 51 — **Walking Ones/Zeros pattern test:** This test writes a walking one pattern into memory, then it is read back and verified. The diagnostic will then write and verify a complementary walking zero pattern. The Walking Ones pattern test will cycle through memory writing 32 patterns (each of 32 bits) into consecutive memory locations. The patterns cycled through are as follows:

```
1000 0000 0000 0000 0000 0000 0000 0000
0100 0000 0000 0000 0000 0000 0000 0000

0010 0000 0000 0000 0000 0000 0000 0000
0001 0000 0000 0000 0000 0000 0000 0000

              . . . etc . . .

0000 0000 0000 0000 0000 0000 0000 1000
0000 0000 0000 0000 0000 0000 0000 0100

0000 0000 0000 0000 0000 0000 0000 0010
0000 0000 0000 0000 0000 0000 0000 0001
```

The diagnostic will then write and verify a complementary walking zero pattern.

Step 52 **Checkerboard pattern/Complement test:** This test will write a checkerboard pattern to the memory space. Afterwards it will read back the memory and verify that the data was stored correctly; then the complement test will be done. The pattern written is the Alternating Ones pattern followed by the complement of this pattern. After each memory address location is written, the pattern is toggled and so on.

Step 53 **All patterns Section 5:** Starting at Page 0, request pages until end of memory is reached. For each page obtained, write and read back *all* patterns in Section 5 into this page.

**POSSIBLE OUTPUT MESSAGES:**

```
   Section 5  :   MEMORY TEST  (5 second wait per BUSY page)

        STEP 50 - Address Uniqueness/Complement Test

-----------------------------------------------------------------------------------------------------------------

     The following PAGE RANGES are being tested          ADDRESS UNIQUENESS

-----------------------------------------------------------------------------------------------------------------

     Start - End       Start - End       Start - End       Start - End

     2598 - 2603       2608 - 2613       2618 - 2622       2625 - 2636

     2638 - 2653       2655 - 2720       2722 - 3015       3018 - 3064

     3066 - 3212       3214 - 3344       3346 - 3425       3430 - 3504

     3506 - 3522       3524 - 3527       3529 - 3555       3557 - 3562

     3564 - 3568       3570 - 3593       3596 - 3625       3627 - 3634

     3636 - 3673       3570 - 3593       3596 - 3625       3627 - 3634

     3636 - 3673       3676 - 3679       3681 - 3770       3772 - 3776

                            ...etc...

   END Section 5
```

**DEFAULT:**

Steps 50, 51, and 52.

## Section 6 : SWEEPER FUNCTION (Unimplemented)

This section runs a basic sweeping function to read all memory locations. If any errors are found, those errors will be logged in an error log for future correction.

A sweeper function will *sweep* through memory from beginning to end and read all memory locations that it can obtain access to. By making a read of a memory location the data from that location will pass through the EDC logic and if there is a SBE error within the data read it will show up in the associated memory controllers status word. The sweeper function will read a memory range and then check the status register to see if an error occurred. If an error is flagged, the sweeper function will log the location where it occurred, clear the error flag and continue the sweep. This function is important because it can help monitor the level of soft errors occurring in memory, this can help reduce the probability of HPMCs (machine crashs) by allowing the user to see possible problem areas in memory. If an area is showing repetitive occurrence of soft errors, then possibly this card should be changed. The type of error, address location, and associated memory controller will be logged in a file for monitoring of soft error levels as a preventative maintenance measure. The diagnostic will display the message shown below at the completion of the sweeping function. The errors logged will be broken down into the categories of *buffer errors* and *card errors*.

This section is currently unimplemented and will be implemented dependent upon Hardware and Operating System changes that are needed to mask any HPMC that could occur as a result of this test. The reason this may cause HPMCs is that by sweeping through memory and reading each location you will be touching areas of memory on a regular basis. If you read a memory location and there is a hard error there (a stuck bit) and a transient error occures during the read (say a glitch) you will have a double bit error which will cause an HPMC on the system. So by regular sweeps through memory you are increasing the probability of causing HPMCs.

POSSIBLE OUTPUT MESSAGES:

```
Section 6  :   SWEEPER FUNCTION TEST

Sweeper function has completed.

TOTAL ERRORS logged ==>> 5
Card   errors :  5
Buffer errors :  0
END Section 6
```

DEFAULT:

```
This section has no steps
```

## Section 7 : EDC LOGIC TEST (Unimplemented)

This section runs an error logic test sequence to verify functionality of the EDC logic on board the memory card(s). This test will write false data into the Syndrome Register usually by writing a force_error register that is architected in the system. This force_error register will cause the EDC hardware to flag a data error because the Syndrome will not match up to the data read. The data finally put out on the bus will be corrected by the EDC logic; but it is corrected only *on-the-fly* and not in the actual memory location. This means that the EDC logic can detect a SBE and correct that single bit value before putting it out onto the bus; however, the value stored in memory is still incorrect.

The force_error register name varies from system to system and some may not even have this feature designed in. If that is the case, the diagnostic will print a message stating that the test cannot be performed because it is not functionally supported.

This section is currently unimplemented and will be implemented dependent upon Hardware and Operating System changes that are needed to mask any HPMC that could occur as a result of this test.

**POSSIBLE OUTPUT MESSAGES:**

```
Section 7  :   EDC LOGIC TEST

EDC Logic Test completed successfully.

END Section 7
```

**DEFAULT:**

```
This section has no steps
```

## Section 8 : USER INTERACTIVE

This section will present the user with a menu presenting different areas requiring user interaction. The user will input the area they wish to proceed in. They will then be prompted for whatever parameters are needed for that area to be performed. This section will be menu driven and grow with the functionality needs of each PA-RISC system that is added to the diagnostic repertoire.

The main menu is explained below along with the prompts that will be displayed with each user selection.

POSSIBLE OUTPUT MESSAGES:

```
   Section 8  :   USER INTERACTIVE

================================================================================
                              MEMDIAG MENU
================================================================================

 1.  Memory Test  (using parameter settings)
 2.  EDC Test     (using parameter settings)
 3.  Sweep Memory (using parameter settings)
 4.  Set MEMDIAG parameters
 5.  View MEMDIAG parameters
 6.  View Configuration Information
 7.  View Page Status Lists
 8.  View Page Status Summary
 9.  View Memory Status (over Range)
10.  HELP Menu
11.  Exit to DUI

Input number >>>
```

If users select *1* from the Main Menu (*Memory Test*), the test will write the selected pattern(s) to all available memory locations, beginning at the starting page and ending at (starting page + page range - 1). This selection will obtain the pages within the page range specified by the user before the test is done, and then will return ownership to the system after the test is performed.

The memory test will run until the End of Range (User Input) or End of Memory is encountered, whichever comes first.

**SAMPLE OUTPUT MESSAGE:**

```
   Section 8 : MEMORY TEST

-------------------------------------------------------------------------------

        The following PAGE RANGES are being tested            ALL PATTERNS

-------------------------------------------------------------------------------

      Start - End        Start - End         Start - End        Start - End

      2598 - 2603        2608 - 2613         2618 - 2622        2625 - 2636

      2638 - 2653        2655 - 2720         2722 - 3015        3018 - 3064

      3066 - 3212        3214 - 3344         3346 - 3425        3430 - 3504

      3506 - 3522        3524 - 3527         3529 - 3555        3557 - 3562

      3564 - 3568        3570 - 3593         3596 - 3625        3627 - 3634

      3636 - 3673        3570 - 3593         3596 - 3625        3627 - 3634

      3636 - 3673        3676 - 3679         3681 - 3770        3772 - 3776

                                  ...etc...
```

Remember that you might not be able to obtain all the pages you requested. Many pages are permanently reserved by the operating system.

To get a general idea of the size and location of your own operating system, you can go into selection 7 ( *View Page Status Lists*) and view all Reserved Pages.

If users select *2* from the Main Menu (*EDC Test Memory Range*), this function will test EDC (Error Detection and Correction) logic corresponding to the memory locations falling between the starting page and (starting page + page range - 1).

When the data is written to a memory location, it will pass through EDC logic, where a Syndrome word is encoded (7 to 9 bits long) and stored in an EDC RAM. This Syndrome word will be read whenever its associated memory location is read. Both the data word and the Syndrome word will pass back through EDC logic where the Syndrome word will be decoded.

This decoded Syndrome word and the actual data word read from the memory location will be compared with each other. If they are not equal, an error exists. The PA-RISC architecture has special registers that can be written to change the correct Syndrome word to a false one. This will test the EDC logic to see if it detects this error.

This selection will obtain the pages within the page range specified by the user before the test is done, and then returns ownership to the system after the test is performed. Remember that you might not be able to obtain all the pages you requested. Many pages are permanently reserved by the operating system.

If users select *3* from the Main Menu ( *Sweep Memory Range* ), the function will read all available memory locations, beginning at the starting page and ending at (starting page + page range - 1). The memory contents will scroll on screen. Every eighth memory address will display above a line containing the contents of that address, and the next seven memory locations after that address. The output will be presented in a hexadecimal format for a more compact display. A sample is shown below:

```
PAGE 0
------------------------------------------------------------------------------------------------

Starting address for this row >>     0x 0000 0000

00000000  00000001  00000002  00000003  00000004  00000005  00000006  00000007
00000008  00000009  0000000A  0000000B  0000000C  0000000D  0000000E  0000000F
00000020  00000021  00000022  00000023  00000024  00000025  00000026  00000027
00000028  00000029  0000002A  000000A3  000000A4  000000A5  000000A6  000000A7

Starting address for this row >>     0x 0000 00A8

000000A8  000000A9  000000AA  000000AB  000000AC  000000AD  000000AE  000000AF
000000B0  000000B1  000000B2  000000B3  000000B4  000000B5  000000B6  000000B7
000000B8  000000B9  000000BA  000000BB  000000BC  000000BD  000000BE  000000BF
000000C0  000000C1  000000C2  000000C3  000000C4  000000C5  000000C6  000000C7

                         ... etc ...
```

If the user selects *4* from the Main Menu (*Set MEMDIAG Parameters*), the option to set a variety of `MEMDIAG` parameters is presented, which permits the user to tailor the diagnostic to more closely conform to the environment in which it will be running:

```
   Set MEMDIAG Parameters

-------------------------------------------------------------------------------

       1. Launch Memory Test NOW
       2. View Current Parameter Settings
       3. Change PATTERN parameter
       4. Change START PAGE parameter
       5. Change PAGE RANGE parameter
       6. Change PAGE WAIT parameter
       7. Change ALLOC SIZE parameter
       8. Change LOOP COUNT parameter
       9. Change REPEAT PROMPT? parameter
      10. Exit to Main Menu
      11. Exit to DUI

   <CR> STEP THROUGH ALL PARAMETER CHOICES

-------------------------------------------------------------------------------
```

If the user wants to change all the parameters, he should hit <CR>, which will walk him through each parameter, displaying its current setting and also the possible input to choose from. If only one (or a few) parameters are to change, select the number corresponding to that parameter(s).

The following list explains the *Set Memdiag Parameter Menu* options:

**1. Launch Memory Test NOW**
This selection will launch the memory test with current parameter settings.

**2. View Current Parameter Settings**
This selection will display the Memdiag Parameter table.

**3. Change PATTERN Parameter**

The user will be presented with a Pattern Menu of 13 choices, The pattern selected will be used when running *Memory Test (#1)*.

```
================================================================================
                       MEMDIAG PATTERN MENU
================================================================================

                         1. All Ones
                         2. All Zeros
                         3. Alternating Ones
                         4. Alternating Zeros
                         5. Walking One
                         6. Walking Zero
                         7. Checkerboard
                         8. Checkerboard complement
                         9. Address Uniqueness
                        10. Address Uniqueness complement
                        11. ALL straight patterns
                        12. ALL complement patterns
                        13. ALL patterns listed above
                       <CR> Keep current  setting


                    Input number of pattern desired >>
```

Pattern reading and writing is the time-intensive part of memory testing. By increasing the number of test patterns used, you increase the time it takes the test to complete. It is recommended that you test system performance with ONE pattern, before trying to batch patterns in a memory test. The following warning will print out before you are presented with the pattern menu shown above:

```
***
***     WARNING                           ( MEMWARN 10285 )
***
***     Selecting groups of patterns in the User Interactive section
***     will cause CPU intensive inner loops and system performance
***     will be reduced.
***
***     If you have not already done so, run with ONE pattern to test
***     out performance degradation before trying multiple pattern tests.
***
```

**4. Change START PAGE parameter** The user will be prompted for a test Starting Page. Memory page mappings will be displayed to allow the user to test a specific memory controller or memory array card.

A sample output is shown below:

```
------------------------------------------------------------------------------------------

   Memory Cont/Array in Slot 7 PAGES       0 to  4095      RANGE = 4096

   Memory Cont/Array in Slot 8 PAGES    4096 to  8191      RANGE = 4096

   Memory Cont/Array in Slot 9 PAGES    8192 to 12287      RANGE = 4096

------------------------------------------------------------------------------------------

   Currently = 0

   Input STARTING PAGE or <CR>to keep current setting >>>
```

**5. Change PAGE RANGE parameter** The user will be prompted for a test Page Range. Memory page mappings will be displayed to allow for the user to test a specific memory controller or memory array card.

A sample output is shown below:

```
------------------------------------------------------------------------------------------

   Memory Cont/Array in Slot 7 PAGES       0 to  4095      RANGE = 4096

   Memory Cont/Array in Slot 8 PAGES    4096 to  8191      RANGE = 4096

   Memory Cont/Array in Slot 9 PAGES    8192 to 12287      RANGE = 4096

------------------------------------------------------------------------------------------

   Currently = 8192

   Input PAGE RANGE or <CR>to keep current setting >>>
```

**6. Change PAGE WAIT parameter** This is the number of seconds that the program will wait on a BUSY page while trying to obtain access to it, before continuing on to request the next virtual memory page.

**7. Change ALLOC SIZE parameter**

The normal memory tests will cycle through memory requesting ONE page at a time. This is due to an MPE operating system limitation. If you are on HP-UX or in Single User Mode, you will probably want to increase this parameter and run tests from within the User Interactive Section.

---

**Warning**

**The MPE XL operating system does not have clean-up routines if MEMDIAG were to fail, meaning if you set ALLOC SIZE = 2048 and MEMDIAG is granted ownership of these pages and then dies, these pages are lost to the system. This is dangerous. The HPUX operating system is able to recover these pages, therefore, this is not a problem for HPUX. It is recommended that you use ONE page for ALLOC SIZE when running MEMDIAG on an MPE system.**

---

**8. Change LOOP COUNT parameter**

The user will input a TEST loop counter for repeat test cycles. Loop Count will be preempted by the Repeat Prompt query.

**9. Change REPEAT TEST PROMPT? parameter**

If ON, prompt the user at the end of the test cycle to repeat last test. If OFF, continue normal diagnostic flow of execution.

The Repeat Prompt Query will take precedence over Loop Count.

All user input parameter values will ONLY affect the TESTS run from WITHIN the User Interactive Section of MEMDIAG (Section 8). Specifically, the test selections affected are selections #1 (*Memory Test*), #2 (*EDC Test*), and #3 (*Memory Sweep*). The User Parameter Table does NOT affect other menu selections.

If the user selects *5* ( *View MEMDIAG Parameters*) from the Main Menu, a menu similar to the following will be displayed:

| USER PARAMETERS | Selected | Valid Range | Default |
|---|---|---|---|
| PATTERN(S) | All Ones | 1 to 13 | ALL ONES |
| START PAGE | 100 | 0 to 20479 | 0 |
| PAGE RANGE | 3000 | 1 to 20480 | 20480 |
| PAGE WAIT | 5 | 0 to 10 | 0 |
| ALLOC SIZE | 1024 | 1 to 2048 | 1 |
| LOOP COUNT | 2 | 1 to 10000 | 1 |
| REPEAT PROMPT? | ON | ON/OFF | OFF |

Typing a carriage return will leave parameters at the current setting.

If the user selects *6* ( *View Configuration Information*) from the Main Menu, ALL configuration information about any Memory Controller and/or Memory Array cards in the system will be displayed. The following is a sample display:

```
------------------------------------------------------------------------------------------

  Memory Controller #         1         2         3         4          5

------------------------------------------------------------------------------------------

  Slot Number        =        4         8         7         6          5

  Hard Physical      =  fff90000  fffa0000  fff9c000  fff98000   fff94000
  Address

  Associated         =        0         0         0         0          0
    MA Cards

  MC chip size       =   1048576   1048576   1048576   1048576    1048576
   (in bits)

  MC memory size     =        8         8         8         8          8
   (in Mbytes)

  Start Page         =        0      4096      8192     12288      16384

  End Page           =     4095      8191     12287     16383      20479

  Page Range         =     4096      4096      4096      4096       4096
------------------------------------------------------------------------------------------
```

If the user selects 7 ( *View Page Listings*) from the Main Menu, a list of the pages in the system that are status pages selected by the user will be printed.

```
========================================================================
                        PAGE STATUS MENU
========================================================================

                         1.  View Reserved   Pages
                         2.  View Available  Pages
                         3.  View Busy       Pages
                         4.  View Hole       Pages
                         5.  View Locked     Pages
                         6.  View Bad        Pages
                         7.  View Bad/Locked Pages
                         8.  View Page Status Summary
                         9.  Exit to Main Menu
                        10.  Exit to DUI

                     Input number >>>
```

A sample Reserved Page List follows:

```
--------------------------------------------------------------------------
                        RESERVED PAGE LIST
--------------------------------------------------------------------------

0     to 3180      4000 to 4200         4210 to 4211        4224 to 5000
5001  to 5009      6001 to 6001         8010 to 8020        9111 to 9111
9200  to 9201      9990 to 9999        10010 to 10100      12000 to 12001
13433 to 13435    13500 to 13555       13599 to 13600      13701 to 13701
                              ... etc ...
--------------------------------------------------------------------------
                   Reserved pages =   4000

                   TOTAL PAGES    =   20479
--------------------------------------------------------------------------
```

Page Status Types are defined under Main Menu selection *9 (View Memory Status).*

If the user selects *8* (*View Memory Status Summary*) from the Main Menu, a synopsis of the ALL System Page Status will be printed out. A sample output follows:

```
-----------------------------------------------------------------------
                      STATUS SUMMARY SNAPSHOT
-----------------------------------------------------------------------
                         Reserved pages    =   3761
                         Available pages   =  16716
                         Busy pages        =      2
                         Locked pages      =      0
                         Bad pages         =      0
                         Bad/Locked pages  =      0
                         Hole pages        =      0
-----------------------------------------------------------------------
                         TOTAL PAGES       =  20479
```

The Reserved pages consist primarily of operating system space. Some pages may stay constantly busy because they are used for system or program stacks. All these page assignments are dynamic. Even the Reserved pages total may vary to a small extent between different Status Summary Snapshots.

Page Status Types are defined under Main Menu selection *9 (View Memory Status)*.

If the user selects *9* ( *View Memory Status Over Range*) from the Main Menu, a starting page and a page range will be prompted for. The Page and Range values input for the Memory Status selection DO NOT change those parameters in the TEST parameter table (set in Main Menu selection #4). This function will check status on all the pages requested, and will present a snapshot of the memory at the time of the request.

The status of the pages will be scrolled on the screen. The output will have a starting page number, followed by the status of that page and of the following seven pages. Viewing memory range does not require that access be obtained to the page in order to view its status; therefore, you may view any page status within the operating system valid range of pages.

The majority of page status types will be Available, Busy, and Reserved. The other four page types are fairly uncommon, and are used for special cases within an operating system.

A sample is presented below:

```
--------------------------------------------------------------------------
(ROW)
Start                     PAGE STATUS
Page
--------------------------------------------------------------------------

0902     | Reserv  Reserv  Reserv  Reserv  Reserv  Reserv  Reserv  Reserv
0911     | Busy    Busy    Busy    Busy    Busy    Busy    Busy    Avail
0920     | Busy    Busy    Avail   Reserv  Reserv  Avail   Busy    Avail
0929     | Reserv  Avail   Avail   Avail   Avail   Avail   Reserv  Avail
0938     | Reserv  Avail   Avail   Avail   Avail   Avail   Reserv  Avail
0947     | Reserv  Avail   Avail   Avail   Avail   Avail   Reserv  Avail
0956     | Reserv  Avail   Avail   Avail   Avail   Avail   Reserv  Avail
0965     | Locked  Locked  Locked  Bad_Lk  Bad_Lk  Bad_Lk  Reserv  Avail

                          ... etc ...
--------------------------------------------------------------------------
```

**Reserv** This means that the page is permanently reserved by the operating system, and cannot be obtained for testing (in an on-line environment).

**Avail** This means that the page is NOT LOCKED, and is possibly available for testing because it has not been allocated to any other process. However, many pages may show Available status and not be LOCKABLE due to constant system usage.

**Busy** This means that the page is currently busy and, therefore, is unavailable for testing until the process currently using it has completed. This is usually associated with I/O.

**Hole** Page numbers with no physical manifestation are termed "holes". For example, if the maximum page of the system is 20479, and somehow page 20500 was requested, it would not exist and the status of "hole" would be returned.

**Locked** This means that the page is currently locked by a process (that process may be your own diagnostic or an external process).

**Bad** This means the page has been marked "bad" by the operating system. For example, if a diagnostic found excessive errors within a page, it could mark the page "bad" and lock out other processes from accessing this corrupted page. This is dependent upon the operating system's ability to bar user access to corrupted pages.

**Bad_Locked** This means that a page is both Locked and deemed Bad within the system.

If the user selects *10* (*Help Menu*) from the Main Menu, the following menu is displayed:

```
=========================================================================
                          MEMDIAG HELP MENU
=========================================================================

                          1.   Memory Test  (using parameter settings)
                          2.   EDC Test     (using parameter settings)
                          3.   Sweep Memory (using parameter settings)
                          4.   Set MEMDIAG parameters
                          5.   View MEMDIAG parameters
                          6.   View Configuration Information
                          7.   View Page Status Lists
                          8.   View Page Status Summary
                          9.   View Memory Status (over Range)
                         10.   Exit to Main Menu
                         11.   Exit to DUI

                         Input the area you want >>>
```

Selecting any of these items will scroll the explanations presented in this manual for each of the listed areas.

If the user selects *11* (*Exit to DUI*) from the Main Menu, control is returned to the DUI command line.

## Section 9 : TROUBLE TREE (Default Test)

This section can be used to perform all possible diagnostics within the **MEMDIAG** repertoire. This is the **default** section for the Memory Array Diagnostic. If you just type `run memdiag`, then this section will be executed. The testing sequence for this section is as follows:

1. Initialize memory controller status to clear status errors.
2. Perform memory pattern test with
   a. All Zeros
   b. All Ones
3. Print configuration information
4. Perform memory pattern test with
   a. Address Uniqueness
   b. Address Uniqueness Complement
   c. Walking Ones
   d. Walking Zeros
   e. Alternating Ones
   f. Alternating Zeros

**POSSIBLE OUTPUT MESSAGES:**

```
------------------------------------------------------------------------------------------------------------------
                                         TROUBLE TREE
------------------------------------------------------------------------------------------------------------------

   >>>  SECTION 9

   >>>  ERROR REGISTERS CLEARED SUCCESSFULLY

   >>>  ONES PATTERNS TEST COMPLETED SUCCESSFULLY

           (tested page list displayed here)

   >>>  ZEROS PATTERNS TEST COMPLETED SUCCESSFULLY

           (tested page list displayed here)

   >>>  MEMORY CONTROLLERS/MEMORY ARRAYS INITIALIZED SUCCESSFULLY

   >>>  SYSTEM CONFIGURATION TABLE

           (tables displayed here)

   >>>  ADDRESS UNIQUENESS PATTERN TEST COMPLETED SUCCESSFULLY

           (tested page list displayed here)

   >>>  WALKING ONES/ZEROS PATTERN TEST COMPLETED SUCCESSFULLY

           (tested page list displayed here)

   >>>  ALTERNATING ONES/ZEROS PATTERN TEST COMPLETED SUCCESSFULLY

           (tested page list displayed here)


   >>>
   >>>
   >>>  MEMDIAG HAS COMPLETED SUCCESSFULLY WITHOUT FINDING ANY MEMORY ERRORS.
   >>>
   >>>


   >>>  END OF SECTION 9
```

**DEFAULT:**

This section has no steps

## Error and Warning Messages

Error messages may be displayed as a result of Syndrome Register reads, or as a result of memory errors occurring during the operation of `MEMDIAG`; memory errors may also generate warning messages.

### Syndrome Register Read

One Syndrome Register is associated with each Memory Controller in a memory system. The Syndrome register will contain the status of the last memory read or write operation to memory under the supervision of the associated Memory Controller. The memory operation will result in a SUCCESSFUL completion (Message 10250) or in a FAILURE to complete (Messages 10251 thru 10258).

The possible status values are listed below. These messages are actual decoding of the hardware status returned after the memory transaction.

**Cause:** Memory hardware failed to complete memory transaction.

**Action:** Replace chip or entire card indicated in the message immediately following this one that will pinpoint the exact Field Replaceable Unit (FRU) that is malfunctioning.

| | |
|---|---|
| **ERROR MESSAGE 10250** | `NO Memory Errors registered.` |
| **ERROR MESSAGE 10251** | `SINGLE BIT ERROR due to UNKNOWN cause.` |
| **ERROR MESSAGE 10252** | `SINGLE BIT ERROR due to DATA bits.` |
| **ERROR MESSAGE 10253** | `SINGLE BIT ERROR due to CHECK bits.` |
| **ERROR MESSAGE 10254** | `SINGLE BIT ERROR logged on PARITY read.` |

| | |
|---|---|
| **ERROR MESSAGE 10255** | SINGLE BIT ERROR logged on PARITY write. |
| **ERROR MESSAGE 10256** | MULTI BIT ERROR due to UNKNOWN cause. |
| **ERROR MESSAGE 10257** | FATAL ERROR registered. |
| **ERROR MESSAGE 10258** | UNKNOWN ERROR registered. |

## Error/Warning Messages

The following are possible Error/Warning messages that may be displayed during the operation of MEMDIAG.

MEMORY ERRORs map to one of two different error types: Card Error or Buffer Error. The most important difference is that Card Errors are physical errors and an FRU will be pinpointed; however, Buffer Errors are virtual errors and no FRU can be determined.

```
ERROR MESSAGE 10259

***
***  Unable to retrieve message # ! from MEMDIAG catalog
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE The message number indicated was not found in the message catalog of the diagnostic.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
ERROR MESSAGE 10260

***
***  Memdiag is unable to OPEN memory, BAD exit status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***  (3) Check to see if /dev/dmem is there, or if /dev/dmem is included in
***      the kernel for an HPUX system.
***
```

CAUSE An operation requested by Memdiag to open memory was not successfully completed due to hardware or software below Memdiag. Usually this means the hardware system is not on (recognized) by the supported list Memdiag officially tests.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
WARNING MESSAGE 10261

***
***       W A R N I N G :
***
***     It is possible that MEMDIAG will force a parity error to
***     occur.  This may cause the machine to freeze or generate
***     a High Priority Machine Check.
***
```

CAUSE This warning will print out when the memory system has parity data checking instead of EDC error checking. Performing memory tests will exercise memory that may not normally be accessed. This will increase the probability of coming across a memory failure. However, with parity memory checking the error will be detected but not corrected and could, therefore, cause an HPMC.

ACTION This warning is basically for old systems that are running with parity cards (ie: Burgundy), which are not "officially" supported but the warning is printed to inform user of possible results.

---

```
WARNING MESSAGE 10262

***
***   Bad RANGE input, this range goes over End Of Memory.
***
***        Please enter a SMALLER number.
***
```

CAUSE User has input a Page Range that when added to the current requested Starting Page will put requested pages over valid page numbers in the memory system.

ACTION Input a smaller Page Range value.

---

```
ERROR MESSAGE 10263

***
***  At the current time, the Operating System was not able to
***  ALLOC the minimum buffer size needed to run MEMDIAG.
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE The minimum buffer required to execute Memdiag was not obtained, the memory space is not available in the present state of the environment. The system is saying it ran out of memory pages to give away.

ACTION Exit the diagnostic and restart again, this will deallocate all memory pages obtained by the diagnostic and try again. If it still does not work collect data about the problem and submit to STD Online Support.

---

```
ERROR MESSAGE 10264

***
***  At the current time, the Sherlock Interface was not able
***  to obtain a pattern buffer from the Operating System of
***  the minimum buffer size needed to run MEMDIAG.
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE The minimum buffer required to execute Memdiag was not obtained, the memory space is not available in the present state of the environment. The system is saying it ran out of memory pages to grant ownership to.

ACTION Exit the diagnostic and restart again, this will deallocate all memory pages obtained by the diagnostic and try again. If it still does not work collect data about the problem and submit to STD Online Support.

---

```
WARNING MESSAGE 10265

***
***   At the current time, the Operating System was not
***   able to allocate the buffer size requested so try
***   requesting a smaller ALLOC SIZE parameter.
***
```

CAUSE At the current time, the operating system does not seem to have the amount of pages necessary to support running Memdiag with the ALLOC SIZE you requested.

ACTION Input a smaller ALLOC SIZE parameter.

---

```
WARNING MESSAGE 10266

***
*** WARNING -- You have reached your maximum error total
***
```

CAUSE The user specified error limit has been reached.

ACTION If you would like to allow for more errors before termination, re-run the diagnostic assigning a larger value to the errcount parameter of the run command.

---

```
WARNING MESSAGE 10267

***
*** WARNING -- Invalid response.  Please try again.
***
```

CAUSE
Valid ranges acceptable for this prompt will be displayed either above the prompt line or at the end of the prompt line.

ACTION
Input value from within displayed range.

---

```
ERROR MESSAGE 10268

***
***  Memdiag has encountered a BUFFER error while testing.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE
This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc).

ACTION
Rerun the test to see if the Buffer Error will repeat itself but this time map to physical memory where we can pinpoint an FRU. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system.

---

ERROR MESSAGE 10269

```
***
***  The Driver indicates that there are NO Memory Controllers in the
***  system.  Check that Memory Controller card is properly installed.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE When the driver has gone out to poll the bus line for Memory Controller cards, it has not found one present. This is obviously an impossibility because a MC card must be present within the system for it to function at all; however, the reporting of the MC card can fail in transit to the Memdiag query. This indicates some sort of system failure outside the scope of Memdiag.

ACTION Check for MC card and make sure it is completed pushed into slot, if it still does not work collect data about the problem and submit to STD Online Support.

---

ERROR MESSAGE 10270

```
***
***  A CARD error was detected and logged in the Syndrome register.
***     The following messages will give details of error location.
***
```

CAUSE A physical memory has occurred during a read or write operation to memory.

ACTION Replace Memory Controller or Memory Array card indentified in the subsequent error messages that will pinpoint the FRU.

---

ERROR MESSAGE 10271

```
***
***  A BUFFER error was detected during a comparision of Expected
***                      data to Actual Data read back from memory.
***
***     This type of error indicates some system malfunction OTHER than the
***     Memory Array cards or memory on board the Memory Controller card.
***     If these were faulty, the error would have shown during the readback
***     of the Syndrome Error Register.
***
***     The MC and/or MA card slots that map to the ERROR PAGE are given
***     only as an environment "dump" of all configuration information
***     associated with the ERROR PAGE at the time of the BUFFER error.
***
***     It is recommended that you retest the ERROR PAGE or the Page Range
***     of the associated Memory Controller and/or Memory Array where the
***     BUFFER error occurred (go into User Interactive, Section 8).
***
***     The following messages will give details on the BUFFER ERROR.
***
```

CAUSE — This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc).

ACTION — Rerun the test to see if the Buffer Error will repeat itself but this time map to physical memory where we can pinpoint an FRU. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system.

```
ERROR MESSAGE 10272

***
***   ERROR found at Page 1840
***
***     Expected Data = 0x       0
***
***     Actual   Data = 0x       1
***
***     Test Pattern  = All Zeros
***
```

CAUSE This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc).

ACTION Rerun the test to see if the Buffer Error will repeat itself but this time map to physical memory where we can pinpoint an FRU. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system.

---

```
ERROR MESSAGE 10273

***
***   MEMORY ERROR DETECTED
***
***     Test Pattern                  =   All Zeros
***
***     Bit  Location                 =   63
***     Chip Location                 =   u0301
***     Bank Location                 =   0
***
***     Memory Controller Slot        =   0
***     Memory Array Slot             =   0
***
```

CAUSE A physical memory has occurred during a read or write operation to memory.

ACTION Replace Memory Controller or Memory Array card indentified in the subsequent error messages that will pinpoint the FRU.

---

ERROR MESSAGE 10275

***** Under Memory Controller in Slot 3 maps to the BUFFER ERROR PAGE.**

CAUSE This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc).

ACTION Rerun the test to see if the Buffer Error will repeat itself but this time map to physical memory where we can pinpoint an FRU. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system.

---

ERROR MESSAGE 10276

***** Memory Array in Slot A maps to the BUFFER ERROR PAGE.**

CAUSE This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc).

ACTION Rerun the test to see if the Buffer Error will repeat itself. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system. If it repeats as a Card Error then the FRU will be identified.

---

ERROR MESSAGE 10277

```
***
***   FAILURE in INITIALIZATION of Memory Controller
***
***   Replace Memory Controller chip or entire card.
***
```

CAUSE A physical memory has occurred during a read or write operation to memory.

ACTION Replace Memory Controller or Memory Array card indentified in the subsequent error messages that will pinpoint the FRU.

---

ERROR MESSAGE 10278

```
***
***  FAILURE in retrieving Configuration Information
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE The configuration tables did not print out successfully. This indicates some sort of print utility failure, it will not effect successful completion of diagnostic.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
ERROR MESSAGE 10279

***
***   FAILURE in Initialization of Memory
***
```

| | |
|---|---|
| CAUSE | A physical memory has occurred during a read or write operation to memory. |
| ACTION | Replace Memory Controller or Memory Array card indentified in the subsequent error messages that will pinpoint the FRU. |

---

```
ERROR MESSAGE 10280

***
***   FAILURE in Pattern Testing of Memory
***
```

| | |
|---|---|
| CAUSE | A physical memory has occurred during a read or write operation to memory. |
| ACTION | Replace Memory Controller or Memory Array card identified in the subsequent error messages that will pinpoint the FRU. |

---

ERROR MESSAGE 10281

```
***
***     The TROUBLE TREE has NOT completed successfully.
***
```

CAUSE This indicates an error occurred somewhere in the testing sequence of the Trouble Tree section. This message is the global summation of all test sequences called for Memdiag, in case the User has run overnight and returned to the console to see the last message printed out before Memdiag terminated.

ACTION Look back through previous messages to find Specific Error Message that will pinpoint a Field Replaceable Unit.

---

ERROR MESSAGE 10282

```
***
***  Unable to successfully acquire DAR buffer.
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE An operation requested by Memdiag was not successfully completed due to hardware or software below Memdiag.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
ERROR MESSAGE 10283

***
***  Unable to initialize Memdiag in diagostic environment.
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE An operation requested by Memdiag was not successfully completed due to hardware or software below Memdiag.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
ERROR MESSAGE 10284

***
***  Loading of a Section or Step failed.
***  MEMDIAG will now terminate. Error Status = !
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE An operation requested by Memdiag was not successfully completed due to hardware or software below Memdiag.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

WARNING MESSAGE 10285

```
***
***    W A R N I N G
***
***    Selecting GROUPS of patterns in the User Interactive section
***    will effect system performance and increase test duration.
***
***    If you have not already done so, run with ONE pattern to test
***    out your systems  performance before trying multiple pattern tests.
***
```

CAUSE Memory tests are CPU intensive tests that may effect system performance adversely.

ACTION It is suggested you run Memdiag using ONE pattern first to test out system performance under your own unique circumstances of system load, system operating system and specific hardware.

---

ERROR MESSAGE 10286

```
***
***  Interrupt POLL failed with Status = !
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE An operation requested by Memdiag was not successfully completed due to hardware or software below Memdiag.

ACTION Get the version number of the diagnostic, indicate what you were attempting to do, and file an SR.

---

```
WARNING MESSAGE 10287

***
***        W A R N I N G
***
***   The section you have selected will run memory tests that
***   take over one half hour to complete (on average).
***
***   If you don't have the time now, go into User Interactive (Sect 8)
***   and setup your own memory test by selecting parameters that will
***   specifically test certain ranges and, therefore, shorten test duration.
***
***   Do you want to continue with this Section now? (y/n) [y]:
***
```

CAUSE This is to give the CE a way out if he is not aware of the time required to run these memory tests.

ACTION If you can run the diagnostic in the background, do so. If you are looking for immediate results, go into the User Interactive (Section 8) and run specific test ranges.

---

```
ERROR MESSAGE 10289

***
*** Memdiag is unable to read Syndrome Register
*** in order to display status,  Bad Exit Status = !
*** MEMDIAG will continue ...
***
*** Action:
*** (1) Try to run the section again with ERRPRINT command modifier. or
*** (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag requested lower level software to read the Syndrome Register in order to decode the error flags set, if any. The request was not successfully completed. This is only for display so, therefore, Memdiag can continue.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10290

```
***
***  Memdiag is unable to read Model Information
***  in order to display Revision Info,  Bad Exit Status = !
***  MEMDIAG will continue ...
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag requested lower level software to read the Model Information in order to display revision information. The request was not successfully completed. This is only for display so, therefore, Memdiag can continue.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10291

```
***
***  Memdiag is unable to Allocate pages.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag was not able to successfully allocate a page from the operating system. This means it could not be granted ownership of it in order to test it.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10292

```
***
***  Memdiag is unable to Deallocate pages.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag was not able to successfully deallocate a page to the operating system. This means it could not be return ownership of it.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10293

```
***
***  Memdiag is unable to User Allocate pages.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag was not able to successfully allocate a page from the operating system. This means it could not be granted ownership of it in order to test it.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

```
ERROR MESSAGE 10294

***
***  Memdiag is unable to Clear Syndrome Register.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag requested lower level software to clear the Syndrome Register. Memdiag must terminate because if the syndrome can't be cleared it will read an error at each cycle of the testing.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

```
ERROR MESSAGE 10295

***
***  Memdiag is unable to read System DETAIL Info.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag requested lower level software to read the system detail information in order to test memory. Memdiag cannot test without knowing what memory controller/array cards are present.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10296

```
***
***  Memdiag is unable to read System CONFIGURATION Info.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

| | |
|---|---|
| CAUSE | Memdiag requested lower level software to read the system configuration information in order to test memory. Memdiag cannot test without knowing what memory controller/array cards are present. |
| ACTION | Try running the section again. If it still fails, record exit status and submit to STD Online Support. |

ERROR MESSAGE 10297

```
***
***  Memdiag is unable to read System MODEL Info.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

| | |
|---|---|
| CAUSE | Memdiag requested lower level software to read the Model Information in order to display revision information. The request was not successfully completed. This is needed for Memdiag to order the SPAs of each memory range, therefore, Memdiag cannot continue without it. |
| ACTION | Try running the section again. If it still fails, record exit status and submit to STD Online Support. |

ERROR MESSAGE 10298

```
***
***  Memdiag is unable to read Syndrome Register.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE — Memdiag requested lower level software to read the Syndrome Register in order to decode the error flags set, if any. The request was not successfully completed. This is for error decoding after the test and is required.

ACTION — Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10299

```
***
***  Memdiag is unable to READ Memory.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE — Memdiag requested lower level software to read the memory after a pattern test write.

ACTION — Try running the section again. If it still fails, record exit status and submit to STD Online Support.

---

ERROR MESSAGE 10300

```
***
***  Memdiag is unable to WRITE Memory.  Bad Exit Status = !
***  MEMDIAG will now terminate.
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

| | |
|---|---|
| CAUSE | Memdiag requested lower level software to write the memory for pattern testing. |
| ACTION | Try running the section again. If it still fails, record exit status and submit to STD Online Support. |

---

ERROR MESSAGE 10301

```
***
***  Memdiag is unable to read Page Status for page !
***  Bad Exit Status = !,  MEMDIAG will continue ...
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

| | |
|---|---|
| CAUSE | Memdiag requested lower level software to read the type of page status associated with the indicated page. |
| ACTION | Try running the section again. If it still fails, record exit status and submit to STD Online Support. |

---

```
ERROR MESSAGE 10304

***
***     Memdiag is unable to log memory error to MEMLOGP.
***     Bad Exit Status = !,  MEMDIAG will continue ...
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

| | |
|---|---|
| CAUSE | Memdiag sent an IPC message to log a memory error to MEMLOG file, but MEMLOGP was not running. |
| ACTION | Type in DIAGSYSTEM RUN MEMLOGP at DUI prompt. Try running the section again. If it still fails, record exit status and submit to STD Online Support. |

---

```
ERROR MESSAGE 10306

***  Bank Number 0 maps to the BUFFER ERROR PAGE.
```

| | |
|---|---|
| CAUSE | This data error maps to Virtual Memory which is outside the scope of the Memory Array Diagnostic. It can indicate a failure anywhere within the HPPA system (ie bus, hard disk, cables, etc). |
| ACTION | Rerun the test to see if the Buffer Error will repeat itself. If the error does not repeat, chances are you will never know what caused it. If it does repeat, and repeats again as a Buffer Error, run other diagnostics to try and locate the failure within the system. If it repeats as a Card Error then the FRU will be identified. |

---

ERROR MESSAGE 10309

```
***
***  Memdiag is unable to get Page Size from MEMDAR
***  Bad Exit Status = !,  MEMDIAG will not continue ...
***
***  Action:
***  (1) Try to run the section again with ERRPRINT command modifier. or
***  (2) Try to run the section again with TRACE=ALL command modifier.
***
```

CAUSE Memdiag requested lower level software to obtain page size for page allocation.

ACTION Try running the section again. If it still fails, record exit status and submit to STD Online Support.

# Contents

# 4

# HPIBDAD/HPIBEISA

## Introduction (HPIBDAD)

The HP-IB Device Adapter Diagnostic (HPIBDAD) is part of the online diagnostic package. It is designed to provide its user with a means of determining if the specified HP-IB Device Adapter (DA) and its related hardware are operating properly, and if not, which FRU should be replaced. There are a variety of tests which the user can run to determine the source of a problem. Some of the tests require writing and reading data to and from (respectively) an external device (e.g., a disk drive). Tests that require such action need only be run during exceptional situations and are included so that the diagnostic can help in determining the source of a problem. However, these tests may not be appropriate to run in all environments. For example, if the HP-IB DA to be tested is connected to a boot device, executing the write/read tests on that device could write over the operating system code. The result of such an action could be a system crash.

**Note**

Any interaction with a device on the HP-IB is administered by the diagnostic user, NOT the diagnostic. Therefore, extreme caution should be taken by the user before such tasks are undertaken.

## Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database referencing product number 30600-10028.

## Minimum Configuration

The hardware required to run HPIBDAD consists of a Hewlett-Packard Precision Bus (PB) HP-IB Device Adapter (part number 28650A), as well as an HP 815 computer system with all the necessary equipment to bring the operating system (HP-UX) up and running.

However, in order to fully test the HP-IB Device Adapter, an HP-IB Talker/Listener/Controller device is required to be connected to the HP-IB. More specifically, there must be a device connected to the HP-IB which has the ability to Talk, and whose output data can be verified (e.g., a digital multimeter which outputs a known voltage); there must be a device connected to the HP-IB which has the ability to Listen, and whose input data (command) can be verified (e.g., a plotter which is given the "command" to lift its pen); and there must be a device that can take control of the HP-IB. It doesn't matter whether or not all of this functionality is implemented in a single device or in more than one device.

In order to run HPIBDAD, the online diagnostic system must be present, along with the portability interface routines. Also, the diagnostic must guarantee that regardless of what condition the HP-IB Device Adapter (DA) to be tested is in, the system will not "crash" by running the diagnostic on it.

Other software that must be present in the system to execute HPIBDAD is the following: 1) the HP-IB Device Access Routines (DAR) which act as an interface between the diagnostic and the Logical Device Manager (LDM), 2) the LDM which is the interface between the DAR and low-level driver, and 3) the HP-IB Device Adapter Manager (DAM) which is the low-level driver to the HP-IB DA. Note that the DAR is technically part of the portability interface, as far as online module structure is concerned. Also note that in this document, references to the DAM imply the LDM as well, unless otherwise noted.

## Operating Instructions

HPIBDAD is accessed by the user via the Diagnostic User Interface (DUI).

### Default Tests

The default sections and steps for this diagnostic are:

```
Section 3    Identify

Section 6    Status

  Step 61      Preliminary Internal State Diagnosis

  Step 62      Read HP-IB Interface Chip STATUS Register

  Step 63      Read HP-IB Interface Chip CONTROL Register

  Step 64      Read HP-IB Interface Chip ADDRESS Register

  Step 65      Read HP-IB Interface Chip PP/ID_BYTE Registers

  Step 66      Read HPIB_STATUS Register

  Step 67      Read BUS_STATUS Register
```

### RUN Command

To bring up the Online Diagnostic subsystem, enter the following command to the HP-UX system prompt:

```
% sysdiag
```

The diagnostic subsystem responds with the following prompt indicating that access has been granted to the user:

```
DUI >
```

Typing HELP causes a summary of the DUI function and its commands to appear on the screen.

**Note** The device to be tested must be powered up and on line. The physical device location (pdev) shown below matches the same device shown on the "typical A1002A" system configuration, described in the chapter on DUI. The pdev value entered must be correct for the system being tested.

After the online diagnostics system has been started, this diagnostic can be executed using the command:

```
run hpibdad
```

All online diagnostics RUN string parameters are acceptable when executing this diagnostic. All parameters available in the `run` command are acceptable as parameters when executing this diagnostic.

---

**Note**

All of the sections in this diagnostic can be executed from any terminal, even if a specific test requires the user to have the capability to run destructive tests. This implementation therefore allows the diagnostic to be run from a remote terminal; however, there may be system limitations that would not allow the use of a remote terminal.

---

## Test Execution

When HPIBDAD is invoked, the following header and welcome messages will be displayed:

```
*****************************************************************
******                                                    ******
******    HPIBDAD : HP-IB Device Adapter Diagnostic       ******
******                                                    ******
******    (c) Copyright Hewlett-Packard Company 1987      ******
******               All Rights Reserved                  ******
******                                                    ******
******                 Version V.UU.FF                    ******
******                                                    ******
*****************************************************************

Welcome, Today is day, date, time
```

After the header and welcome messages are displayed via a program services call, the diagnostic will call another program services routine in order to obtain access to the device that was selected for testing (in addition to setting up the sections and steps to be run).

This routine will exit with its *status* parameter (passed by reference) being any one of three possible values. The first of which is *successful*. This indicates that all sections and steps have been validated and that the system granted access to the device.

The second possible value is *dssd_device_in_use*. If this value is returned, it indicates that the system did not grant access to the device. If this happens, the following error message will be issued by the diagnostic:

```
*** ERROR -- HP-IB DEVICE ADAPTER ALREADY IN USE BY THE
***          DIAGNOSTIC SYSTEM.                                 (HDADERR 5000)
***
***          Someone has already gained exclusive rights to the
***          HP-IB Device Adapter that you requested, and it is illegal
***          to have two copies of the HPIBDAD diagnosing the same HP-IB
***          Device Adapter simultaneously.
```

The diagnostic will terminate execution after outputting this error message.

The third possible status value is *dssd_internal_error*. When this value is active upon exiting the subroutine, it indicates that an error such as no device adapter at the specified LDEV was found. The online diagnostics themselves will output the error message for this situation, *not* HPIBDAD. The diagnostic will terminate upon regaining control.

If all went well up to this point, the sections and steps specified by the user will be executed and the results displayed. If the user did not specify any sections/steps to be run, the default sections and steps will be executed (Sections 3 and 6). If at any time, the number of errors generated exceeds the limit specified by the user in the *errcount* parameter (of the DUI *run* command), the following message will be output:

```
*** WARNING -- The maximum specified number of error occurrences has
***            been exceeded.                                  (HDADWARN 6000)
```

The diagnostic will then terminate its execution. If the *errpause* parameter of the **run** command was assigned to "on", then the diagnostic will stop after each error is generated and ask the user if the testing should continue. The prompt that will be displayed is as follows:

```
Do you wish to continue? (Y/N) [Y] :
```

If the response is "Y", then the testing will resume (if possible), and if the response is "N", the diagnostic will terminate its execution. If the sections and steps specified by the user were executed the number of times specified in the *loop* parameter of the **run** command without the number of errors exceeding the *errcount* value, the diagnostic will terminate normally.

At any time that the diagnostic is prompting the user for information, the user may enter "exit" to terminate its execution, or enter "suspend" to temporarily suspend its execution. Either the entire word or any number of characters which uniquely identify the respective language localized command may be entered. Moreover, the letters entered may be in any combination of upper and lower case characters. If the user exits in this fashion, the following message is displayed:

```
... Exiting HPIBDAD per your request.
```

If the user temporarily suspends execution in this manner, the message that will be displayed is as follows:

```
... HPIBDAD suspended per your request.
```

The user can then perform tasks through the Diagnostic User Interface (DUI) and subsequently resume execution of `HPIBDAD`, or he/she can abort the `HPIBDAD` entirely.

**Note**

In situations such as timeouts, this diagnostic will inform the user which operation was taking place when the timeout occurred; moreover, the user will be given a list of commands which executed successfully (if any). It is believed that the value of this information outweighs the risk of inundating the user with output.

## Detailed Test Descriptions

The remainder of this section discusses each section and step in detail. As a quick reference, the following table was included to list all of the sections and steps available for use in HPIBDAD.

```
  Section No.       Diagnostic Function
================================================================================

       1            More Help

       2            Reset

       3            Identify

       4            Local Loopback
                       Step 42 - Loopback from PB Interface Chip
                       Step 43 - Loopback from HP-IB Interface Chip

       5            Hardware Test

       6            Status
                       Step 61 - Preliminary Internal State Diagnosis
                       Step 62 - Read HP-IB Interface Chip STATUS Register
                       Step 63 - Read HP-IB Interface Chip CONTROL Register
                       Step 64 - Read HP-IB Interface Chip ADDRESS Register
                       Step 65 - Read HP-IB Interface Chip
                                     PP/ID_BYTE Registers
                       Step 66 - Read HPIB_STATUS Register
                       Step 67 - Read BUS_STATUS Register

      10            Register Level Input/Output Transactions

      11            Data/Command Transaction on HP-IB
```

## Section 1—More Help

```
Minimum Mode Required       : Normal

Terminal Used for Execution : Any

In Default Set?             : No

In Auto-diagnostic Set?     : No
```

`More Help` is an interactive section which allows the user to obtain more information about a particular section than is given when typing `help hpibdad` at the DUI prompt. This is needed because it is not desirable to spew large help screens at the user when he/she is looking for general help, but it is desirable to give more information about certain sections when requested.

This section allows all users from any terminal to obtain the additional information that they request.

**Possible Output Messages:**

```
Section 1 -- More Help

    This Section allows you to get more information on any of the Sections
    [1..6, 10, 11] of this diagnostic.  Please indicate the number of the
    section for which you require more information.  Entering a lone <CR> to
    the prompt exits this section.

    More Help (1..6, 10, 11, <CR>) : [Return]

End of Section 1 -- More Help
```

If the user enters a section number in response to the prompt, the pertinent information would be displayed for the user to read. The numbering scheme used for the help messages is as follows: the number of the message is equal to *(section_number) * 100 + 10000*. This allows for 100 messages per section. Note that when multiple messages exist for a given section, all of the corresponding messages are displayed.

**Note**

Informational messages are indented four spaces from the left margin and implement standard capitalization rules. This is done in order to make the error/warning messages more obvious to the user (which are not indented from the left margin, and are preceded by three asterisks).

**Possible Error/Warning Messages:**

If the user inputs a number of a section that is not implemented (i.e. not in the set {1,2,3,4,5,6,10,11}), then the following message is output:

```
*** WARNING -- Invalid response.  Please answer the question with one
***            of the choices given.                        (HDADWARN 6001)
```

```
More Help (1..6, 10, 11, <CR>) :
```

Note that the user is prompted again for input.

**HP-IB DAR Operations Used:**

None specified by the diagnostic.

## Section 2—Reset

```
Minimum Mode Required       : Destructive

Terminal Used for Execution : Any

In Default Set?             : No

In Auto-diagnostic Set?     : No
```

This section informs the Device Adapter Manager (DAM)—via the Device Access Routines—to reset the HP-IB Device Adapter (DA) and DAM to its power-on state. The DAM will then configure the card with information it maintains internally.

**Possible Output Messages:**

```
Section 2 -- Reset

   NO ERRORS DETECTED while resetting the device adapter.

End of Section 2 -- Reset
```

**Possible Error/Warning Messages:**

```
None specified by the diagnostic.
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_RESET
HPIB_UNLOCK
```

## Section 3—Identify

```
Minimum Mode Required       : Destructive

Terminal Used for Execution : Any

In Default Set?             : Yes

In Auto-diagnostic Set?     : No
```

Identify issues an HPIB_IDENTIFY command to the HP-IB DAR, which acquires the requested information from the HP-IB DAM, which in turn acquires most of the information from the HP-IB DA's IODC (the only information not retrieved from the IODC is the DAM, LDM, and DAR version codes). The diagnostic then decodes the information obtained and displays it in a manner that is informative to the user. This section can be used to determine the HP-IB DA's hardware and software versions, as well as the version of the DAM, LDM, and DAR being used. Moreover, a checksum is calculated by the system on the appropriate IODC information when this section is executed. If the checksum test fails, an error message will be displayed. This section has an added benefit in that if it executes successfully, the path from the diagnostic to the HP-IB DA is known to be at least partially functional.

**Possible Output Messages:**

```
Section 3 -- Identify


   Hardware Version                 : 0x?

   Soft Physical Address Capability : 0x80

   Type of Module                   : 4 (Type A DMA I/O Adapter)

   Software Version                 : 0x?

   IODC Revision                    : ?

   ... Checksum Verified.

   Device Adapter Manager Version   : ?

   Logical Device Manager Version   : ?

   Device Access Routine Version    : ?

   Note: The "0x" prefix is used to specify that the respective number
         is in hexadecimal format.  Also note that version numbers
         depicted by '---' implies that the actual version number was not
         accessible.


End of Section 3 -- Identify
```

**Note**

The Hardware Version, Software Version, IODC Revision, Device Adapter Manager Version, Logical Device Manager Version, and Device Access Routine Version fields may vary in time, therefore cannot be explicitly specified within this document.

**Possible Error/Warning Messages:**

If the Soft Physical Address Capability is not 0, the following message is displayed:

```
*** WARNING -- Soft Physical Address Capability = 0x!,
***            I expected  0x80.                         (HDADWARN 6010)
```

If the module type value returned is not that of a Type A DMA I/O Adapter, the following message is displayed:

```
*** WARNING -- Type of Module = ! (UNKNOWN PRODUCT),
***            I expected a 4 (TP_A_DMA).                (HDADWARN 6011)
```

If the checksum does not equal zero, the following error message is displayed:

```
*** ERROR -- IODC CHECKSUM FAILED.                        (HDADERR 5005)
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_IDENTIFY
HPIB_UNLOCK
```

## Section 4—Local Loopback

```
Minimum Mode Required         : Destructive

Terminal Used for Execution : Any

In Default Set?               : No

In Auto-diagnostic Set?       : No
```

The local loopback tests will determine the operational status of the backplane and midplane of the HP-IB Device Adapter. This will be accomplished by writing and reading data to and from (respectively) the HP-IB DA via HPIBDAD—in an "onion skin" fashion. For this reason, the local loopback tests are divided into two steps according to how "deeply" they interact with the HP-IB DA.

If this section is run without the desired steps being explicitly specified, Steps 42 and 43 will both be run by default.

**Note**

Since the HP-IB requires bidirectional drivers on the frontplane, the frontplane cannot be tested via local loopback. The only way to test the frontplane would be with either an active loopback hood, or by transmitting data to and from an external device. Since the former option is not very practical, the latter option was chosen for this diagnostic. However, this functionality is NOT controlled by HPIBDAD, but rather is controlled by the user. For more information, please see Section 11, "Data/Command Transaction on HP-IB".

## Step 42—Loopback from PB Interface Chip

In this step, data will be written to the PB Interface Chip and read back, testing the hardware layer "just below" the backplane (recall that the "onion skin" approach to diagnostics is implemented).

**Note** The chip is used to interface the PB backplane to the midplane of the HP-IB DA.

HPIBDAD will compare the data read with the data written, and display the appropriate messages.

**Possible Output Messages:**

```
Section 4 -- Local Loopback

   Step 42 - Loopback from PB Interface Chip

      NO ERRORS DETECTED while executing loopback from the PB
      Interface Chip.

   End of Step 42 - Loopback from PB Interface Chip

End of Section 4 -- Local Loopback
```

**Possible Error/Warning Messages:**

```
*** ERROR -- PB INTERFACE CHIP (PBIC) LOOPBACK FAILED.          (HDADERR 5010)
***
***         Data written to PBIC (in hex)       Data read from PBIC (in hex)
***         -----------------------------       ----------------------------
***                      !                                   !
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_WRITE_REG
HPIB_READ_REG
HPIB_RESET
HPIB_UNLOCK
```

## Step 43—Loopback from HP-IB Interface Chip

In this step, data will be written to the HP-IB Interface Chip and read back, testing the hardware up to but not including the frontplane transceivers.

**Note** The chip is an HP-IB Talker/Listener/Controller.

HPIBDAD will compare the data read with the data written, and display the appropriate messages.

**Possible Output Messages:**

```
Section 4 -- Local Loopback

   Step 43 - Loopback from HP-IB Interface Chip

      NO ERRORS DETECTED while executing loopback from the HP-IB
      Interface Chip.

   End of Step 43 - Loopback from HP-IB Interface Chip

End of Section 4 -- Local Loopback
```

**Possible Error/Warning Messages:**

```
*** ERROR -- HP-IB INTERFACE CHIP (HIC) LOOPBACK FAILED.        (HDADERR 5020)
***
***          Data written to HIC (in hex)          Data read from HIC (in hex)
***          ----------------------------          ---------------------------
***                        !                                    !
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_WRITE_REG
HPIB_READ_REG
HPIB_RESET
HPIB_UNLOCK
```

## Section 5—Hardware Test

```
Minimum Mode Required        : Destructive

Terminal Used for Execution  : Any

In Default Set?              : No

In Auto-diagnostic Set?      : No
```

Executing this section will perform a thorough hardware test on the HP-IB DA. Not only will the PB Interface and the frontplane interface chips be tested, but the glue logic and backplane transceivers will also be tested. The only components that will be untested after this section is executed is the glue logic that is inaccessible to HPIBDAD (obviously), and the frontplane transceivers. The reason why the frontplane transceivers will remain untested is because there is no straightforward way to test them. The only way they can be tested is by going off the card, and since this functionality would be impractical to include in HPIBDAD, it was chosen not to test the frontplane transceivers. However, the user does have the capability to do I/O with devices connected to the HP-IB; therefore, it is not impossible to test the frontplane transceivers with this diagnostic software.

The user will be given the capability to go onto the HP-IB since he/she should know what devices are "out there" and should also know their respective command sets; whereas to include the capability for HPIBDAD to be able to identify a particular device (if possible) and be able to communicate with it would be a supererogatory effort on the part of the diagnostic writer.

Since this hardware testing brings the card off-line, it should only be done when absolutely necessary.

**Possible Output Messages:**

```
Section 5 -- Hardware Test

   NO ERRORS DETECTED while testing the PB Interface Chip's
        IO_EIM register.

   NO ERRORS DETECTED while testing the PB Interface Chip's
        IO_DMA_LINK register.

   NO ERRORS DETECTED while testing the PB Interface Chip's
        IO_DMA_COUNT register.

   NO ERRORS DETECTED while testing the HP-IB Interface Chip's
        INTERRUPTING_MASK register.

   NO ERRORS DETECTED while testing the HP-IB Interface Chip's
        HP-IB Interface Chip CONTROL register.

   NO ERRORS DETECTED while testing the HP-IB Interface Chip's
        ADDRESS register.

   NO ERRORS DETECTED while testing the HP-IB Interface Chip's
        PARALLEL_POLL_MASK/FIRST_ID_BYTE register.
   NO ERRORS DETECTED while testing the HP-IB Interface Chip's
        PARALLEL_POLL_SENSE/SECOND_ID_BYTE register.

   NO ERRORS DETECTED during the "HPIB_STATUS Register" Test
   NO ERRORS DETECTED during the "IFC" Test
   NO ERRORS DETECTED during the "Talk/Listen" Test
   NO ERRORS DETECTED during the "REN" Test
   NO ERRORS DETECTED during the "FIFO" Test
   NO ERRORS DETECTED during the "SRQ" Test
   NO ERRORS DETECTED during the "Parallel Poll" Test
   NO ERRORS DETECTED during the "Secondary Address" Test
   NO ERRORS DETECTED during the "CRC" Test
   NO ERRORS DETECTED "DMA Test Number 1"
   NO ERRORS DETECTED "DMA Test Number 2"
   NO ERRORS DETECTED "DMA Test Number 3"
   NO ERRORS DETECTED "DMA Test Number 4"
   NO ERRORS DETECTED "DMA Test Number 5"
   NO ERRORS DETECTED "DMA Test Number 6"
   NO ERRORS DETECTED "DMA Test Number 7"
   NO ERRORS DETECTED "DMA Test Number 8"
   NO ERRORS DETECTED the "GET/HP-IB Interface Chip Interrupt Test"
   NO ERRORS DETECTED the "On-Line" Test
   NO ERRORS DETECTED the "IFC Interrupt" Test

End of Section 5 -- Hardware Test
```

**Possible Error/Warning Messages:**

```
*** WARNING -- The device adapter under test is NOT the System
***              Controller, therefore the "On-Line Test" cannot be
***              executed.                                       (HDADWARN 6015)

*** WARNING -- The device adapter under test is NOT the System
***              Controller, therefore the "IFC Interrupt Test" cannot
***              be executed.                                    (HDADWARN 6016)

*** ERROR -- PB INTERFACE CHIP (PBIC) TEST FAILED.             (HDADERR 5030)
***          ! register failed the "Register Verification" test.
***
***          Data written to PBIC (in hex)       Data read from PBIC (in hex)
***          -----------------------------       ----------------------------

*** ERROR -- HP-IB INTERFACE CHIP (HIC) TEST FAILED.           (HDADERR 5031)
***          ! register failed the
***          "Register Verification" test.
***
***          Data written to HIC (in hex)        Data read from HIC (in hex)
***          ----------------------------        ---------------------------

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5035)
***          The interface chip "IFC" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5036)
***          The interface chip "Talk/Listen" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5037)
***          The interface chip "REN" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5038)
***          The interface chip "FIFO" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5039)
***          The interface chip "SRQ" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5040)
***          The interface chip "Parallel Poll" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5041)
***          The interface chip "Secondary Address" test failed.

*** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                 (HDADERR 5042)
***          The interface chip "CRC" test failed.

*** ERROR -- HPIB_CONTROL/STATUS REGISTER SET VERIFICATION
***          TEST FAILED.                                      (HDADERR 5045)

*** ERROR -- GROUP EXECUTE TRIGGER OR HP-IB INTERFACE CHIP'S INTERRUPT
```

```
***          CIRCUITRY FAILED TEST.                              (HDADERR 5046)

*** ERROR -- ON-LINE TEST FAILED.                                (HDADERR 5047)
***
***          This test is targeted to check the frontplane's General Interface
***          Management Lines (control transceiver), and BUS_STATUS register.

*** ERROR -- IFC INTERRUPT TEST FAILED.                          (HDADERR 5048)
***
***          This test is targeted to check the frontplane's IFC
***          interrupt circuitry.

*** ERROR -- DMA TEST NUMBER ! FAILED.                           (HDADERR 5049)
***          Data was corrupted.

*** ERROR -- DMA TEST NUMBER ! FAILED.                           (HDADERR 5050)
***          DMA termination condition incorrect.

*** ERROR -- DMA TEST NUMBER ! FAILED.                           (HDADERR 5051)
***          Incorrect residue value in the IO_DMA_COUNT register.

*** ERROR -- DMA TEST NUMBER ! FAILED.                           (HDADERR 5052)
***          Internal Software Error.

*** ERROR -- DMA TEST NUMBER ! FAILED.                           (HDADERR 5053)
***          Inconsistency within PB Interface Chip with respect to
***          reporting "DMA length conflict" status.
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_WRITE_REG
HPIB_READ_REG
HPIB_OUTPUT
HPIB_INPUT
HPIB_RESET
HPIB_UNLOCK
```

## Section 6—Status

```
Minimum Mode Required        : Destructive

Terminal Used for Execution : Any

In Default Set?              : Yes

In Auto-diagnostic Set?      : No
```

This section is broken down into seven steps so that the diagnostician can convey to HPIBDAD precisely what status information he/she wishes to examine. Moreover, the diagnostician can have a very thorough status report of the HP-IB Device Adapter if he/she desires it by executing all seven steps. In order to make this ample supply of status information easier for the diagnostician to assimilate, it is fully decoded and transformed into user-friendly messages by HPIBDAD before being displayed to the diagnostic user.

If this section is run without the desired steps being explicitly specified, Steps 61, 66, and 67 will run by default.

## Step 61—Preliminary Internal Status Diagnosis

Executing this step instructs the diagnostic to gather and interpret the necessary data in order to report a preliminary device adapter internal state diagnosis. The information gathering portion of this step consists of Device Access Routine (DAR) calls—register reads—which return the contents of the following registers:

- PB Interface Chip IO_STATUS Register
- PB Interface Chip DIAGNOSTIC_STATUS Register
- HP-IB Interface Chip INTERRUPTING_CONDITIONS Register
- HP-IB Interface Chip INTERRUPT_MASK Register

The interpretation phase of this data is considerably more complex and consists primarily of the following functions:

- Decode and report (in a user-friendly fashion) the information maintained by the various registers.
- If more than one register reports the same status condition (e.g. an interrupt is requested), `HPIBDAD` will compare the respective bits of those registers to check status consistency within the device adapter. Any inconsistency will be reported to the user as an error condition.
- If an interrupt is requested by the frontplane interface chip, the diagnostic will determine the interrupt condition and report it. However, if `HPIBDAD` discovers that the particular interrupt should have been masked because the INTERRUPT_MASK register contains a "0" at the respective bit position, an error message will be issued.
- If an interrupt is not requested, but the contents of INTERRUPTING_CONDITIONS and INTERRUPT_MASK indicate an interrupt request should have been issued, an error message will be displayed to the user.

**Possible Output Messages:**

```
Section 6 -- Status

   Step 61 - Preliminary Internal Status Diagnosis

      PB Interface Chip IO_STATUS Register Data:

         An interrupt has NOT occurred since the last write to
         IO_COMMAND.                                     (Bit  0 = 0)

         An interrupt message HAS been sent since the last write
         to the IO_COMMAND register.                     (Bit  0 = 1)

         An interrupt message was NOT issued due to circuitry external
         to the PB Interface Chip.                       (Bit  2 = 0)

         An interrupt HAS been issued due to circuitry external to
         the PB Interface Chip.                          (Bit  2 = 1)


         Transfer NOT completed.                         (Bit  3 = 0)

         Transfer completed.                             (Bit  3 = 1)

         A soft error (length conflict) has NOT occurred since the
         last write to the IO_COMMAND register.          (Bit 22 = 0)

         A fatal error has NOT occurred since the last write to
         the IO_COMMAND register.                        (Bit 24 = 0)

         The device adapter is NOT ready for a new command.  (Bit 25 = 0)

         The device adapter IS ready for a new command.  (Bit 25 = 1)

         An interrupt message has NOT been sent since last
         ii_clear.                                       (Bit 26 = 0)

         An interrupt message HAS been sent since last
         ii_clear.                                       (Bit 26 = 1)

         The EOC bit of the IO_DMA_LINK register
         is NOT set.                                     (Bit 31 = 0)

         The EOC bit of the IO_DMA_LINK register IS set. (Bit 31 = 1)


      PB Interface Chip DIAGNOSTIC_STATUS Register Data:

         Circuitry external to the PB Interface Chip HAS issued
           an interrupt request.                         (Bit  0 = 0)
```

```
   Circuitry external to the PB Interface Chip has NOT
     issued an interrupt request.                         (Bit  0 = 1)

   The PB Interface Chip did NOT generate a PB error since the
     last reset command.                                  (Bit  1 = 1)

   DMA is DISABLED.                                       (Bit  4 = 0)
   DMA is ENABLED.                                        (Bit  4 = 1)

   Interrupt message transmission is DISABLED.            (Bit 19 = 0)
   Interrupt message transmission is ENABLED.             (Bit 19 = 1)

   The FIFO is NOT FULL.                                  (Bit 20 = 0)
   The FIFO is FULL.                                      (Bit 20 = 1)

   The FIFO is NOT EMPTY.                                 (Bit 21 = 0)
   The FIFO is EMPTY.                                     (Bit 21 = 1)

   There is 1 byte in the FIFO.                           (Bits 23..27)
   There are ! bytes in the FIFO.                         (Bits 23..27)


HP-IB Interface Chip INTERRUPTING_CONDITIONS Register Data:

   An interrupt is NOT pending.                           (Bit  0 = 0)
   An interrupt IS pending.                               (Bit  0 = 1)

   "Parity Error" interrupt recorded.                     (Bit  1 = 1)

   "Status Change" interrupt recorded.                    (Bit  8 = 1)

   "Processor Handshake Abort" interrupt recorded.        (Bit  9 = 1)

   "Parallel Poll Response" interrupt recorded.           (Bit 10 = 1)

   "Service Request" interrupt recorded.                  (Bit 11 = 1)

   "FIFO Room Available" interrupt recorded.              (Bit 12 = 1)

   "FIFO Byte Available" interrupt recorded.              (Bit 13 = 1)

   "FIFO Idle" interrupt recorded.                        (Bit 14 = 1)

   "Device Clear" interrupt recorded.                     (Bit 15 = 1)


HP-IB Interface Chip INTERRUPT_MASK Register data:

   Interrupts are DISABLED.                               (Bit  0 = 0)
   Interrupts are ENABLED.                                (Bit  0 = 1)
```

```
"Parity Error" interrupt MASKED.                     (Bit  1 = 0)
"Parity Error" interrupt UNMASKED.                   (Bit  1 = 1)

"Status Change" interrupt MASKED.                    (Bit  8 = 0)
"Status Change" interrupt UNMASKED.                  (Bit  8 = 1)

"Processor Handshake Abort" interrupt MASKED.        (Bit  9 = 0)
"Processor Handshake Abort" interrupt UNMASKED.      (Bit  9 = 1)

"Parallel Poll Response" interrupt MASKED.           (Bit 10 = 0)
"Parallel Poll Response" interrupt UNMASKED.         (Bit 10 = 1)

"Service Request" interrupt MASKED.                  (Bit 11 = 0)
"Service Request" interrupt UNMASKED.                (Bit 11 = 1)

"FIFO Room Available" interrupt MASKED.              (Bit 12 = 0)
"FIFO Room Available" interrupt UNMASKED.            (Bit 12 = 1)

"FIFO Byte Available" interrupt MASKED.              (Bit 13 = 0)
"FIFO Byte Available" interrupt UNMASKED.            (Bit 13 = 1)

"FIFO Idle" interrupt MASKED.                        (Bit 14 = 0)
"FIFO Idle" interrupt UNMASKED.                      (Bit 14 = 1)

"Device Clear" interrupt MASKED.                     (Bit 15 = 0)
"Device Clear" interrupt UNMASKED.                   (Bit 15 = 1)
```

```
A binary representation of the respective registers follows:
NOTE: Dashes represent undefined bits.

   --------------------------------------------------------
   IO_STATUS register's image:

   Bit #:     0    4    8    12   16   20   24   28

   Value:     #### #### #### #### #### #### #### ####
   --------------------------------------------------------
   DIAGNOSTIC_STATUS register's image:

   Bit #:     0    4    8    12   16   20   24   28

   Value:     #### #### #### #### #### #### #### ####
   --------------------------------------------------------
   INTERRUPTING_CONDITIONS register's image:

   Bit #:     0    4    8    12   16   20   24   28

   Value:     #### #### #### #### #### #### #### ####
   --------------------------------------------------------
```

```
INTERRUPT_MASK register's image:

Bit #:     0     4     8     12    16    20    24    28

Value:     ####  ####  ####  ####  ####  ####  ####  ####
---------------------------------------------------------

End of Step 61 - Preliminary Internal State Diagnosis

End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

```
*** WARNING -- The IO_STATUS register has one or more undefined bits
***            reading in as a 1 (I expected 0's).          (HDADWARN 6020)

*** WARNING -- A SOFT ERROR (length conflict) occurred since the last
***            write to the IO_COMMAND register.
***            (IO_STATUS Bit 22 = 1)                       (HDADWARN 6022)

*** WARNING -- A FATAL ERROR occurred since the last write to the
***            IO_COMMAND register.
***            (IO_STATUS Bit 24 = 1)                       (HDADWARN 6023)

*** WARNING -- PB Interface Chip HAS generated a PB bus error since
***            the last reset command.
***            (DIAGNOSTIC_STATUS Bit 1 = 0)                (HDADWARN 6025)


*** ERROR -- PB INTERFACE CHIP FAILURE.                     (HDADERR 5060)
***
***          PB Interface Chip reported to have ! bytes in FIFO,
***          however only 24 bytes are available.

*** ERROR -- PB INTERFACE CHIP FAILURE.                     (HDADERR 5061)
***
***          DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) is set
***          (implying FIFO CNT should = 24); however, the FIFO CNT field
***          (Bits 23..27) = ! (decimal).

*** ERROR -- PB INTERFACE CHIP FAILURE.                     (HDADERR 5062)
***
***          DIAGNOSTIC_STATUS register's FIFO EMPTY bit (Bit 21) is set
***          (implying FIFO CNT should = 0); however, the FIFO CNT field
***          (Bits 23..27) = ! (decimal).

*** ERROR -- PB INTERFACE CHIP FAILURE.                     (HDADERR 5063)
***
***          DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) and
***          FIFO EMPTY bit (Bit 21) are both set.
```

```
*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                     (HDADERR 5064)
***
***        THE "INTERRUPT PENDING" BIT WAS INADVERTENTLY SET:
***          The INTERRUPT PENDING bit (Bit 0) of
***          INTERRUPTING_CONDITIONS = 1; however, interrupts are
***          disabled, depicted by the INTERRUPT ENABLE bit (Bit 0) of
***          INTERRUPT_MASK = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                     (HDADERR 5065)
***
***        AN INTERRUPTING CONDITION BIT WAS INADVERTENTLY SET:
***         One or more bits of INTERRUPTING_CONDITIONS = 1, even though the
***         interrupts have been masked, depicted by the respective bits of
***         INTERRUPT_MASK = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                     (HDADERR 5066)
***
***        THE "INTERRUPT PENDING" BIT FAILED TO BE SET:
***          One or more bits of INTERRUPTING_CONDITIONS and Bit 0 of
***          INTERRUPTING_MASK = 1; however, the INTERRUPT PENDING bit
***          (Bit 0) of INTERRUPTING_CONDITIONS = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                     (HDADERR 5067)
***
***        THE "INTERRUPT PENDING" BIT WAS INADVERTENTLY SET:
***          No interrupting condition bit is set, however the
***          INTERRUPT PENDING bit (Bit 0) of INTERRUPT_CONDITIONS = 1.

*** ERROR -- INCONSISTENCY IN INTERRUPT STATUS.                (HDADERR 5068)
***
***        The "INTERRUPT PENDING" bit (Bit 0) of
***        INTERRUPTING_CONDITIONS = 1;
***        however, the interrupt input bit of the DIAGNOSTIC_STATUS
***        register (Bit 0) = 1, which indicates that the PB Interface
***        Chip did not recognize the interrupt request.

*** ERROR -- PB INTERFACE CHIP FAILURE.                        (HDADERR 5069)
***
***        DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) = 0
***        (implying FIFO CNT should be less than 24); however, the
***        FIFO CNT field (Bits 23..27) = ! (decimal).

*** ERROR -- PB INTERFACE CHIP FAILURE.                        (HDADERR 5070)
***
***        DIAGNOSTIC_STATUS register's FIFO EMPTY bit (Bit 21) = 0
***        (implying FIFO CNT should be greater than 0); however, the
***        FIFO CNT field (Bits 23..27) = ! (decimal).
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 62—Read HP-IB Interface Chip STATUS Register

Executing this step will return to the user various status conditions of the HP-IB Interface Chip. Once the HP-IB Interface Chip's STATUS register contents have been decoded by HPIBDAD, user-friendly output illustrating the current status information will be displayed.

---

**Note**

CONTROL and ADDRESS registers will need to be accessed in order to fully decode the STATUS register. This is because the on-line/off-line status (within the ADDRESS register) is needed to decode the HP-IB SYSTEM CONTROLLER bit (Bit 12). Also, the TALK ALWAYS and LISTEN ALWAYS bits within the ADDRESS register are needed for error detection purposes. The status of the IFC VALUE bit within the CONTROL register is needed for error detection concerning the HP-IB CONTROLLER bit (Bit 11).

In addition to the frontplane interface chip's own registers required to fully decode this register, the Device Adapter STATUS register is required in order to detect faults within the the frontplane interface chip. Specifically, the SYSTEM CNTL bit of the Device Adapter STATUS register is required to determine whether or not the HP-IB SYSTEM CONTROLLER bit of this register should be set.

---

**Possible Output Messages:**

```
Section 6 -- Status

   Step 62 - Read HP-IB Interface Chip STATUS Register

      High order access field:                            (Bit  8 = !)
                                                          (Bit  9 = !)

      Device Adapter is NOT in the remote state.          (Bit 10 = 0)
      Device Adapter IS in the remote state.              (Bit 10 = 1)

      Device Adapter is NOT the current HP-IB Controller.  (Bit 11 = 0)
      Device Adapter IS the current HP-IB Controller.      (Bit 11 = 1)

      Device Adapter is NOT the HP-IB System Controller.   (Bit 12 = 0)
      Device Adapter IS the HP-IB System Controller.       (Bit 12 = 1)
      Device Adapter IS the HP-IB System Controller, however
        it is also OFF-LINE.                               (Bit 12 = 1)

      Device Adapter has NOT been addressed to Talk OR
        to Identify.                                       (Bit 13 = 0)
      Device Adapter HAS been addressed to Talk OR Identify. (Bit 13 = 1)

      Device Adapter has NOT been addressed to Listen.     (Bit 14 = 0)
      Device Adapter HAS been addressed to Listen.         (Bit 14 = 1)

      Device Adapter's outbound data is NOT frozen.        (Bit 15 = 0)
      Device Adapter's outbound data IS frozen.            (Bit 15 = 1)

      A binary representation of the respective register follows:
      NOTE: Dashes represent undefined bits.

         ---------------------------------------------------------
         HP-IB Interface Chip STATUS register's image:

         Bit #:     0    4    8    12   16   20   24   28

         Value:     #### #### #### #### #### #### #### ####
         ---------------------------------------------------------

   End of Step 62 - Read HP-IB Interface Chip STATUS Register

End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

```
*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5080)
***
***         HP-IB SYSTEM CONTROLLER bit (Bit 12) should be set because
***         the HP-IB Interface Chip is off-line; however, Bit 12 = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5081)
***
***         HP-IB CONTROLLER bit (Bit 11) should be set because the device
***         adapter IS the System Controller and it has asserted the IFC line;
***         however, Bit 11 = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5082)
***
***         ADDRESSED TO TALK OR IDENTIFY bit (Bit 13) should be set because
***         the TALK ALWAYS bit is set (within the ADDRESS register);
***         however, Bit 13 = 0.

*** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5083)
***
***         ADDRESSED TO LISTEN bit (Bit 14) should be set because the
***         LISTEN ALWAYS bit is set (within the ADDRESS register);
***         however, Bit 14 = 0.

*** ERROR -- INCONSISTENCY IN SYSTEM CONTROLLER STATUS.       (HDADERR 5084)
***
***         The SYSTEM CNTL bit within the HPIB_STATUS register
***         is reset, and the HP-IB Interface Chip is on-line;
***         however, Bit 12 = 1.

*** ERROR -- INCONSISTENCY IN SYSTEM CONTROLLER STATUS.       (HDADERR 5085)
***
***         The SYSTEM CNTL bit within the HPIB_STATUS register
***         is set; however, Bit 12 = 0.
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 63—Read HP-IB Interface Chip CONTROL Register

This step will retrieve the contents of the frontplane interface chip's CONTROL Register and decode the respective control bits in order to display user-friendly messages for the user.

**Note**

The frontplane interface chip's STATUS and ADDRESS registers will need to be accessed in order to fully decode the CONTROL register. This is because it must be known whether or not the device adapter is the System Controller (and whether the card is on-line or off-line) in order to decode the REN VALUE and IFC VALUE bits. The OUTBOUND DATA FREEZE bit in the STATUS register is needed to fully decode the implications of the RESPOND TO PARALLEL POLL bit.

**Possible Output Messages:**

```
Section 6 -- Status

   Step 63 - Read HP-IB Interface Chip CONTROL Register

      The OUTBOUND DATA FREEZE restriction is NOT in affect on the
        Parallel Poll response.                            (Bit  0 = 0)
      The OUTBOUND DATA FREEZE restriction IS in affect on the
        Parallel Poll response.                            (Bit  0 = 1)

      NDAC or NRFD signals are NOT delayed to the Source Handshake
        circuitry during DATA transfers.                   (Bit  1 = 0)
      NDAC or NRFD signals ARE delayed to the Source Handshake
        circuitry during DATA transfers.                   (Bit  1 = 1)

      Currently utilizing the standard 10-bit data path.   (Bit  8 = 0)
      Currently utilizing the 8-bit data path.             (Bit  8 = 1)

      Even parity Interface Commands ARE accepted.         (Bit  9 = 0)
      Even parity Interface Commands are NOT accepted.     (Bit  9 = 1)

      The REN line is at a logic LOW state.                (Bit 10 = 0)
      The local REN line is at a logic LOW state (the HP-IB
        Interface Chip is off-line).                       (Bit 10 = 0)
      The REN line is at a logic HIGH state.               (Bit 10 = 1)
      The local REN line is at a logic HIGH state (the HP-IB
        Interface Chip is off-line).                       (Bit 10 = 1)
      Decoding the REN VALUE bit is noninformative because the
        device adapter is not the System Controller.       (Bit 10)

      The IFC line is at a logic LOW state.                (Bit 11 = 0)
      The local IFC line is at a logic LOW state (the HP-IB
        Interface Chip is off-line).                       (Bit 11 = 0)
      The IFC line is at a logic HIGH state.               (Bit 11 = 1)
      The local IFC line is at a logic HIGH state (the HP-IB
        Interface Chip is off-line).                       (Bit 11 = 1)
```

```
Decoding the IFC VALUE bit is noninformative because the
  device adapter is not the System Controller.        (Bit 11)

The RESPOND TO PARALLEL POLL bit = 0; therefore the HP-IB
  Interface Chip is not in need of service.           (Bit 12 = 0)
The RESPOND TO PARALLEL POLL bit = 1; therefore the HP-IB
  Interface Chip will indicate the need for service during
  any Parallel Poll if it has the response capability. (Bit 12 = 1)
The RESPOND TO PARALLEL POLL bit = 1; however, since both
  the OUTBOUND DATA FREEZE and POLL HOLDOFF bits are set,
  the HP-IB Interface Chip will NOT respond affirmatively
  to a Parallel Poll.                                 (Bit 12 = 1)

The REQUEST SERVICE bit = 0; therefore the HP-IB
  Interface Chip will NOT request service during the
  next Serial Poll.                                   (Bit 13 = 0)
The REQUEST SERVICE bit = 1; depicting that the HP-IB
  Interface Chip has asserted the SRQ line and during the
  next Serial Poll will request service from the
  HP-IB Controller.                                   (Bit 13 = 1)

A DMA request will be issued when the Inbound FIFO is
  ready for a READ operation.                         (Bit 14 = 0)
A DMA request will be issued when the Outbound FIFO is
  ready for a WRITE operation.                        (Bit 14 = 1)

The INITIALIZE OUTBOUND FIFO bit is not architected to
  be read (0 is always returned).                     (Bit 15 = 0)

A binary representation of the respective register follows:
NOTE: Dashes represent undefined bits.

   ---------------------------------------------------------
   HP-IB Interface Chip CONTROL register's image:

   Bit #:     0    4    8    12   16   20   24   28

   Value:     #### #### #### #### #### #### #### ####
   ---------------------------------------------------------

End of Step 63 - Read HP-IB Interface Chip CONTROL Register

End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

```
*** WARNING -- The INITIALIZE OUTBOUND FIFO bit value was read in as
***            being 1, but should always be read as 0.        (HDADWARN 6030)
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 64—Read HP-IB Interface Chip ADDRESS Register

When this step is run, HPIBDAD will read, decode, and display the contents of the HP-IB Interface Chip's ADDRESS register. The information maintained by this register pertains to the HP-IB address of the device adapter, as well as related control information.

**Note**

In order to decode the full implications of the TALK ALWAYS and LISTEN ALWAYS bits, the frontplane interface chip's STATUS register must be accessed. Specifically, the ADDRESSED TO TALK and ADDRESSED TO LISTEN bits must be retrieved.

**Possible Output Messages:**

```
Section 6 -- Status

   Step 64 - Read HP-IB Interface Chip ADDRESS Register

      The CRC capabilities are DISABLED.                    (Bit  0 = 0)
      The CRC capabilities are ENABLED.                     (Bit  0 = 1)

      HP-IB commands originating from the Outbound FIFO will
        have ODD parity.                                    (Bit  1 = 0)
      HP-IB commands originating from the Outbound FIFO will
        have EVEN parity.                                   (Bit  1 = 1)

      The HP-IB Interface Chip is OFF-LINE.                 (Bit  8 = 0)
      The HP-IB Interface Chip is ON-LINE.                  (Bit  8 = 1)

      The TALK ALWAYS bit is RESET.                         (Bit  9 = 0)
      The TALK ALWAYS bit is reset; however, the device
        adapter IS addressed to Talk.                       (Bit  9 = 0)
      The TALK ALWAYS bit is SET.                           (Bit  9 = 1)

      The LISTEN ALWAYS bit is RESET.                       (Bit 10 = 0)
      The LISTEN ALWAYS bit is reset; however, the device
        adapter IS addressed to Listen.                     (Bit 10 = 0)
      The LISTEN ALWAYS bit is SET.                         (Bit 10 = 1)

      The HP-IB address for the device adapter
        is as follows (in decimal): !                       (Bits 11..15)
```

```
        A binary representation of the respective register follows:
        NOTE: Dashes represent undefined bits.

           --------------------------------------------------------
           ADDRESS register's image:

           Bit #:      0     4     8    12    16    20    24    28

           Value:     ####  ####  ####  ####  ####  ####  ####  ####
           --------------------------------------------------------

     End of Step 64 - Read HP-IB Interface Chip ADDRESS Register

  End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

```
None specified by the diagnostic.
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 65—Read HP-IB Interface Chip PP/ID_BYTE Registers

Executing this step will result in HPIBDAD displaying the contents of the PARALLEL_POLL_MASK/FIRST_ID_BYTE and PARALLEL_POLL_SENSE/SECOND_ID_BYTE registers to the diagnostician. The respective register's contents will not be decoded and displayed in a user-friendly fashion for the user in this step, but rather simply displayed as a binary bit pattern. The reason being that decoding the bits into user-friendly messages would actually be encoding the bits into "user-frustrating" messages. The double role of the two registers also adds to the argument for not decoding the bits, since the diagnostic cannot be sure of the registers' current role (i.e. which decode template to utilize when displaying the contents).

**Possible Output Messages:**

```
Section 6 -- Status

   Step 65 - Read HP-IB Interface Chip PP/ID_BYTE Registers

      A binary representation of the respective registers follows:
      NOTE: Dashes represent undefined bits.

         --------------------------------------------------------
         PARALLEL_POLL_MASK/FIRST_ID_BYTE register's image:

         Bit #:     0     4     8     12    16    20    24    28

         Value:     ####  ####  ####  ####  ####  ####  ####  ####
         --------------------------------------------------------
         PARALLEL_POLL_SENSE/SECOND_ID_BYTE register's image:

         Bit #:     0     4     8     12    16    20    24    28

         Value:     ####  ####  ####  ####  ####  ####  ####  ####
         --------------------------------------------------------

   End of Step 65 - Read HP-IB Interface Chip PP/ID_BYTE Registers

End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

None specified by the diagnostic.

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 66—Read HPIB_STATUS Register

This step is used to obtain information about the current status (configuration) of the HP-IB and the HP-IB Device Adapter Talker/Listener/Controller Chip . This information is maintained in the read-only HPIB_STATUS register on the HP-IB DA. After this register is read in by the diagnostic program, the respective bits are decoded and displayed to the operator in a user-friendly manner.

**Note** In order to determine the consistency of reporting pending interrupts, the HP-IB Interface Chip's INTERRUPTING_CONDITIONS register will have to be read in.

**Possible Output Messages:**

```
Section 6 -- Status

   Step 66 - Read HPIB_STATUS Register

      HP-IB Interface Chip is NOT the System Controller.    (Bit 24 = 0)
      HP-IB Interface Chip IS the System Controller.        (Bit 24 = 1)

      HP-IB Interface Chip is in SLOW mode.                 (Bit 25 = 0)
      HP-IB Interface Chip is in FAST mode.                 (Bit 25 = 1)

      HP-IB Interface Chip's D1 value from the last read
        operation was 0.                                    (Bit 26 = 0)
      HP-IB Interface Chip's D1 value from the last read
        operation was 1.                                    (Bit 26 = 1)

      HP-IB Interface Chip's DO value from the last read
        operation was 0.                                    (Bit 27 = 0)
      HP-IB Interface Chip's DO value from the last read
        operation was 1.                                    (Bit 27 = 1)

      HP-IB Interface Chip is NOT interrupting.             (Bit 28 = 0)
      HP-IB Interface Chip IS interrupting.                 (Bit 28 = 1)

      Group Execute Trigger has NOT interrupted.            (Bit 29 = 0)
      Group Execute Trigger HAS interrupted.                (Bit 29 = 1)

      Interface Clear has NOT interrupted.                  (Bit 30 = 0)
      Interface Clear HAS interrupted.                      (Bit 30 = 1)

      Hardware did NOT pass most recent test.               (Bit 31 = 0)
      Hardware PASSED most recent test.                     (Bit 31 = 1)


      A binary representation of the respective register follows:
      NOTE: Dashes represent undefined bits.
```

```
--------------------------------------------------------
HPIB_STATUS register's image:

Bit #:     0     4     8     12    16    20    24    28

Value:     ####  ####  ####  ####  ####  ####  ####  ####
--------------------------------------------------------
```

End of Step 66 - Read HPIB_STATUS Register

End of Section 6 -- Status

**Possible Error/Warning Messages:**

```
*** ERROR -- INCONSISTENCY IN INTERRUPT PENDING STATUS.        (HDADERR 5090)
***
***            INTERRUPTING_CONDITIONS register: bit  0 = 1
***            HPIB_STATUS register            : bit 28 = 0

*** ERROR -- INCONSISTENCY IN INTERRUPT PENDING STATUS.        (HDADERR 5091)
***
***            INTERRUPTING_CONDITIONS register: bit  0 = 0
***            HPIB_STATUS register            : bit 28 = 1
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Step 67—Read BUS_STATUS Register

This step is used to obtain information about the current state of the HP-IB control bus (5 general interface management lines and 3 handshake lines) by reading in the contents of the BUS_STATUS Register. The image of the HP-IB control bus is latched into this register with the leading edge of the select signal for the register. The output of this step is the fully decoded representation of the HP-IB status.

---

**Note**

Reading this register is an effective way to determine if there is any device in handshake mode connected to the HP-IB. The reason for this is that when a handshake device is in fact connected to the HP-IB, at least one of the following signals will be at the logic zero level: NDAC, NRFD. When there are no handshake devices connected to the HP-IB, these signals will both be at the logic one level.

---

**Possible Output Messages:**

```
Section 6 -- Status

   Step 67 - Read BUS_STATUS Register

      The logic levels of the respective bits are as follows:

                      Line ID                 Logic Level
                      -------                 -----------
         EOI - End or Identify    (Bit 24):        #
         REN - Remote Enable      (Bit 25):        #
         SRQ - Service Request    (Bit 26):        #
         ATN - Attention          (Bit 27):        #
         IFC - Interface Clear    (Bit 28):        #

         DAV - Data valid         (Bit 29):        #
        NDAC - Not Data Accepted  (Bit 30):        #
        NRFD - Not Ready For Data (Bit 31):        #

      A binary representation of the respective register follows:
      NOTE: Dashes represent undefined bits.

         ---------------------------------------------------------
         BUS_STATUS register's image:

         Bit #:     0     4     8     12    16    20    24    28

         Value:     ####  ####  ####  ####  ####  ####  ####  ####
         ---------------------------------------------------------


   End of Step 67 - Read BUS_STATUS Register

End of Section 6 -- Status
```

**Possible Error/Warning Messages:**

```
*** WARNING -- No device in handshake mode is connected to the HP-IB,
***            depicted by the NDAC and NRFD lines both being at the
***            logic one level.                                  (HDADWARN 6050)
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_READ_REG
HPIB_UNLOCK
```

## Section 10—Register Level Input/Output Transactions

```
Minimum Mode Required        : Destructive

Terminal Used for Execution  : Any

In Default Set?              : No

In Auto-diagnostic Set?      : No
```

**Warning** **The user of this section must exercise EXTREME CAUTION when sending information over the HP-IB. If a negligent user sends "garbage" information to a system disk, the result could be a corrupted disk and a system crash!**

This section can be executed by the diagnostician when he/she wishes to do "peeks" and/or "pokes" to the device adapter. That is to say, this section allows the user to read/write any register in the device adapter's address space.

**Warning** **In order for this section to be of any value, the device adapter being diagnosed should not be accessed by any process other than HPIBDAD. This is to ensure that any configuration modifications made by the diagnostic are pending for succeeding transactions.**

**Possible Output Messages:**

```
Section 10 -- Register Level Input/Output Transactions

   At the prompt, enter one of the following commands:
     1) "Input Register" Command:

             i <decimal register number>

        For Example:
          To input register number 159, type "i 159<CR>"

     2) "Output Register" Command:

             o <decimal register number> <hex data>

        For Example:
          To output 0x02BAD to register number 12 (decimal), type
          "o 12 2bad<CR>"

     3) A lone <CR> will cause this section to terminate.

   %

    : !

End of Section 10 -- Register Level Input/Output Transactions
```

**Note**

All commands/data can be entered in upper and/or lower case characters.

**Possible Error/Warning Messages:**

```
*** WARNING -- This section is intended for personnel which have a good
***            understanding of the HP-IB Device Adapter's architecture.
***            It is strongly recommended that you print off "More Help" on
***            Section 10, or reference the user's manual before using this
***            section.                                        (HDADWARN 6060)

*** WARNING -- Invalid or missing register number ... the register number
***            value is not a properly formatted DECIMAL integer.
***            Please try again.                               (HDADWARN 6061)

*** WARNING -- Invalid register number ... the register number entered
***            is beyond the device adapter's address space (highest
***            register number is decimal 1023).
***            Please try again.                               (HDADWARN 6062)

*** WARNING -- Invalid or missing datum ... the datum value is not a properly
***            formatted HEXADECIMAL integer.
***            Please try again.                               (HDADWARN 6064)

*** WARNING -- Invalid command ... the command entered cannot be decoded.
***            Please try again.                               (HDADWARN 6065)
```

**HP-IB DAR Operations Used:**

```
HPIB_LOCK
HPIB_WRITE_REG
HPIB_READ_REG
HPIB_UNLOCK
```

## Section 11—Data/Command Transaction on HP-IB

```
Minimum Mode Required        : Destructive

Terminal Used for Execution  : Any

In Default Set?              : No

In Auto-diagnostic Set?      : No
```

Executing this section will allow the user to send data/commands to a particular device connected to the interface bus, as well as receive data from the device. Essentially, this section gives the user complete control of the information to be sent to the frontplane interface chip's Register 2 (Outbound FIFO). This implies that the user must have a very good working knowledge of the HP-IB protocol; however, examples of some of the more common transactions can be observed by running Section 1 (**More Help**) of this diagnostic and entering "11" at the prompt.

When this section is run, data entered is expected to be in hexadecimal format. Also, the user may restart information entry by entering a lone semicolon (;) at any prompt. Entering a lone <CR> will terminate information entry, whereas entering a lone colon (:) terminates data entry AND conveys to the diagnostic that return data is expected from a device on the HP-IB. When expected data is returned within the timeout period, it will be displayed in hexadecimal format; however, if the timeout period elapses, an appropriate error message will be output.

In order to make this section a little more beneficial to the user, the HPIBDAD will take care of issuing the necessary (Un)Talk/(Un)Listen addressing. Therefore, the user will be required to simply enter the address of the device he/she wishes to communicate with in addition to the data to be sent.

The user should make sure that if a command is being sent to a device, that it is "terminated" according to the protocol of the particular device. For example, some devices accept a semicolon (;) as a command delimiter, others expect the EOI line to be asserted, etc. (to assert the EOI line, the number two (2) should be inserted as the most significant digit into the byte where the EOI is desired; e.g. if an EOI is desired while sending the hex byte 49, the user should enter the hex datum of 249—this sets the END bit in the frontplane interface chip's Outbound FIFO).

**Note**

In order for this diagnostic section to execute properly, the HP-IB Device Adapter being diagnosed must be the System Controller. Therefore, the HPIB_STATUS register will be read in at the onset of this section in order to verify the System Controller status. If the device adapter under test is not the System Controller, a warning message will be output and the section will be exited. However, it is not expected that this scenario will often occur, since the device adapter under test will usually be the System Controller.

Also note that the REN (Remote Enable) Line will be asserted on the DA under test by HPIBDAD before the data/command transaction takes place on the HP-IB.

**Caution**

A timeout error condition may exist if the device adapter is not able to send data/commands over the HP-IB due to a "configuration" problem (e.g. DA is neither the HP-IB Controller nor addressed to talk). Therefore, in this type of situation, there actually isn't a hardware/software problem, but the user may infer this from the error message. Similarly, the user may have requested information to be returned from a device on the HP-IB, but that device may be very slow in collecting the requisite data, therefore causing an error message to be output when an actual hardware/software error may not exist.

**Warning**

**The user of this section must exercise EXTREME CAUTION when sending information over the HP-IB. If a negligent user sends "garbage" information to a system disk, the result could be a corrupted disk and a system crash!**

**Possible Output Messages:**

```
Section 11 -- Data/Command Transaction on HP-IB

   Example scripts can be observed by executing Section 1 -- More Help
   and entering "11" at the prompt.

   Please enter the HP-IB address of the device you wish to
   transact with (0..29) :

   At a prompt, enter one of the following:
     1) A hexadecimal integer (10-bit maximum per line--i.e. 0..3FF)
     2) A lone <CR> to terminate information entry
     3) A lone colon (:) to terminate information entry AND convey to the
        diagnostic that return data is expected
     4) A lone semicolon (;) to abort the current information entry session
        and to start over

   CAUTION: Care must be taken not to enter a hexadecimal integer that
   matches a language localized control message.  For example, if a lone
   'e' is entered, it is an indication to the diagnostic to exit.  In this
   case, "0e" should be entered.

   >>

   The device RETURNED the following data (displayed in hex format):

   ... Information entry to be restarted.

   Information being sent to HP-IB Interface Chip's Outbound FIFO ...

End of Section 11 -- Data/Command Transaction on HP-IB
```

**Possible Error/Warning Messages:**

```
*** ERROR -- THE NUMBER OF DATA BYTES RETURNED BY THE DEVICE EXCEEDED
***          THE SPECIFIED BYTE COUNT.                        (HDADERR 5100)
***
***          Byte count was set to: !
***          Data  bytes  returned: !

*** WARNING -- Invalid response.  Please answer the question with one
***            of the choices given.                          (HDADWARN 6001)

*** WARNING -- It is strongly recommended that you print off "More Help" on
***            Section 11, or reference the user's manual before using this
***            section.                                       (HDADWARN 6070)

*** WARNING -- The device adapter under test is NOT the System
***            Controller, therefore this section cannot be
***            executed.                                      (HDADWARN 6072)

*** WARNING -- Information entered is not a properly formatted
***            HEXADECIMAL integer.
***            Please try again.  For example: "2BE<CR>".     (HDADWARN 6075)

*** WARNING -- Information entered exceeds the 10-bit limit.
***            Please try again.  For example: "3FF<CR>".     (HDADWARN 6076)

*** WARNING -- The number of data units entered has exceeded the buffer size.
***            Please terminate the current data entry
***            when prompted.                                 (HDADWARN 6077)
```

**HP-IB DAR Operations Used:**

HPIB_LOCK
HPIB_WRITE_REG
HPIB_READ_REG
HPIB_INPUT
HPIB_RESET
HPIB_UNLOCK

## Error and Warning Messages

The following provides a listing, in numerical order, of the most significant error and warning messages displayed by the system.

### Error Messages

The following provides a listing, in numerical order, of the most significant error messages displayed by the system.

```
5005 *** ERROR -- IODC CHECKSUM FAILED.                          (HDADERR 5005)
```

---

```
5010 *** ERROR -- PB INTERFACE CHIP (PBIC) LOOPBACK FAILED.      (HDADERR 5010)
     ***
     ***       Data written to PBIC (in hex)      Data read from PBIC (in hex)
     ***       -----------------------------      ----------------------------
     ***                 !                                    !
```

---

```
5020 *** ERROR -- HP-IB INTERFACE CHIP (HIC) LOOPBACK FAILED.    (HDADERR 5020)
     ***
     ***       Data written to HIC (in hex)       Data read from HIC (in hex)
     ***       ----------------------------       ---------------------------
     ***                 !                                    !
```

---

```
5030 *** ERROR -- PB INTERFACE CHIP (PBIC) TEST FAILED.          (HDADERR 5030)
     ***       ! register failed the "Register Verification" test.
     ***
     ***       Data written to PBIC (in hex)      Data read from PBIC (in hex)
     ***       -----------------------------      ----------------------------
```

---

```
5031 *** ERROR -- HP-IB INTERFACE CHIP (HIC) TEST FAILED.          (HDADERR 5031)
     ***         ! register failed the
     ***         "Register Verification" test.
     ***
     ***         Data written to HIC (in hex)       Data read from HIC (in hex)
     ***         ----------------------------       ---------------------------
```

---

```
5035 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5035)
     ***         The interface chip "IFC" test failed.
```

---

```
5036 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5036)
     ***         The interface chip "Talk/Listen" test failed.
```

---

```
5037 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5037)
     ***         The interface chip "REN" test failed.
```

---

```
5038 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5038)
     ***         The interface chip "FIFO" test failed.
```

---

```
5039 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5039)
     ***         The interface chip "SRQ" test failed.
```

---

```
5040 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5040)
     ***         The interface chip "Parallel Poll" test failed.
```

---

```
5041 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.                (HDADERR 5041)
     ***         The interface chip "Secondary Address" test failed.
```

---

```
5042 *** ERROR -- HP-IB INTERFACE CHIP TEST FAILED.              (HDADERR 5042)
     ***         The interface chip "CRC" test failed.
```

---

```
5045 *** ERROR -- HPIB_CONTROL/STATUS REGISTER SET VERIFICATION
     ***         TEST FAILED.                                    (HDADERR 5045)
```

---

```
5046 *** ERROR -- GROUP EXECUTE TRIGGER OR HP-IB INTERFACE CHIP'S INTERRUPT
     ***         CIRCUITRY FAILED TEST.                          (HDADERR 5046)
```

---

```
5047 *** ERROR -- ON-LINE TEST FAILED.                            (HDADERR 5047)
     ***
     ***         This test is targeted to check the frontplane's General Interface
     ***         Management Lines (control transceiver), and BUS_STATUS register.
```

---

```
5048 *** ERROR -- IFC INTERRUPT TEST FAILED.                      (HDADERR 5048)
     ***
     ***         This test is targeted to check the frontplane's IFC
     ***         interrupt circuitry.
```

---

```
5049 *** ERROR -- DMA TEST NUMBER ! FAILED.                       (HDADERR 5049)
     ***         Data was corrupted.
```

---

```
5050 *** ERROR -- DMA TEST NUMBER ! FAILED.                       (HDADERR 5050)
     ***         DMA termination condition incorrect.
```

---

```
5051 *** ERROR -- DMA TEST NUMBER ! FAILED.                       (HDADERR 5051)
     ***         Incorrect residue value in the IO_DMA_COUNT register.
```

---

```
5052 *** ERROR -- DMA TEST NUMBER ! FAILED.                    (HDADERR 5052)
     ***         Internal Software Error.
```

---

```
5053 *** ERROR -- DMA TEST NUMBER ! FAILED.                    (HDADERR 5053)
     ***         Inconsistency within PB Interface Chip with respect to
     ***         reporting "DMA length conflict" status.
```

---

```
5060 *** ERROR -- PB INTERFACE CHIP FAILURE.                   (HDADERR 5060)
     ***
     ***         PB Interface Chip reported to have ! bytes in FIFO,
     ***         however only 24 bytes are available.
```

---

```
5061 *** ERROR -- PB INTERFACE CHIP FAILURE.                   (HDADERR 5061)
     ***
     ***         DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) is set
     ***         (implying FIFO CNT should = 24); however, the FIFO CNT field
     ***         (Bits 23..27) = ! (decimal).
```

---

```
5062 *** ERROR -- PB INTERFACE CHIP FAILURE.                   (HDADERR 5062)
     ***
     ***         DIAGNOSTIC_STATUS register's FIFO EMPTY bit (Bit 21) is set
     ***         (implying FIFO CNT should = 0); however, the FIFO CNT field
     ***         (Bits 23..27) = ! (decimal).
```

---

```
5063 *** ERROR -- PB INTERFACE CHIP FAILURE.                   (HDADERR 5063)
     ***
     ***         DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) and
     ***         FIFO EMPTY bit (Bit 21) are both set.
```

---

```
5064 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5064)
     ***
     ***        THE "INTERRUPT PENDING" BIT WAS INADVERTENTLY SET:
     ***          The INTERRUPT PENDING bit (Bit 0) of
     ***          INTERRUPTING_CONDITIONS = 1; however, interrupts are
     ***          disabled, depicted by the INTERRUPT ENABLE bit (Bit 0) of
     ***          INTERRUPT_MASK = 0.
```

---

```
5065 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5065)
     ***
     ***        AN INTERRUPTING CONDITION BIT WAS INADVERTENTLY SET:
     ***         One or more bits of INTERRUPTING_CONDITIONS = 1, even though the
     ***         interrupts have been masked, depicted by the respective bits of
     ***         INTERRUPT_MASK = 0.
```

---

```
5066 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5066)
     ***
     ***        THE "INTERRUPT PENDING" BIT FAILED TO BE SET:
     ***          One or more bits of INTERRUPTING_CONDITIONS and Bit 0 of
     ***          INTERRUPTING_MASK = 1; however, the INTERRUPT PENDING bit
     ***          (Bit 0) of INTERRUPTING_CONDITIONS = 0.
```

---

```
5067 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5067)
     ***
     ***        THE "INTERRUPT PENDING" BIT WAS INADVERTENTLY SET:
     ***          No interrupting condition bit is set, however the
     ***          INTERRUPT PENDING bit (Bit 0) of INTERRUPT_CONDITIONS = 1.
```

---

```
5068 *** ERROR -- INCONSISTENCY IN INTERRUPT STATUS.                 (HDADERR 5068)
     ***
     ***          The "INTERRUPT PENDING" bit (Bit 0) of
     ***          INTERRUPTING_CONDITIONS = 1;
     ***          however, the interrupt input bit of the DIAGNOSTIC_STATUS
     ***          register (Bit 0) = 1, which indicates that the PB Interface
     ***          Chip did not recognize the interrupt request.
```

---

```
5069 *** ERROR -- PB INTERFACE CHIP FAILURE.                          (HDADERR 5069)
     ***
     ***          DIAGNOSTIC_STATUS register's FIFO FULL bit (Bit 20) = 0
     ***          (implying FIFO CNT should be less than 24); however, the
     ***          FIFO CNT field (Bits 23..27) = ! (decimal).
```

---

```
5070 *** ERROR -- PB INTERFACE CHIP FAILURE.                          (HDADERR 5070)
     ***
     ***          DIAGNOSTIC_STATUS register's FIFO EMPTY bit (Bit 21) = 0
     ***          (implying FIFO CNT should be greater than 0); however, the
     ***          FIFO CNT field (Bits 23..27) = ! (decimal).
```

---

```
5080 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                       (HDADERR 5080)
     ***
     ***          HP-IB SYSTEM CONTROLLER bit (Bit 12) should be set because
     ***          the HP-IB Interface Chip is off-line; however, Bit 12 = 0.
```

---

```
5081 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5081)
     ***
     ***          HP-IB CONTROLLER bit (Bit 11) should be set because the device
     ***          adapter IS the System Controller and it has asserted the IFC line;
     ***          however, Bit 11 = 0.
```

---

```
5082 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5082)
     ***
     ***          ADDRESSED TO TALK OR IDENTIFY bit (Bit 13) should be set because
     ***          the TALK ALWAYS bit is set (within the ADDRESS register);
     ***          however, Bit 13 = 0.
```

---

```
5083 *** ERROR -- HP-IB INTERFACE CHIP FAILURE.                    (HDADERR 5083)
     ***
     ***          ADDRESSED TO LISTEN bit (Bit 14) should be set because the
     ***          LISTEN ALWAYS bit is set (within the ADDRESS register);
     ***          however, Bit 14 = 0.
```

---

```
5084 *** ERROR -- INCONSISTENCY IN SYSTEM CONTROLLER STATUS.        (HDADERR 5084)
     ***
     ***          The SYSTEM CNTL bit within the HPIB_STATUS register
     ***          is reset, and the HP-IB Interface Chip is on-line;
     ***          however, Bit 12 = 1.
```

---

```
5085 *** ERROR -- INCONSISTENCY IN SYSTEM CONTROLLER STATUS.        (HDADERR 5085)
     ***
     ***         The SYSTEM CNTL bit within the HPIB_STATUS register
     ***         is set; however, Bit 12 = 0.
```

---

```
5090 *** ERROR -- INCONSISTENCY IN INTERRUPT PENDING STATUS.        (HDADERR 5090)
     ***
     ***           INTERRUPTING_CONDITIONS register: bit  0 = 1
     ***           HPIB_STATUS register            : bit 28 = 0
```

---

```
5091 *** ERROR -- INCONSISTENCY IN INTERRUPT PENDING STATUS.        (HDADERR 5091)
     ***
     ***           INTERRUPTING_CONDITIONS register: bit  0 = 0
     ***           HPIB_STATUS register            : bit 28 = 1
```

---

```
5100 *** ERROR -- THE NUMBER OF DATA BYTES RETURNED BY THE DEVICE EXCEEDED
     ***         THE SPECIFIED BYTE COUNT.                          (HDADERR 5100)
     ***
     ***         Byte count was set to: !
     ***         Data  bytes  returned: !
```

---

## Warning Messages

The following provides a listing, in numerical order, of the most significant warning messages displayed by the system.

**6001**

```
*** WARNING -- Invalid response.  Please answer the question with one
***           of the choices given.
(HDADWARN 6001)
More Help (1..6, 10, 11, <CR>) :
```

---

**6010**

```
*** WARNING -- Soft Physical Address Capability = 0x!,
***           I expected  0x80.
(HDADWARN 6010)
```

---

**6011**

```
*** WARNING -- Type of Module = ! (UNKNOWN PRODUCT),
***           I expected a 4 (TP_A_DMA).
(HDADWARN 6011)
```

---

**6015**

```
*** WARNING -- The device adapter under test is NOT the System
***           Controller, therefore the "On-Line Test" cannot be
***           executed.
(HDADWARN 6015)
```

---

```
6016    *** WARNING -- The device adapter under test is NOT the System
        ***            Controller, therefore the "IFC Interrupt Test" cannot

        ***            be executed.
        (HDADWARN 6016)
```

---

```
6020    *** WARNING -- The IO_STATUS register has one or more undefined bits
        ***            reading in as a 1 (I expected 0's).
        (HDADWARN 6020)
```

---

```
6022    *** WARNING -- A SOFT ERROR (length conflict) occurred since the last
        ***            write to the IO_COMMAND register.

        ***            (IO_STATUS Bit 22 = 1)
        (HDADWARN 6022)
```

---

```
6023    *** WARNING -- A FATAL ERROR occurred since the last write to the
        ***            IO_COMMAND register.

        ***            (IO_STATUS Bit 24 = 1)
        (HDADWARN 6023)
```

---

```
6025    *** WARNING -- PB Interface Chip HAS generated a PB bus error since
        ***            the last reset command.

        ***            (DIAGNOSTIC_STATUS Bit 1 = 0)
        (HDADWARN 6025)
```

---

6030 *** WARNING -- The INITIALIZE OUTBOUND FIFO bit value was read in as
*** being 1, but should always be read as 0.
(HDADWARN 6030)

---

6050 *** WARNING -- No device in handshake mode is connected to the HP-IB,
*** depicted by the NDAC and NRFD lines both being at the
*** logic one level.
(HDADWARN 6050)

---

6060 *** WARNING -- This section is intended for personnel which have a good
*** understanding of the HP-IB Device Adapter's architecture.
*** It is strongly recommended that you print off "More Help"
on
*** Section 10, or reference the user's manual before using
this
*** section.
(HDADWARN 6060)

---

6061 *** WARNING -- Invalid or missing register number ... the register
number
*** value is not a properly formatted DECIMAL integer.
*** Please try again.
(HDADWARN 6061)

---

6062 *** WARNING -- Invalid register number ... the register number entered
*** is beyond the device adapter's address space (highest
*** register number is decimal 1023).
*** Please try again.
(HDADWARN 6062)

---

| | |
|---|---|
| 6064 | *** WARNING -- Invalid or missing datum ... the datum value is not a properly<br>*** formatted HEXADECIMAL integer.<br>*** Please try again.<br>(HDADWARN 6064) |
| 6065 | *** WARNING -- Invalid command ... the command entered cannot be decoded.<br>*** Please try again.<br>(HDADWARN 6065) |
| 6070 | *** WARNING -- It is strongly recommended that you print off "More Help" on<br>*** Section 11, or reference the user's manual before using thie<br>*** eection.<br>(HDADWARN 6070) |
| 6072 | *** WARNING -- The device adapter under test ie NOT the System<br>*** Controller, therefore this section cannot be<br>*** executed.<br>(HDADWARN 6072) |
| 6075 | *** WARNING -- Information entered is not a properly formatted<br>*** HEXADECIMAL integer.<br>*** Please try again. For example: "2BE<CR>".<br>(HDADWARN 6075) |

6076 *** WARNING -- Information entered exceeds the 10-bit limit.

*** Please try again. For example: "3FF<CR>".

(HDADWARN 6076)

---

6077 *** WARNING -- The number of data units entered has exceeded the buffer size.

*** Please terminate the current data entry

*** when prompted.

(HDADWARN 6077)

---

## Introduction (HPIBEISA)

The **HPIBEISA** diagnostic is leveraged from the **HPIBDAD** diagnostic. The more significant differences from a user standpoint are discussed briefly in this section.

## Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database, referencing product number 30600-10053.

## Running HPIBEISA

To run **HPIBEISA**, enter the following at the **DUI** prompt:

```
DUI>run hpibeisa pdev=4.1.0 <run command options>
```

You may specify the test section you wish to run; if you do not specify a test section in the run command, the default sections (Sections 3 and 6) will be run.

## Diagnostic Sections

The following table lists the available HPIBEISA diagnostic sections:

| Section | Description | Mode |
|---|---|---|
| **Section 1** | **More Help** | *Not implemented; supported by "DUI"* |
| **Section 2** | **Reset** | **Exclusive** |
| **Section 3** | **Identify** | **Non-Exclusive** |
| **Section 4** | **Local Loopback**<br>Walking one pattern<br>Test on I/O configuration<br>Registers and MEDUSA registers | **Exclusive/Destructive** |
| **Section 5** | **Hardware Test**<br>Step 1: Register Test<br>Step 2: MEDUSA Functional Test<br>Step 3: Miscellaneous Test<br>Step 4: DMA Test | **Exclusive/Destructive** |
| **Section 6** | **Status**<br>Step 1: Display/Check MEDUSA Instr. Condition & Mask Reg.<br>Step 2: Display/Check MEDUSA Status Reg.<br>Step 3: Display/Check MEDUSA Control Reg.<br>Step 4: Display/Check MEDUSA Address Reg.<br>Step 5: Display/Check MEDUSA PP Mask and Sense Reg.<br>Step 6: Display/Check I/O Config. Regs.<br>Step 7: Display/Check HPIB Bus Status Reg. | **Exclusive** |
| **Section 10** | **Register I/O** | **Exclusive/Destructive** |
| **Section 11** | **Data Transaction** | **Exclusive/Destructive** |

**Note**

Section 11 data transactions are restricted to 8-byte DMA I/O transactions.

# Contents

# 5

# LAN Diagnostic

## Introduction

The LAN Diagnostic (Local Area Network Device Adapter Diagnostic, LANDAD) tests the local area network interface controller (LANIC), part number 27125-60201; the HP-PB LANIC, part number 28652-60001; or the VSC (Viper System Connect) LANIC, part number A1094-66530. The diagnostic will run on any HP Precision Architecture RISC computer system which supports the Online Diagnostic subsystem. LANDAD is capable of detecting a failure in one or more field replaceable units (FRUs). For LANDAD, an FRU is considered to be the LAN interface controller (LANIC) card (which may contain non-LAN circuitry as well), the LANIC connector cable, the attachment unit interface (AUI) cable, the transceiver, and the coaxial tap or BNC tee. LANDAD will accomplish the following:

- Identify the product type and node address of the LANIC
- Report the status of the LANIC
- Report the link statistics of the LANIC
- Reset the LANIC
- Perform selftest on the LANIC
- Execute a local or external loopback
- Send TEST or XID (exchange identification) packets to a remote node and interpret the results
- Perform AUI cable and transceiver fault tests

## Defects and Enhancements

Submit defect reports and enhancement requests for this diagnostic through the STARS database referencing Product Number 30600-10013.

## Minimum Configuration

The minimum required hardware and software consists of the following:

- A functional HP Precision Architecture RISC computer system
- LANIC (part number 27125-60201), HP-PB LAN (part number 28652-60001), or VSC LANIC (part number A1094-66530)
- Connector/stub cable (part number 27125-63009), which is not applicable for HP-PB LAN
- HP 92254A 6-Meter AUI Cable (only used with the HP 30241A transceiver)
- HP 30241A Transceiver ("known good transceiver") for use on ThickLAN cable
- HP 92257B Transceiver Test Fixture for use with the HP 30241A Transceiver,
- HP 24861A Thintransceiver ("known good transceiver") for use on ThinLAN cable
- HP 92227Q Thintransceiver Test Fixture for use with the HP 24861A Thintransceiver
- HP 28664A Transceiver for use on Starlan 10
- HP 5061-4977 Starlan 10 Transceiver Test Fixture for use with HP 28664A Starlan 10 Transceiver
- HP 28685A Ethertwist Transceiver
- HP 5061-4977 Test Fixture for use with the HP 28685A Transceiver
- HP 28683A Fiber Optic Transceiver
- TBD Test Fixture for HP 28683A Transceiver
- MPE XL or HP-UX operating system
- Online diagnostics subsystem (includes LANDAD)

## Operating Instructions

Before attempting to run the diagnostic, ensure that all LAN components are installed, connected, and that all LAN links have been configured.

**Note**

As of MPE XL 1.2, `LANDAD` can no longer verify the LAN card hardware, before the software is installed and configured. Specifically, the network interface must be configured, and the `NETCONTROL START;NET=LAN` command must be issued, before `LANDAD` can verify the hardware.

As of MPE XL 2.0, this will no longer be a concern.

### Default Tests

If you do not specify sections and steps to be run, the following default sections and steps will be executed:

| | |
|---|---|
| Section 3 | Identify |
| Section 4 | Local Loopback |
| Section 6 | Status |

## RUN Command

To bring up the Online Diagnostic subsystem, enter the following command to the system prompt:

```
sysdiag
```

The system responds with the following prompt indicating that access has been gained to the Online Diagnostic User Interface (DUI).

```
DUI>
```

Typing HELP causes a summary of the DUI and its commands to be printed. Refer to the DUI chapter of this manual for details.

To run the diagnostic, enter:

```
DUI > RUN LANDAD pdev=4.4 <RUN Command Options>
                       |            |
                       |      none required for
                       |      default test suite
                       |
               insert physical location of
               device adapter to be tested here,
               or for MPE XL, type the ldev number
               or for HP-UX, type the ldev name
```

## Test Execution

If the LANIC is already in use by the diagnostic system, the system will not grant access to it and the following message will be returned:

```
*** ERROR -- LANIC ALREADY IN USE BY DIAGNOSTIC SYSTEM (LANDADERR 5000)
*** Someone is already diagnosing the LANIC that you requested.
*** It is illegal to have two copies of LANDAD diagnosing the same LANIC
*** at the same time.
```

The diagnostic terminates after outputting this error message. Control will return to the Online Diagnostic subsystem upon completion of the requested/default sections and steps.

If the user specifies a destructive section among those requested, the security level of the user is checked to verify that it is adequate to perform the test (Level 0 or Level 1). If the security level is not adequate, the following error message is printed and the user is returned to the DUI prompt.

```
*** INSUFFICIENT SECURITY LEVEL.  (LANDADERR 5043)
```

If the user specified destructive sections, the following message is printed and the user is prompted for a yes or no answer.

```
A destructive section has been selected.
Do you wish to continue (Y/N) [ N ]?
```

The user must type "y" or "yes" to continue. To terminate the diagnostic, the user must type "n" or "no" or a <cr>. Any other user response will cause the following message to be echoed to the screen:

```
Please answer yes or no
```

The user will then be reprompted.

On MPE XL systems, when the diagnostic is invoked, it checks if the user is at a terminal that is connected to the computer via a Distributed Terminal Controller (DTC) port that uses the LANIC that is to be diagnosed. If the terminal is not connected via such a port, the diagnostic proceeds normally. If the user is at a port of this type, the diagnostic checks to see if any destructive sections are specified. If not, the diagnostic proceeds normally. If there are, the following message is displayed:

```
*** WARNING -- DESTRUCTIVE SECTIONS CAN NOT BE RUN FROM YOUR TERMINAL.
*** DESTRUCTIVE SECTIONS MAY ONLY BE RUN FROM A TERMINAL THAT IS NOT
*** CONNECTED THROUGH THE LANIC TO BE DIAGNOSED.  THE FOLLOWING
*** SECTIONS CAN BE PERFORMED FROM YOUR TERMINAL: 1,3,4,6,7,9,10.
*** NO OTHER SECTIONS CAN BE SPECIFIED. (LANDADWARN 6000)
```

If at any time the number of errors generated reaches the limit specified by you in the ERRCOUNT parameter, the following message will be output:

```
*** THE MAXIMUM NUMBER OF ERROR MESSAGES HAS BEEN EXCEEDED.  (LANDADERR 99)
```

The diagnostic will then terminate. If the ERRPAUSE parameter of the `RUN` command was assigned to "on", the diagnostic will stop after each error is generated and ask if the test should be continued:

```
Do you wish to continue (Y/N) [Y]?
```

If the response is "Y", the test will resume (if possible); if the response is "N", the diagnostic will terminate. If the sections and steps specified by you were executed the number of times specified in the LOOP parameter of the `RUN` command without the number of errors exceeding the ERRCOUNT value, the diagnostic will terminate normally.

At any time that the diagnostic is prompting for information, you may enter "exit" to terminate the diagnostic. Either the entire word or only the first letter of the word 'exit' need be entered, in either upper or lower case. If you exit in this manner, the following message is displayed:

```
Exiting LANDAD per user request...
```

Any time that the diagnostic is not prompting for information, you may enter an interrupt character (CTRL Y for MPE XL and usually CTRL C for HP-UX). When the diagnostic detects the interrupt, one of two actions will occur. If the diagnostic is at a point where it can suspend, it will print the following message and return control to the Online Diagnostic subsystem.

```
LANDAD suspended per user request...
```

You may then either resume or abort the diagnostic. If the diagnostic cannot suspend at this point, the following message will be printed:

```
Unable to suspend in current state, Aborting LANDAD...
```

At this point, LANDAD will be aborted and control will be returned to the Online Diagnostic subsystem.

**Caution**

On MPE XL, you should never abort LANDAD when sections 3, 4, 9, or 10 are specified. This can cause the diagnostic to lose functionality the next time the diagnostic is run.

Upon termination of the diagnostic, control will return to the Online Diagnostic subsystem.

## Test Execution

The following example illustrates running sections 1 through 12 of the diagnostic on an HP-UX system, using the CIO LANIC number 27125.

All user input appears to the right of any system, subsystem, or diagnostic program prompt. Section 1—More Help pauses after printing the first paragraph. If you need information about other sections, enter the appropriate section number. If you want to continue to section 2, merely press (Return) (see the example below).

```
%sysdiag

   *****************************************************************
   ******                                                    ******
   ******            ONLINE DIAGNOSTIC SUBSYSTEM             ******
   ******                                                    ******
   ******          (C) Hewlett Packard Co. 1987              ******
   ******               All Rights Reserved.                 ******
   ******                 DUI version 1.0                    ******
   ******                                                    ******
   *****************************************************************

DUI 1>run landad pdev=8.4 sect=1/12

   *****************************************************************
   ******                                                    ******
   ******     LANDAD LAN Device Adapter Diagnostic           ******
   ******                                                    ******
   ******          (C) Hewlett Packard Co. 1986,1987,1988    ******
   ******              All Rights Reserved.                  ******
   ******                Version A.01.00                     ******
   ******                                                    ******
   *****************************************************************

   Welcome, Today is Wed Nov 04 11:01:19 1987

   A destructive section has been selected.
   Do you wish to continue (Y/N) [ N ]? y

   Section 1 -- More Help

      This Section is no longer supported.  For additional information
      on a given section, use the sections option of the help command
      from the DUI prompt.  For example, if you want more information
      about section 9, type the following:

      DUI> help landad sections 9

   End of Section 1 -- More Help

   Section 2 -- Reset

   End of Section 2 -- Reset
```

```
Section 3 -- Identify

   ID byte  = $46 (CIO LANIC).
   Hardware revcode  = 2.
   CIO firmware datecode = 2716.
   CIO firmware ID   = 1.
   NOVRAM (permanent) station address     = $08-00-09-00-AE-5B.
   RAM (currently active) station address = $08-00-09-00-AE-5B.

End of Section 3 -- Identify

Section 4 -- Local Loopback

    Logging SSAP with driver...
    Sending data to LAN Interface Controller...
    Receiving data from LAN Interface Controller...
    A frame has been successfully transmitted onto the network media.
    Path to LAN Interface Controller is functional.

End of Section 4 -- Local Loopback

Section 5 -- Selftest

    Selftest Completed Successfully.
    The LAN Interface Controller is functional.

End of Section 5 -- Selftest

Section 6 -- Status

    LANIC status has been read successfully.
    LANIC passed selftest.
    LANIC is online.
    transceiver power fuse is OK.
    Free transmit buffers =     4; Maximum =     4.
    Full receive buffers  =     0; Maximum =    16.
    Read data ARQ frame threshold =     1.
    Read data ARQ timeout limit   =     1.
```

```
                          RAM value          NOVRAM value
==========================(Current)==========(Default)=========
Station address           $08-00-09-00-AE-5B $08-00-09-00-AE-5B
Receive bad frames        Disabled           Disabled
Receive multicast frames  Enabled            Disabled
Receive broadcast frames  Enabled            Disabled
Receive all frames        Disabled           Disabled
===============================================================
The following multicast addresses are recognized:
$09-00-09-00-00-01

End of Section 6 -- Status

Section 7 -- Link Statistics

   Step 71 - Read and Display Link Statistics
      Link level statistics have been read successfully.

      Transmit Statistics=====================================
      TOTAL frames transmitted without error.................0
        Deferred transmits...................................0
        One collision transmits..............................0
        More than one collision transmits....................0
      TOTAL frames NOT transmitted...........................0
        Retry errors.........................................0
        Late collisions......................................0
        Loss of carrier during transmit......................0
      No heartbeat detected after transmission...............0
      No free transmit buffers...............................0
      TDR of last retry error................................0
      Infinite deferral errors...............................0
    Receive Statistics========================================
      TOTAL frames received without error....................0
      Frames rejected by address filter......................0
      Frames rejected due to CRC errors......................0
      Frames rejected due to alignment errors................0
      Frames rejected due to oversize length.................0
      LAN controller indicated one or more frames lost.......0
      No free receive buffers................................0
   End of Step 71 - Read and Display Link Statistics

End of Section 7 -- Link Statistics

Section 8 -- External Loopback

   A link frame has been successfully transmitted
   and received from the network cable.

End of Section 8 -- External Loopback
```

```
Section 9 -- Remote Node Test

   This section sends a TEST frame and waits for a response from
   a specified remote node for a specified number of iterations.

   The following success/failure indicators are used:
   "." = The test frame bounced successfully.
   "#" = The test frame was not received before the timeout period.

   Remote Node Address (Six HEX bytes) => 08000900ae5b
   Number of test frames to send ("0" for infinite) [10] =>
   Length of test frames in bytes (60..1514) [500] =>

   Press the interrupt character (usually <Control-C>)
   to prematurely stop the test.

   ..........

   10 out of 10 TEST frames echoed successfully (100%).

End of Section 9 -- Remote Node Test

Section 10 -- Remote XID Test

   This section sends an IEEE 802.2 XID frame to a user specified
   remote node and waits for an IEEE 802.2 XID response frame from
   that remote node.

   Remote Node Address (Six HEX bytes) => 08000900ae5b
   Remote DSAP Address (one even hex byte between $00 and $FE) [$00] =>
   Sending XID command frame...
   Received XID response frame...

   Remote DSAP $00 has class I service.

End of Section 10 -- Remote XID Test
```

```
Section 11 -- AUI Cable Fault Isolation

   This section sends a number of external loopback frames out to a
   transceiver connected to a terminated loopback fixture
   to form a loopback hood.  The status from each frame is displayed on the
   screen.  By moving the transceiver and loopback fixture to various
   AUI cable connector junctions, faults in the AUI cable can be located.

   The following activity indicators are used:
   "P" = External loopback was successful.
   "L" = Loss of carrier detected.
         This usually indicates that there is a loose AUI connector or a broken
         AUI cable between the LANIC and the loopback connector.  In
         a Starlan 10 (Twisted Pair Transceiver) or Fiber-Optic installation,
         it will occur if loopback is attempted without a loopback fixture
         attached while the loopback switch on the transceiver is in the "Test"
         position.  In an EtherTwist installation, it may indicate that the
         setting of the "linkbeat" switch on the local transceiver is
         incompatible with the remote transceiver.
   "R" = Retry fault detected.  (Check loopback transceiver and
         loopback fixture.)
         In a Thicklan or Thinlan installation, this usually indicates that
         there is a loose or broken terminator on the loopback hood, or a
         transceiver or transceiver tap/tee problem.  In a Starlan 10 (Twisted
         Pair Transceiver) or Fiber-Optic installation, it will occur if
         loopback is attempted with a loopback fixture attached while the
         loopback switch on the transceiver is in  the "Normal" position.
         In an EtherTwist installation, it will occur if loopback is attempted
         with a loopback fixture attached.
   "I" = Infinite deferral detected.  (Check terminators.)
         In a Thicklan installation, this could occur if the transceiver is not
         connected to the transceiver tap.  In either a Thicklan or a Thinlan
         installation, it could occur if the network cable is not terminated at
         both ends.  This could also indicate that the LANIC is
         defective.

   A new indicator is posted after each frame is sent.

   Press the interrupt character (usually <Control-C>)
   to stop the test.

   PPPPPInterrupt

P

    Type <return> after the AUI cable has been replaced:

End of Section 11 -- AUI Cable Fault Isolation
```

```
Section 12 -- Offline transceiver Test

    This section requires that the transceiver be removed from
    the LAN cable and connected to a terminated loopback fixture, or if
    your transceiver has a loopback switch, it must be set to 'normal'.

    Please input type of transceiver that this LANIC is connected to:
    Choices are:
    1) HP30241A (Thicklan Transceiver)
    2) HP28641A (Thinlan Transceiver)
    3) HP28664A (Starlan 10 or Twisted Pair Transceiver)
    4) HP28685A (EtherTwist Transceiver)
    5) HP28683A (Fiber-Optic Transceiver)
    6) OTHER

   Step 121 - Two Terminator Test

     Please connect the HP30241A transceiver to a HP92257B terminated
     loopback fixture.  Be certain that both 50-ohm terminators are
     firmly attached.


        The following activity indicators are used:
        "P" = External loopback was successful.
        "R" = Retry fault detected.  (Expected with one missing terminator.)
        "L" = Loss of carrier detected.  (Check AUI cable connections.)
        "I" = Infinite deferral detected.  (Check terminators.)

     A healthy transceiver should show 8 "P"'s...
     PPPPPPPP
   End of Step 121 - Two Terminator Test

   Step 122 - One Terminator Test

      Please connect the HP30241A transceiver to a HP92257B terminated
      loopback fixture.  Remove one 50-ohm terminator from the
      loopback fixture.
      Type <return> after terminator has been removed:

        The following activity indicators are used:
        "P" = External loopback was successful.
        "R" = Retry fault detected.  (Expected with one missing terminator.)
        "L" = Loss of carrier detected.  (Check AUI cable connections.)
        "I" = Infinite deferral detected.  (Check terminators.)

      A healthy transceiver should show 8 "R"'s...

      RRRRRRRR
      Type <return> after the terminator has been replaced:

   End of Step 122 - One Terminator Test
```

```
   End of Section 12 -- Offline Transceiver Test

   landad terminated (pid 7495).  Exit status = 0.
DUI 2>exit #
```

## Test Section Descriptions

There are twelve diagnostic program sections available with LANDAD. You may also select individual steps to be run for sections 7 and 12. LANDAD's sections and steps are summarized below.

| | |
|---|---|
| Section 1 | More Help |
| Section 2 | Reset |
| Section 3 | Identify |
| Section 4 | Local Loopback (to LANIC and back) |
| Section 5 | Selftest |
| Section 6 | Status |
| Section 7 | Link Statistics |
| Step 71 | Read and decode link statistics |
| Step 72 | Reset link statistics |
| Section 8 | External Loopback |
| Section 9 | Remote Node Test |
| Section 10 | Remote XID Test |
| Section 11 | AUI Cable Fault Isolation Test |
| Section 12 | Offline Transceiver Test |
| Step 121 | Two Terminator Test |
| Step 122 | One Terminator Test |

## Section 1—MORE HELP

Since the online diagnostics subsystem now has section help, this section is no longer needed. It is now unsupported, and if users specify this section, the following message will be displayed:

```
This Section is no longer supported.  For additional information
on a given section, use the section option of the help command
from the DUI prompt.  For example, if you want more information
about section 9, type the following:

DUI> help landad sections 9
```

## Section 2—RESET

Reset causes a reset of the LANIC to its power on state. All pertinent data needed by the LANIC to operate properly will then be downloaded to the LANIC. If, after a reset of a LANIC that is offline due to bad hardware, the LANIC indicates that it passed its selftest, the system will put the LANIC into the online state.

**Note** It is better to do a SELFTEST command (Section 5) to bring the LANIC back online because it checks status of the LANIC and displays what has failed if the LANIC is really broken.

If Section 2 completes successfully, it will put the LANIC into the online state, even if it is in the offline state when Section 2 is called.

If no errors are generated, the diagnostic will output the following message:

```
Section 2 -- Reset

End of Section 2 -- Reset
```

## Section 3—IDENTIFY

Identify tells the system to issue a Status command to the LANIC. This command then decodes the information obtained and displays it in a manner that is informative to the user. This section can be used to determine what the LANIC hardware and firmware datecodes are, if applicable (only CIO and NIO have firmware and hardware datecodes). It is also useful in that if it executes successfully, the path from the diagnostic to the LANIC is at least partially functional.

**Output:**

If you are diagnosing a CIO LANIC:

```
Section 3 -- Identify

ID byte               = $06 (CIO LANIC).
Hardware revcode      = 2.
CIO firmware datecode = 2716.
CIO firmware ID       = 1.
NOVRAM (permanent) station address     = $08-00-09-00-AE-5B.
RAM (currently active) station address = $08-00-09-00-AE-5B.

End of Section 3 -- Identify
```

**Note**

If you are running this diagnostic on an MPE XL system, you will receive a driver revcode value in addition to the information displayed above.

If you are diagnosing an HP-PB LANIC:

```
Section 3 -- Identify

ID byte               = $06 (HP-PB LANIC).
HP-PB firmware part number 1 = 28652-81002.
HP-PB firmware part number 2 = 28652-81003.
Hardware revcode      = 2.
NOVRAM (permanent) station address     = $08-00-09-00-AE-5B.
RAM (currently active) station address = $08-00-09-00-AE-5B.

End of Section 3 -- Identify
```

**Note**

If you are running this diagnostic on an MPE XL system, you will receive a driver revcode value in addition to the information displayed above.

If the "NO SQE" jumper is configured, the following message will be displayed:

```
"NO SQE" (ETHERNET 1.0) jumper is configured.
```

If the card ID byte is not $06, $46, or $47, the following message is displayed:

```
*** ID byte              = ! (UNKNOWN PRODUCT);
*** Expecting $06 (CIO LANIC) or $46 (HP-PB LANIC) or $47 (VSC LANIC).
```

If you are running on an VSC LANIC:

```
Section 3 -- Identify

 ID byte              = $47 (VSC LANIC).
 NOVRAM (permanent) station address     = $08-00-09-FF-FF-FF
 RAM (currently active) station address = $08-00-09-FE-FE-FE
 Driver revcode       = $1

 End of Section 3 -- Identify
```

If the LANIC has previously issued a Protocol ERror (PER) or a Dead Or Dying (DOD) AES status message, the following message will be printed:

```
Last failure code = $n
```

The datecodes are displayed in decimal; the station addresses are displayed as 6 hex bytes.

**Note**

Although Identify does not send an external loopback frame onto the network cable, it does report the status of the last external loopback that was performed. It is possible that an error could be reported for an error condition that has already been fixed. If this is the case, an external loopback (Section 8) should be performed. This sends out an external loopback frame and will clear the error (assuming that the problem has been fixed).

If an application has previously logged SSAP $F4, LANDAD will give the user three chances to enter a new SSAP to log. If SSAP $F4 is currently in use, the following message is displayed:

```
*** WARNING -- SSAP n ALREADY IN USE BY ANOTHER PROCESS
    (LANDADWARN 6006)

Please enter a different SSAP to log to (one hex byte) =>
```

If you enter the same SSAP, the following error message is displayed and the question is asked again:

```
** New SSAP must be different from old SSAP.
```

If you enter a non-valid SSAP address, the following error message is displayed and the question is asked again:

```
** n is not a valid SSAP.  A valid SSAP is an even number
** between $02 and $FE.  The SSAP must be entered in hex.
** The leading '$' is optional.
```

If you enter a (Return), the following error message is displayed and the question is asked again:

```
** There is no default for this question.
```

If you try three times and none of these is successful, the following message is displayed and the rest of this section is not executed:

```
*** WARNING -- DIAGNOSTIC SSAP IN USE BY ANOTHER PROCESS (LANDADWARN 6004)
*** Another process has bound to the diagnostic SSAP.
*** This section can only be run after the other process finishes.
```

---

**Note**

Once a good SSAP has been found to log, the diagnostic will use that same SSAP when logging within the same instance of the diagnostic. It is possible that between the time that the diagnostic has terminated and when it is run again, another process will come along and log to that SSAP. If that is the case, the diagnostic will ask the above question again.

---

## Section 4—LOCAL LOOPBACK

Local Loopback opens up the system for normal use and transmits a frame addressed to itself. The LANIC will loop this frame back on the card and send it back to the diagnostic. This test will test the data path from the diagnostic to the card and back. The data in the frame is known and a byte-for-byte comparison of the data is made to be certain that the data was not corrupted. Since the LANIC only loops back the frame if the transmission onto the network medium is successful, this test also checks all components from the network medium to the driver. If the transmission is not successful, LANDAD prints out what it thinks the problem was with the transmission. If no errors are generated, the diagnostic will output the following message:

```
Section 4 -- Local Loopback

   Binding to DAM...(MPE XL) or Logging SSAP with driver...(HP-UX)
   Sending data to LAN Interface Controller...
   Receiving data from LAN Interface Controller...
   A frame was successfully transmitted onto the network media.
   Path to LAN Interface Controller is functional.
   Unbinding from DAM...(MPE XL)

End of Section 4 -- Local Loopback
```

**Note**

Since the packet is *not* sent onto the network when using a VSC LANIC, only the path to the driver is tested. Instead of the display above, you will see the following:

```
                          ************
                          *** NOTE ***
                          ************
Since you are running this section on a VSC LANIC, this section
will only test the path through the driver.  To test the path to
the hardware, run  EXTERNAL LOOPBACK (section 8) or any
destructive test (sections 2,5,8,11,12).
Sending data to Driver...
Receiving data from Driver...
A frame has been successfully transmitted through the Driver.
Path to Driver is functional.
```

If an application has previously logged SSAP $F4, LANDAD will give the user three chances to enter a new SSAP to log. If SSAP $F4 is currently in use, the following message is displayed:

```
*** WARNING -- SSAP n ALREADY IN USE BY ANOTHER PROCESS
    (LANDADWARN 6006)

    Please enter a different SSAP to log to (one hex byte) =>
```

If you enter the same SSAP, the following error message is displayed and the question is asked again:

```
** New SSAP must be different from old SSAP.
```

If you enter a non-valid SSAP address, the following error message is displayed and the question is asked again:

```
** n is not a valid SSAP.  A valid SSAP is an even number
** between $02 and $FE.  The SSAP must be entered in hex.
** The leading '$' is optional.
```

If you enter a (Return), the following error message is displayed and the question is asked again:

```
** There is no default for this question.
```

If you try three times and none of these is successful, the following message is displayed and the rest of this section is not executed:

```
*** WARNING -- DIAGNOSTIC SSAP IN USE BY ANOTHER PROCESS (LANDADWARN 6004)
*** Another process has bound to the diagnostic SSAP.
*** This section can only be run after the other process finishes.
```

**Note**

Once a good SSAP has been found to log, the diagnostic will use that same SSAP when logging within the same instance of the diagnostic. It is possible that between the time that the diagnostic has terminated and when it is run again, another process will come along and log to that SSAP. If that is the case, the diagnostic will ask the above question again.

## Section 5—SELFTEST

Selftest tells the LANIC to perform a hardware selftest. If the returned selftest status is abnormal, messages indicating the problem are displayed. Since selftest brings the card offline and aborts all current information transfers, it should only be done when necessary. If selftest passes, it will put the LANIC into the online state. If message LANDADWARN 6005 is ever displayed, Section 6 (Status) should be executed to see why the card went offline and then selftest should be performed to determine if the problem is still present.

**Note**

If this section completes successfully, it will put the LANIC into the online state, even if it is in the offline state when the section is called.

If no errors are generated, the diagnostic will output the following message:

```
Section 5 -- Selftest

   Selftest completed successfully.
   The LAN Interface Controller is functional.

End of Section 5 -- Selftest
```

## Section 6—STATUS

Status is used to obtain information about the current state of the LANIC. If no errors are generated, the diagnostic will output the following message:

If you are diagnosing a CIO LANIC:

```
    Section 6 -- Status

         LANIC status has been read successfully.
         LANIC passed selftest.
         LANIC is online.
         transceiver power fuse is OK.
         Free transmit buffers = n; Maximum = n.
         Full receive buffers  = n; Maximum = nn.
         Read data ARQ frame threshold = n.
         Read data ARQ timeout limit  n.
                             RAM value             NOVRAM value
  ==========================(Current)=============(Default)========
  Station address           $nn-nn-nn-nn-nn-nn$nn-nn-nn-nn-nn-nn
  Receive bad frames        Disabled               Disabled
  Receive multicast frames  Enabled                Enabled
  Receive broadcast frames  Enabled                Enabled
  Receive all frames        Disabled               Disabled
  ================================================================
  The following multicast addresses are recognized:
  $09-00-09-00-00-01
  $09-00-09-00-00-02

     End of Section 6 -- Status
```

If you are diagnosing an HP-PB LANIC:

```
      Section 6 -- Status

      LANIC status has been read successfully.
      LANIC passed selftest.
      LANIC is online.
      transceiver power fuse is OK.
      Free transmit buffers = n; Maximum = n.
      Full receive buffers  = n; Maximum = nn.
      Using (internal/external) transceiver.
                              RAM value           NOVRAM value
      ==========================(Current)===========(Default)=========
      Station address           $nn-nn-nn-nn-nn-nn$nn-nn-nn-nn-nn-nn
      Receive multicast frames  Enabled             Not Applicable
      Receive broadcast frames  Enabled             Not Applicable
      ================================================================
      The following multicast addresses are recognized:
      $09-00-09-00-00-01
```

```
$09-00-09-00-00-02

End of Section 6 -- Status
```

If the LANIC failed selftest, the following message will replace the LANIC **Passed Selftest** message:

```
*** LANIC failed selftest. *** REPLACE THE LANIC ***
```

If the LANIC's power fuse is blown, the following message is displayed:

```
*** Transceiver power fuse is BLOWN. *** REPLACE FUSE ***
```

If the LANIC is offline, the following message replaces the LANIC **is online message**:

```
LANIC is offline.
```

The entire multicast list is printed. If no multicast addresses **are in the multicast list, the** following message is printed:

```
NO multicast addresses were found.
```

If the no SQE jumper is installed, the following message is printed:

```
"NO SQE" (Ethernet 1.0) jumper is configured.
```

The HP-PB LANIC may also display:

```
Remote reset enabled.
```

or

```
***INTERNAL/EXTERNAL TRANSCEIVER JUMPER MISSING OR MISALIGNED.
```

If you are diagnosing a VSC LANIC:

```
Section 6 -- Status

LANIC status has been read successfully.
LANIC passed selftest.
LANIC is online.
AUI power fuse is OK.
Using External Transceiver.
"NO SQE" (Ethernet 1.0) jumper is configured.
                             RAM value           NOVRAM value
=========================(Current)===========(Default)=========
Station address              $08-00-09-FE-FE-FE  $08-00-09-FF-FF-FF
Receive multicast frames     Enabled             Not applicable
Receive broadcast frames     Enabled             Not applicable
==============================================================
The following multicast addresses are recognized:
No multicast addresses were found.

End of Section 6 -- Status
```

If the AUI power fuse is not blown, the following message will be **displayed**:

```
AUI power fuse is OK.
```

If the LANIC's AUI power fuse is blown, the following message will **be displayed**:

```
*** AUI power fuse is BLOWN. *** REPLACE FUSE ***
```

## Explanation of Status Values for CIO LANIC

This is a line by line explanation of the fields that the status command returns. The following is a listing of a typical status display with line numbers added. This listing is used to show which line number is being explained.

```
 1 =>   Section 6 -- Status
 2 =>
 3 =>   LANIC status has been read successfully.
 4 =>   LANIC passed selftest.
 5 =>   LANIC is online.
 6 =>   Transceiver power fuse is OK.
 7 =>   Free transmit buffers =     4; Maximum =     4.
 8 =>   Full receive buffers  =     0, Maximum =    16.
 9 =>   Read data ARQ frame threshold =     1.
10 =>   Read data ARQ timeout limit   =     1.
11 =>                             RAM value             NOVRAM value
12 =>   ==========================(Current)=============(Default)========
13 =>   Station address           $08-00-09-00-14-01   $08-00-09-00-14-01
14 =>   Receive bad frames        Disabled             Disabled
15 =>   Receive multicast frames  Enabled              Enabled
16 =>   Receive broadcast frames  Enabled              Enabled
17 =>   Receive all frames        Disabled             Disabled
18 =>   ==================================================================
19 =>   The following multicast addresses are recognized:
20 =>   $09-00-09-00-00-01
21 =>   $09-00-09-00-00-02
22 =>
23 =>   End of Section 6 -- Status
```

**LANIC selftest status (line 4)** This line indicates whether the LANIC passed or failed selftest. If the LAN passed selftest, the following message will be displayed:

```
LANIC passed selftest.
```

If it failed selftest, the following message will be printed:

```
*** LANIC failed selftest. *** REPLACE THE LANIC ***
```

If this message is displayed, the LANIC interface card should be replaced. Note also that the rest of the information may or may not be valid.

**LANIC online or offline status (line 5)** This indicates if the LANIC is online or offline. Online means that the LANIC is ready to transmit and receive frames from the network media.

**Transceiver power fuse status (line 6)** This line indicates the status of the transceiver power fuse. If the fuse is OK, the following message is displayed:

```
Transceiver power fuse is OK.
```

If the fuse is blown, the following message is printed:

```
*** Transceiver power fuse is BLOWN. *** REPLACE FUSE ***
```

**Free transmit buffers (line 7)** This is the number of buffers that the LANIC has currently available for transmit frames. There is a one for one correspondence between frames and buffers. The maximum will equal 4 for the current CIO LANIC. It is normal for this number to vary between 0 and the maximum during normal network usage.

**Full receive buffers (line 8)** This is the number of buffers that the LANIC has received from the network media that have yet to be read by the driver. There is a one for one correspondence between frames and buffers. The maximum will equal 16 for the current CIO LANIC. It is normal for this number to vary between 0 and the maximum during normal network usage.

**Read data ARQ frame threshold (line 9)** This is the number of frames that must be received before the CIO LANIC will interrupt the host. If this value is non-zero and the read data ARQ timeout limit is zero, the CIO LANIC will interrupt as soon as a frame arrives.

**Read data ARQ timeout limit (line 10)** This is the number of 10 millisecond periods that the CIO LANIC will interrupt the host after receiving a frame from the link but not receiving read ARQ frame threshold frames from the network media. If this value is non-zero and the frame threshold is zero, the CIO LANIC will interrupt as soon as a frame arrives.

Parameters in lines 13 through 17 have two values. The first is the RAM value. This is the value that the LANIC is currently using. The second is the NOVRAM value. This is the default value for that parameter that the LANIC will use when it is initially powered up.

**Station address (line 13)** This is the six byte address that the LANIC will respond to. This address will also be used when sending out frames to the network media.

**Receive bad frames (line 14)** This is either enabled or disabled. When enabled, the CIO LANIC will save bad frames that it receives and pass them up to the driver. When disabled, bad frames will be counted in statistics, but will be discarded.

**Receive multicast frames (line 15)** This is either enabled or disabled. When enabled, the LANIC will receive frames sent to multicast addresses that it has been set up to receive. A list of up to 64 multicast addresses can be downloaded to the LANIC.

**Receive broadcast frames (line 16)** This is either enabled or disabled. When enabled, the LANIC will receive frames sent to the broadcast address.

**Receive all frames (line 17)** This is either enabled or disabled. When enabled, the CIO LANIC will attempt to receive all frames from the network media. When disabled, the LANIC will only receive frames sent to it's address and if receive broadcast frames is enabled, also frames sent to the broadcast address.

**The following multicast addresses are recognized (line 19)** This is a list of all multicast addresses that the LANIC will respond to.

**Note**

On HP-UX, only the first 16 multicast addresses can be displayed.

### Explanation of Status Values for HP-PB LANIC

This is a line by line explanation of the fields that the status command returns. The following is a listing of a typical status display with line numbers added. This listing is used to show which line number is being explained.

```
 1 =>   Section 6 -- Status
 2 =>
 3 =>   LANIC status has been read successfully.
 4 =>   LANIC passed selftest.
 5 =>   LANIC is online.
 6 =>   transceiver power fuse is OK.
 7 =>   Free transmit buffers =     4; Maximum =     4.
 8 =>   Full receive buffers  =     0; Maximum =    32.
 9 =>   Using INTERNAL transceiver.
10 =>                              RAM value            NOVRAM value
11 =>   ==========================(Current)=============(Default)========
12 =>   Station address           $08-00-09-00-14-01   $08-00-09-00-14-01
13 =>   Receive multicast frames  Enabled              Not applicable
14 =>   Receive broadcast frames  Enabled              Not applicable
15 =>   =================================================================
16 =>   The following multicast addresses are recognized:
17 =>   $09-00-09-00-00-01
18 =>   $09-00-09-00-00-02
19 =>
20 =>   End of Section 6 -- Status
```

**LANIC selftest status (line 4)**

This line indicates whether the LANIC passed or failed selftest. If the LANIC passed selftest, the following message will be displayed:

```
LANIC passed selftest.
```

If it failed selftest, the following message will be printed:

```
*** LANIC failed selftest. *** REPLACE THE LANIC ***
```

If this message is displayed, the LANIC interface card should be replaced. Note also that the rest of the information may or may not be valid.

**LANIC online or offline status (line 5)**

This indicates if the LANIC is online or offline. Online means that the LANIC is ready to transmit and receive frames from the network media.

**Transceiver power fuse status (line 6)**

This line indicates the status of the transceiver power fuse. If the fuse is OK, the following message is displayed:

```
Transceiver power fuse is OK.
```

If the fuse is blown, the following message is printed:

```
*** Transceiver power fuse is BLOWN. *** REPLACE FUSE ***
```

**Free transmit buffers (line 7)**
This is the number of buffers that the LANIC has currently available for transmit frames. There is a one for one correspondence between frames and buffers. The maximum will equal 4 for the current HP-PB LANIC. It is normal for this number to vary between 0 and the maximum during normal network usage.

**Full receive buffers (line 8)**
This is the number of buffers that the LANIC has received from the network media that have yet to be read by the driver. There is a one for one correspondence between frames and buffers. The maximum will equal 32 for the current HP-PB LANIC. It is normal for this number to vary between 0 and the maximum during normal network usage.

**Using INTERNAL or EXTERNAL Transceiver (line 9)**
This indicates the state of the internal/external transceiver jumpers. If the jumper is on the INTERNAL transceiver block, the Using INTERNAL transceiver message is printed. If the jumper is on the EXTERNAL block, the Using EXTERNAL transceiver message is displayed. If the jumper is missing or misaligned, the following error message is displayed:

```
*** INTERNAL/EXTERNAL TRANSCEIVER JUMPER MISSING OR MISALIGNED.
```

Parameters in lines 12 through 14 have two values. The first is the RAM value. This is the value that the LANIC is currently using. The second is the NOVRAM value. This is the default value for that parameter that the LANIC will use when it is initially powered up.

**Station address (line 12)**
This is the six byte address that the LANIC will respond to. This address will also be used when sending out frames to the network media.

**Receive multicast frames (line 13)**
This is either enabled or disabled. When enabled, the LANIC will receive frames sent to multicast addresses that it has been set up to receive. A list of up to 64 multicast addresses can be downloaded to the LANIC. Upon power-on or reset, this is disabled.

**Receive broadcast frames (line 14)**
This is either enabled or disabled. When enabled, the LANIC will receive frames sent to the broadcast address. Upon power-on or reset, this is enabled.

**The following multicast addresses are recognized (line 16)**
This is a list of all multicast addresses that the LANIC will respond to.

**Note** On HP-UX, only the first 16 multicast addresses can be displayed.

### Explanation of Status Values for VSC LANIC

This is a line by line explanation of the fields that the status command returns. The following is a listing of a typical status display with line numbers added. This listing is used to show which line number is being explained.

```
 1 =>   Section 6 -- Status
 2 =>
 3 =>   LANIC status has been read successfully.
 4 =>   LANIC passed selftest.
 5 =>   LANIC is online.
 6 =>   AUI power fuse is OK.
 7 =>   Using INTERNAL transceiver.
 8 =>                             RAM value            EEPROM value
 9 =>   ==========================(Current)=============(Default)========
10 =>   Station address           $08-00-09-00-14-01   $08-00-09-00-14-01
11 =>   Receive multicast frames  Enabled              Not applicable
12 =>   Receive broadcast frames  Enabled              Not applicable
13 =>   =================================================================
14 =>   The following multicast addresses are recognized:
15 =>   $09-00-09-00-00-01
16 =>   $09-00-09-00-00-02
17 =>
18 =>   End of Section 6 -- Status
```

**LANIC selftest status (line 4)** This line indicates whether the LANIC passed or failed selftest. If the LANIC passed selftest, the following message will be displayed:

```
LANIC passed selftest.
```

If it failed selftest, the following message will be printed:

```
*** LANIC failed selftest. *** REPLACE THE LANIC ***
```

If this message is displayed, the LANIC interface card should be replaced. Note also that the rest of the information may or may not be valid.

**LANIC online or offline status (line 5)** This indicates if the LANIC is online or offline. Online means that the LANIC is ready to transmit and receive frames from the network media.

**AUI power fuse status (line 6)** This line indicates the status of the AUI power fuse. If the fuse is OK, the following message is displayed:

```
AUI power fuse is OK.
```

If the fuse is blown, the following message is printed:

```
*** AUI power fuse is BLOWN. *** REPLACE FUSE ***
```

**Using INTERNAL or EXTERNAL Transceiver (line 7)** This indicates the state of the internal/external transceiver jumpers. If the jumper is on the INTERNAL transceiver block, the Using INTERNAL Transceiver message is printed. If the jumper is on the EXTERNAL block, the Using EXTERNAL Transceiver message is displayed. If the jumper is missing or misaligned, the following error message is displayed:

```
*** INTERNAL/EXTERNAL TRANSCEIVER JUMPER MISSING OR MISALIGNED.
```

Parameters in lines 10 through 12 have two values. The first is the RAM value. This is the value that the LANIC is currently using. The second is the EEPROM value. This is the default value for that parameter that the LANIC will use when it is initially powered up.

**Station address (line 10)** This is the six byte address that the LANIC will respond to. This address will also be used when sending out frames to the network media.

**Receive multicast frames (line 11)** This is either enabled or disabled. When enabled, the LANIC will receive frames sent to multicast addresses that it has been set up to receive. A list of up to 64 multicast addresses can be downloaded to the LANIC. Upon power-on or reset, this is disabled.

**Receive broadcast frames (line 12)** This is either enabled or disabled. When enabled, the LANIC will receive frames sent to the broadcast address. Upon power-on or reset, this is enabled.

**The following multicast addresses are recognized (line 14)** This is a list of all multicast addresses that the LANIC will respond to.

---

**Note** On HP-UX, only the first 16 multicast addresses can be displayed.

---

## Section 7—LINK STATISTICS

Link Statistics allows you to read and display link statistics that the LANIC keeps. It also allows you to reset these link statistics.

This function has two steps: Step 71 is the default step. It reads link statistics from the LANIC and decodes them. To invoke this step, enter:

```
DUI > run landad ldev=/dev/diag/lan0 section=7
```

The second function is the reset statistics function, Step 72. This function is disruptive since it modifies data on the LANIC. Since it is not a default step, you must specify step=72 in the `RUN` command, as follows:

```
DUI > run landad ldev=/dev/diag/lan0 section=7 [72]
```

### Step 71 - Read and Display Link Statistics

This step requests link level statistics from the LANIC through the system and displays the statistics. If no errors are generated, the diagnostic will output the following message:

```
Section 7 -- Link Statistics

   Step 71 - Read and Display Link Statistics
   Link level statistics have been read successfully.

   Transmit Statistics ================================
     TOTAL frames transmitted without error...........n
        Deferred transmits............................n
        One collision transmits.......................n
        More than one collision transmits.............n
     TOTAL frames NOT transmitted.....................n
        Retry errors..................................n
        Late collision................................n
        Loss of carrier during transmit...............n
     No heartbeat detected after transmission.........n
     No free transmit buffers.........................n
     TDR of last retry error..........................n
     Indefinite deferral errors.......................n
   Receive Statistics=================================
     TOTAL frames received without error..............n
     Frames rejected by address filter................n
     Frames rejected due to CRC errors................n
     Frames rejected due to alignment errors..........n
     Frames rejected due to oversize length...........n
     LAN Controller indicated one or more frames lost..n
     No free receive buffers..........................n
   End of Step 71 - Read and Display Link Statistics

End of Section 7 -- Link Statistics
```

### Step 72 - Reset Link Statistics

This section resets the link statistics on the LANIC. If no errors are generated, the diagnostic will output the following message:

```
Section 7 -- Link Statistics

   Step 72 - Reset Link Statistics
   Link statistics reset successfully.
   Link level statistics have been read successfully

   Transmit Statistics ==================================
     TOTAL frames transmitted without error...........0
        Deferred transmits............................0
        One collision transmits.......................0
        More than one collision transmits.............0
     TOTAL frames NOT transmitted.....................0
        Retry errors..................................0
        Late collision................................0
        Loss of carrier during transmit...............0
     No heartbeat detected after transmission.........0
     No free transmit buffers.........................0
     TDR of last retry error..........................0
     Indefinite deferral errors.......................0
   Receive Statistics===================================
     TOTAL frames received without error..............0
     Frames rejected by address filter................0
     Frames rejected due to CRC errors................0
     Frames rejected due to alignment errors..........0
     Frames rejected due to oversize length...........0
     LAN Controller indicated one or more frames lost..0
     No free receive buffers..........................0
   End of Step 72 - Reset Link Statistics

End of Section 7 -- Link Statistics
```

## Section 8—EXTERNAL LOOPBACK

External Loopback first takes the card offline and then tells the LANIC to perform an external loopback test. This test transmits and receives a frame from the network cable. If this test passes, the following things have a high probability of being functional:

- This network cable segment
- Both 50 ohm terminators
- This node's transceiver tap
- This node's transceiver
- This node's AUI cable(s)
- This node's stub cable
- This node's LANIC

**Note** If this section completes successfully, it will put the LANIC into the online state, even if it is in the offline state when the section is called.

If no errors are generated, the diagnostic will output the following message:

```
Section 8 -- External Loopback

    A link frame has been successfully transmitted
    and received from the network cable.

End of Section 8 -- External Loopback
```

## Section 9—REMOTE NODE TEST

Remote Node tests the ability of this node to bounce a packet off another node connected to the same physical (or logical if there are repeaters and/or bridges in the network) network. This is useful for two reasons: First, it illustrates that the node can communicate with a remote node. Second, it can point to upper level software problems. If a frame can be bounced off another node using the diagnostic, but normal NS communications do not work, the problem is not the hardware, it's the upper level software.

This section sends an IEEE 802.2 test frame. This test frame can be any length from 60 bytes (a minimum 802.3 frame) to 1514 bytes (a maximum length 802.3 frame). The default is 500 bytes. When a test response frame is received from the remote station, its length is checked for being either a minimum size frame or for being the specified length -0/+1. If the response frame is not a minimum size frame, then the data is checked against the data sent. If it is not the same, then the test frame part of the test fails.

This section will allow communication only to individual network addresses. If you input a broadcast or multicast address as a response to the Remote Node Address prompt, an error message will be issued and you will be prompted again for a valid remote node address.

**Note**

Part of what this section tests is the receive threshold of the transceiver at the two nodes involved in the test. The worst case for this is when the two nodes are the maximum 500 meters (185 meters for ThinLAN) apart. Therefore, it is best if you attempt to bounce frames off a distant node.

**Caution**

The remote node MUST be capable of responding to IEEE 802.2 test frames, and that node must be in a state to answer those frames. For example, most systems must have the LANIC device driver installed and operating before test frames will be answered.

If no errors are generated, the diagnostic will output the following message:

```
Section 9 -- Remote Node Test

   This Section sends a TEST frame and waits for a response from
   a specified remote node for a specified number of iterations.

   The following success/failure indicators are used:
   "." = The test frame bounced successfully.
   "#" = The test frame was not received before the timeout period.

   Remote Node Address (Six HEX bytes) => 0800 0900 1401
   Number of test frames to send ("0" for infinite) [10] =>
   Length of test frames in bytes (60..1514) [500] =>
   Press the interrupt character (usually <Control-C>)
   to prematurely stop the test.

   ..........

   10 out of 10 TEST frames echoed successfully (100%).

End of Section 9 -- Remote Node Test
```

If the test is being run on an HP-UX system, the message is:

```
Press the interrupt character (usually <CTRL-C>)
to stop the test.
```

If you input an invalid remote node address, the following message will be displayed and you will be prompted again:

```
Address must be 12 hexadecimal digits with any combination of delimiters
Hexadecimal digits are in the set:
['0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'].
Hexadecimal digits may be either upper or lower case.

Delimiters are in the set: [' ','-','$']
Examples:
  080009123ABC
  08-00-09-12-3A-BC
  $0800-0912-3ABC
  08 00    09 12   3a Bc
  0-$ 800--$-09-1$2--3abc
```

Only the first twelve digits are read including zeros. Since the address must be an individual address, the low bit of the high byte of the address is checked to see if it is set. If it is, the following message is displayed and you are re-prompted for the address:

```
Address must be an individual address i.e. the most significant byte
of the address must be even.  (An odd first byte indicates a group
address.)
Examples:
  Correct:
    08 00 09 00 12 AB

Incorrect:
  FF FF FF FF FF FF
  09 00 09 00 00 01
  01 02 03 04 05 06
```

If an application has previously logged SSAP $F4, LANDAD will give you three chances to enter a new SSAP to log to. If SSAP $F4 is currently in use, the following message is displayed:

```
*** WARNING -- SSAP n ALREADY IN USE BY ANOTHER PROCESS
    (LANDADWARN 6006)

    Please enter a different SSAP to bind to =>
```

If you enter the same SSAP, the following error message is displayed and the question is asked again:

```
** New SSAP must be different from old SSAP.
```

If you enter a non-valid SSAP address, the following error message is displayed and the question is asked again:

```
** n is not a valid SSAP.  A valid SSAP is an even number
** between $02 and $FE.  The SSAP must be entered in hex.
** The leading '$' is optional.
```

If you enter a (Return), the following error message is displayed and the question is asked again.

```
** There is no default for this question.
```

If you try three times and none of these is successful, the following message is displayed and the rest of this section is not executed:

```
*** WARNING -- DIAGNOSTIC SSAP IN USE BY ANOTHER PROCESS (LANDADWARN 6004)
*** Another process has bound to the diagnostic SSAP.
*** This section can only be run after the other process finishes.
```

**Note**

Once a good SSAP has been found to log, the diagnostic will use that same SSAP when logging within the same instance of the diagnostic. It is possible that between the time that the diagnostic has terminated and when it is run again, another process will come along and log to that SSAP. If that is the case, the diagnostic will ask the above question again.

## Section 10—REMOTE XID TEST

Remote XID Test allows you to send IEEE 802.2 XID command frames to a specified remote node and receive the response frame from the remote node. This section also decodes the response and displays what type of service is available at the remote node.

You are prompted for both the 6-byte remote node address and the 1-byte DSAP of the service on the remote system to which the XID frame should be sent. The addresses that you give must be individual addresses, i.e., they cannot be broadcast or multicast. Should you input one of these illegal addresses, an error message will be issued and you will be prompted again for a valid address. If no errors are generated, the diagnostic will output the following message:

```
Section 10 -- Remote XID Test

  This section sends an IEEE 802.2 XID frame to a user specified
  remote node and waits for an IEEE 802.2 XID response frame from
  that remote node.

  Remote Node Address (Six HEX bytes) => 0800 0900 1401
  Remote DSAP Address (one even hex byte between $00 and $FE) [$00] =>
  Sending XID command frame...
  Received XID response frame...

  Remote DSAP n has class I service.

End of Section 10 -- Remote XID Test
```

If the remote DSAP has class II service, the following message is displayed:

```
Remote DSAP n has class II service, window size = n.
```

If the remote DSAP has class III service, the following message is displayed:

```
Remote DSAP n has class III service.
```

If the remote DSAP has class IV service, the following message is displayed:

```
Remote DSAP n has class IV service.
```

If the remote node does not send back a response frame, the following message is displayed:

```
No response received from remote node.
```

If you input an invalid remote node address, the following message will be displayed and you will be prompted again:

```
Address must be 12 hexadecimal digits with any combination of delimiters
Hexadecimal digits are in the set:
['0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'].
Hexadecimal digits may be either upper or lower case.
Delimiters are in the set: [' ','-','$']
Examples:
  080009123ABC
  08-00-09-12-3A-BC
  $0800-0912-3ABC
  08 00    09 12   3a Bc
  0-$ 800--$-09-1$2--3abc
```

Since the address must be an individual address, the low bit of the high byte of the address is checked to see if it is set. If it is, the following message is displayed and you are re-prompted for the address:

```
Address must be an individual address i.e. the most significant byte
of the address must be even.  (An odd first byte indicates a group address.)
Examples:
  Correct:
   08 00 09 00 12 AB

  Incorrect:
    FF FF FF FF FF FF
    09 00 09 00 00 01
    01 02 03 04 05 06
```

If an application has previously logged SSAP $F4, LANDAD will give you three chances to enter a new SSAP to log to. If SSAP $F4 is currently in use, the following message is displayed:

```
*** WARNING -- SSAP n ALREADY IN USE BY ANOTHER PROCESS
    (LANDADWARN 6006)

Please enter a different SSAP to bind to =>
```

If you enter the same SSAP, the following error message is displayed and the question is asked again:

```
** New SSAP must be different from old SSAP.
```

If you enter a non-valid SSAP address, the following error message is displayed and the question is asked again:

```
** n is not a valid SSAP.  A valid SSAP is an even number
**  between $02 and $FE.  The SSAP must be entered in hex.
**  The leading '$' is optional.
```

If you enter a (Return), the following error message is displayed and the question is asked again:

```
** There is no default for this question
```

If you try three times and none of these is successful, the following message is displayed and the rest of this section is not executed:

```
*** WARNING -- DIAGNOSTIC SSAP IN USE BY ANOTHER PROCESS (LANDADWARN 6004)
*** Another process has bound to the diagnostic SSAP.
*** This section can only be run after the other process finishes.
```

**Note**

Once a good SSAP has been found to log to, the diagnostic will use that same SSAP when logging within the same instance of the diagnostic. It is possible that between the time that the diagnostic has terminated and when it is run again, another process will come along and log to that SSAP. If that is the case, the diagnostic will ask the above question again.

## Section 11—AUI CABLE FAULT ISOLATION

AUI Cable Fault Isolation isolates a broken cable in the AUI cable segment. This is done by repeatedly sending external loopback frames and checking to see if the frame loopback was successful. If it was successful, a "P" is printed. If it was not, the reason that it was not is given on the screen. If the reason was loss of carrier error, an "L" is printed. This might indicate a broken AUI cable. If the reason is retry fault, an "R" is printed. This may indicate that the problem is a bad loopback hood.

To run this section, you first connect a terminated loopback fixture to the end of the stub cable. Then the test is started. Next, you disconnect the loopback hood at the stub cable, reconnect the stub cable to the AUI cable, and connect the loopback fixture to the opposite end of the AUI cable. If there are multiple AUI cables, you continue to do this until you get to the AUI cable which connects to the transceiver. After doing all of this, the test is stopped (via [&[CTRL]]-C on HP-UX or [&[CTRL]]-Y on MPE XL) and the pattern of activity indicators is analyzed. If the pattern looks like this:

```
LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
```

The problem is most likely a bad stub cable. If the pattern looks like this:

```
PPPPLLLLPPPPLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL
```

The problem is most likely the second AUI cable, or if there is only a single AUI cable, then the transceiver. (This may not be obvious. The first set of Ps is where the stub cable passed the test. The next set of Ls is where the loopback hood was disconnected to connect it back to the AUI cable. The next set of Ps is where the first AUI cable passed. At this point, the problem is beyond the first AUI cable.)

Note: The last thing that this section does is reset the LANIC and the system. If the reset is successful, the LANIC is put into the online state, even if it is in the offline state when the section is started.

If no errors are generated, the diagnostic will output the following message:

```
Section 11 -- AUI Cable Fault Isolation

    This section sends a number of external loopback frames out to a
    transceiver connected to a terminated loopback fixture
    to form a loopback hood.  The status from each frame is displayed on the
    screen.  By moving the transceiver and loopback fixture to various
    AUI cable connector junctions, faults in the AUI cable can be located.

    The following activity indicators are used:
    "P" = External loopback was successful.
    "L" = Loss of carrier detected.
          This usually indicates that there is a loose AUI connector or a broken
          AUI cable between the LANIC and the loopback connector.  In
          a Starlan 10 (Twisted Pair Transceiver) or Fiber-Optic installation,
          it will occur if loopback is attempted without a loopback fixture
          attached while the loopback switch on the transceiver is in the "Test"
          position.  In an EtherTwist installation, it may indicate that the
          setting of the "linkbeat" switch on the local transceiver is
          incompatible with the remote transceiver.
```

```
"R" = Retry fault detected.  (Check loopback transceiver and
      loopback fixture.)
      In a Thicklan or Thinlan installation, this usually indicates that
      there is a loose or broken terminator on the loopback hood, or a
      transceiver or transceiver tap/tee problem.  In a Starlan 10 (Twisted
      Pair Transceiver) or Fiber-Optic installation, it will occur if
      loopback is attempted with a loopback fixture attached while the
      loopback switch on the transceiver is in  the "Normal" position.
      In an EtherTwist installation, it will occur if loopback is attempted
      with a loopback fixture attached.
"I" = Infinite deferral detected.  (Check terminators.)
      In a Thicklan installation, this could occur if the transceiver is not
      connected to the transceiver tap.  In either a Thicklan or a Thinlan
      installation, it could occur if the network cable is not terminated at
      both ends.  This could also indicate that the LANIC is
      defective.

A new indicator is posted after each frame is sent.

Press <control-Y> to stop the test.

PPPPLLLLPPPPLLLLLLLLLLL CTRL-Y
Type <Return> after the AUI cable has been replaced.
End of Section 11 -- AUI Cable Fault Isolation
```

If the diagnostic is being run on an HP-UX machine, the "<CTRL-Y> to stop" message is replaced by the following message:

```
Press the interrupt character (usually <CTRL-C>) to stop the test.
```

If the device being diagnosed is an HP-PB LANIC or an VSC LANIC, and the internal transceiver is configured, the following message is displayed and the section is not executed:

```
This test is not valid when using the Internal Thintransceiver
on the HP-PB LANIC or the VSC LANIC

No AUI Cable Fault Isolation Test Performed.
```

**Note** The HP-PB LANIC has no stub cable.

## Section 12—OFFLINE TRANSCEIVER TEST

The Offline Transceiver Test provides a way to verify that a transceiver is operating properly. In order to run this test, the transceiver should be taken off the network cable and a terminated loopback hood (HP 92257B or HP 92257Q) should be attached to it.

There are two different steps in this section. The first step requires that

- For the HP30241A (Thicklan Transceiver) and HP28641A (Thinlan Transceiver), the loopback fixture should be connected to the transceiver. Both terminators must be attached to the loopback fixture.
- For the HP28664A (Starlan 10 or Twisted Pair Transceiver) or the HP28683A (Fiber-Optic Transceiver), the loopback fixture should be attached to the transceiver. The loopback switch on the transceiver should be set to the "Test" position.
- For the HP28685A (EtherTwist Transceiver), no loopback fixture should be used. The linkbeat switch on the transceiver should be set to the "disabled" position.

Eight external loopback frames are then sent to the transceiver. These packets should be sent successfully, which will be indicated by "P"'s being displayed.

The second section requires that

- For the HP30241A (Thicklan Transceiver) and HP28641A (Thinlan Transceiver), one of the 50-ohm terminators should be removed from the loopback fixture.
- For the HP28664A (Starlan 10 or Twisted Pair Transceiver) or HP28683A (Fiber-Optic Transceiver), the loopback fixture should be attached to the transceiver. The loopback switch on the transceiver should be set to the "Normal" position.
- For the HP28685A (EtherTwist Transceiver), the loopback fixture should be attached to the transceiver. The linkbeat switch on the transceiver should be set to the position in which it is used in normal operation.

Eight more external loopback frames are then sent to the transceiver. These should fail because of retry errors, indicated by "R"'s being displayed. If the diagnostic displays "P"'s for the first section and "R"'s for the second section, the transceiver passes. If the diagnostic displays "P"'s for both sections, the transceiver's collision detection circuitry is not working and the transceiver should be replaced. If the diagnostic displays "R"'s for both sections, the TLF may have a loose terminator or be broken, or the transceiver may be broken. If the diagnostic displays "L"'s for both sections, there may be a short in the TLF, or an AUI cable may be bad, or the transceiver may be broken.

If you are using the internal Thin transceiver on the HP-PB LANIC or the VSC LANIC, the transceiver type question is not asked.

**Note** The last thing that this section does is to reset the LANIC and the DAM or driver. If the reset is successful, the LANIC is put into the online state, even if it was in the offline state when the section was called.

If no errors are generated, the diagnostic will output the following message:

```
Section 12 -- Offline Transceiver Test

   This section requires that the transceiver be removed from the LAN
   cable and connected to a terminated loopback fixture.

   Please input type of transceiver that this LANIC is connected to:
   Choices are:
    1) HP30241A (Thicklan Transceiver)
    2) HP28641A (Thinlan Transceiver)
    3) HP28664A (Starlan 10 or Twisted Pair Transceiver)
    4) HP28685A (EtherTwist Transceiver)
    5) HP28683A (Fiber-Optic Transceiver)
    6) OTHER
   Please select transceiver type (1-3) => 1

   Step 121 - Two Terminator Test

      Please connect the HP30241A transceiver to a HP92257B terminated
      loopback fixture.  Be certain that both 50-ohm terminators are
      firmly attached.
      Type <return> when loopback hood is connected:

      The following activity indicators are used:
      "P" = External loopback was successful.
      "R" = Retry fault detected.  (Expected with one missing terminator.)
      "L" = Loss of carrier detected.  (Check AUI cable connections.)
      "I" = Infinite deferral detected.  (Check terminators.)

      A healthy transceiver should show 8 "P"s...

      PPPPPPPP

   End of Step 121 - Two Terminator Test
```

```
Step 122 - One Terminator Test

  Please connect the HP30241A transceiver to a HP92257B terminated
  loopback fixture.  Remove one 50-ohm terminator from the loopback
  fixture.
  Type <return> after terminator has been removed:

  The following activity indicators are used:
  "P" = External loopback was successful.
  "R" = Retry fault detected.  (Expected with one missing terminator.)
  "L" = Loss of carrier detected.  (Check AUI cable connections.)
  "I" = Infinite deferral detected.  (Check terminators.)

  A healthy transceiver should show 8 "R"s...

  RRRRRRRR
  Type <Return> after the terminator has been replaced.
End of Step 122 - One Terminator Test

End of Section 12 -- Offline Transceiver Test
```

## Error Messages

The following is a list of error messages which may appear when using LANDAD. The messages are listed in numerical order, and, where applicable, the probable cause and recommended action are part of the message. The wording of some messages is slightly different for HP-UX and MPE XL. Where this is the case, it is pointed out by enclosing the MPE XL version in square brackets. Only those errors which are defined by LANDAD are shown here; however, other error messages may also be displayed. For other errors, consult the DUI section of this manual and the operating system manuals.

| | |
|---|---|
| 99 | `*** THE MAXIMUM NUMBER OF ERROR MESSAGES HAS BEEN EXCEEDED. (LANDADERR 99)` |
| CAUSE | The maximum errors which were indicated in the run string by the user have been exceeded. |
| ACTION | Either increase the maximum errors parameter in the run string or let it default to infinity. |

| | |
|---|---|
| 407 | `*** ERROR -- LENGTH OF RECEIVED FRAME <> LENGTH OF SENT FRAME. (LANDADERR 407)`<br>`*** Length of actual frame sent = !.`<br>`*** Length of actual frame received = !.` |
| CAUSE | This could be caused by a number of problems: errors on the network, faulty software on the remote node, or faulty software on the local node. |
| ACTION | Retry the test. If the error still occurs, contact HP support representative. |

| | |
|---|---|
| 920 | `*** ERROR -- REMOTE RESPONDED, BUT SSAP IS WRONG. (LANDADERR 920)`<br>`*** Expected response SSAP = $01; received SSAP = !.`<br>`*** Probably bound over another application.` |
| CAUSE | The remote node responded from the wrong SSAP. |
| ACTION | Reset the card, and retry the test. |

| | |
|---|---|
| 921 | `*** ERROR -- REMOTE RESPONDED, BUT CONTROL FIELD IS WRONG. (LANDADERR 921)`<br>`*** Expected control field = $F3; received control field = !.`<br>`*** Probably bound over another application.` |
| CAUSE | The remote node responded with the wrong CONTROL field. |
| ACTION | Reset the card, and retry the test. |

| | |
|---|---|
| 922 | `*** ERROR -- REMOTE RESPONDED, BUT LENGTH FIELD IS WRONG. (LANDADERR 922)`<br>`*** Expected length field is one of ! or $0003.`<br>`*** Received length field = !.` |
| CAUSE | Remote node responded to a TEST packet, but sent the wrong length data. |
| ACTION | Re-run the test again. If the same results are obtained, run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative. |

| | |
|---|---|
| **923** | `*** ERROR -- REMOTE RESPONDED, BUT ACTUAL LENGTH <> LENGTH FIELD.`<br>`*** (LANDADERR 923)`<br>`*** Frame length field = !; frame actual length = !.` |
| CAUSE | Remote node responded to a TEST packet, but sent the packet size was incorrect. |
| ACTION | Re-run the test again. If the same results are obtained, run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative. |
| **924** | `*** ERROR -- REMOTE RESPONDED, BUT SEQUENCE NUMBER WRONG.`<br>`(LANDADERR 924)`<br>`*** Expected sequence number = !; received sequence number = !.` |
| CAUSE | Remote node responded to a TEST packet, but sequence number was was incorrect. |
| ACTION | Re-run the test again. If the same results are obtained, run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative. |
| **1020** | `*** ERROR -- REMOTE RESPONDED, BUT SSAP IS WRONG. (LANDADERR 1020)`<br>`*** Expected response SSAP = !; received SSAP = !.`<br>`*** Probably bound over another application.` |
| CAUSE | The remote node responded from the wroug SSAP. |
| ACTION | Reset the card, and retry the test. |
| **1021** | `*** ERROR -- REMOTE RESPONDED, BUT FRAME IS TOO SHORT.`<br>`(LANDADERR 1021)`<br>`*** Expected frame length > 20 bytes; received frame length = !.` |
| CAUSE | Remote node responded to a TEST packet, but frame length was too short. |
| ACTION | Re-run the test again. If the same results are obtained, run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative. |
| **1022** | `*** ERROR -- REMOTE RESPONDED, BUT LENGTH FIELD IS WRONG.`<br>`(LANDADERR 1022)`<br>`*** Expected length field = 6; received length field = !.` |
| CAUSE | Remote node responded to an XID packet, but the length field in the packet was incorrect. |
| ACTION | Re-run the test agaiu. If the same results are obtained, then run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative. |

**1023**

```
*** ERROR -- REMOTE RESPONDED, BUT CONTROL FIELD IS WRONG.
(LANDADERR 1023)
*** Expected control field = $BF; received control field = !.
*** Probably logged over another application.
```

CAUSE
Remote node responded to an XID packet, but the control field in the packet was incorrect.

ACTION
Re-run the test again. If the same results are obtained, then run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative.

---

**1024**

```
*** ERROR -- REMOTE RESPONDED, BUT FORMAT IDENTIFIER IS WRONG.
(LANDADERR 1024)
*** Expected format identifier = $81; received format identifier = !.
```

CAUSE
Remote node responded to an XID packet, but the format identifier field in the packet was incorrect.

ACTION
Re-run the test again. If the same results are obtained, then run diagnostics on the remote node. If the same results are obtained, contact your HP Support Representative.

---

**1025**

```
*** ERROR -- REMOTE RESPONDED, BUT UNKNOWN CLASS OF SERVICE.
(LANDADERR 1025)
*** Expected either $01, $03, $05, or $07;  received !.
```

CAUSE
The remote node responded to an XID packet with an unknown class of service. This could be caused by data corruption on the network or by the remote node having implemented a new class of service unknown to this node.

ACTION
Retry the test. If the same results occur, contact your HP support representative.

---

**5000**

```
*** ERROR -- LANIC ALREADY IN USE BY DIAGNOSTIC SYSTEM.
(LANDADERR 5000)
*** Someone is already diagnosing the LANIC that you requested.
*** It is illegal to have two copies of LANDAD diagnosing the same
LANIC
*** at the same time.
```

CAUSE
LANIC already in use by diagnostic system.

ACTION
Wait until the other user is finished.

---

**5001**

```
*** ERROR -- LANIC DID NOT PASS SELFTEST. (LANDADERR 5001)
*** The LANIC failed its internal selftest.
*** Please replace the LANIC.
```

CAUSE
Hardware Failure on LANIC.

ACTION
Replace the LANIC.

---

**5002**

```
*** ERROR -- LANIC DOES NOT RESPOND. (LANDADERR 5002)
*** The problem may be one of the following:
*** 1) The driver may not be configured properly.
*** 2) The device is not a LANIC (LAN Interface Card).
*** 3) The LANIC is completely inoperable.
*** 4) There is no LANIC in the proper slot of the card cage.
```

CAUSE The problem may be one of the following: 1) The driver may not be configured properly. 2) The device is not a LANIC (LAN Interface Controller). 3) The LANIC is completely inoperable. 4) There is no LANIC in the proper slot of the card cage.

ACTION 1) Check driver configuration. 2) Verify that the device specified is a LANIC. 3) Verify that the LANIC is in the proper slot of the card cage. 4) If the above are correct, replace the LANIC. 5) If this also fails, contact your Hewlett-Packard Service Representative.

---

**5004**

```
*** INTERNAL ERROR -- BAD SOFTWARE STATUS RECEIVED FROM LAN_DAR.
*** (LANDADERR 5004)
```

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5005**

```
*** INTERNAL ERROR -- LAN_DAR INITIALIZATION FAILED.
 (LANDADERR 5005)
```

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5006**

```
*** ERROR -- TRANSCEIVER POWER FUSE BLOWN ON LANIC. (LANDADERR 5006)
*** Replace Fuse on the LANIC.
*** Beware of these other possible problems:
*** 1) Broken transceiver.
*** 2) Shorted power lines in cabling to the transceiver
***    (AUI cable or stub connector cable).
*** 3) Broken LANIC.
```

CAUSE One of the following:
1) Broken transceiver.
2) Shorted power lines in cabling to the transceiver (AUI cable or stub connector cable).
3) Broken LANIC.

ACTION Check for the following hardware problems:
1) Broken transceiver.
2) Shorted power lines in cabling to the transceiver
(AUI cable or stub connector cable).
3) Broken LANIC.

---

**5007**

```
*** ERROR -- DATA COMPARE ERRORS. (LANDADERR 5007)
$ Note: message 26 and messags 27 usually follow this
$ error message.  Theee meeeagee indicate what the data
$ compare errors actually are and the total number of errors.
```

CAUSE The cause could be one of the following: network corruption, software error on the local system, or software error on the remote system.

ACTION Run selftest to attempt to isolate the problem.

---

**5008**

```
*** ERROR -- RETRY FAULT DURING TRANSMIT. (LANDADERR 5008)
*** Time Domain Reflectometer (TDR) = !.
*** Check the following:
***    1) Unterminated NETWORK CABLE; check terminators.
***    2) Open or shorted terminator; check terminator resistance.
***    3) transceiver not connscted to NETWORK CABLE; check transceiver
tee or tap.
***    4) NETWORK CABLE open or shortsd.
***    5) Faulty transceiver.
```

CAUSE Retry fault detected during data transmission.

ACTION Be sure to check for the hardware error conditions in the order listed in the diagnostic message.

---

**5009**

```
*** ERROR -- LOSS OF CARRIER DETECTED DURING TRANSMIT.
 (LANDADERR 5009)
*** Check the following:
***    1) AUI cable not connected to frontplane connector.
***    2) AUI cable not connected to transceiver.
***    3) Broken transceiver.
***    4) Broken AUI cable.
***    5) transceiver TAP shorted.
***    6) NETWORK CABLE shorted.
***    7) INT/EXT transceiver jumper block defective.
***    8) LANIC broken.
```

CAUSE Loss of carrier experienced during transmission.

ACTION Be sure to check for the hardware error conditions in the order listed in the diagnostic message.

---

**5010**

```
*** ERROR -- LOSS OF CARRIER DETECTED DURING TRANSMIT.
 (LANDADERR 5010)
*** Check the following:
***    1) Stub cable not connected to frontplane connector.
***    2) AUI cable not connected to stub cable.
***    3) AUI cable not connected to transceiver.
***    4) Broken transceiver.
***    5) Broken AUI cable.
***    6) Broken stub connector cable.
***    7) Transceiver TAP shorted.
***    8) NETWORK CABLE shorted.
***    9) LANIC broken.
```

CAUSE Loss of carrier experienced during transmission.

ACTION Be sure to check for the hardware error conditions in the order listed in the diagnostic message.

---

**5011**

```
*** ERROR -- LOSS OF CARRIER DETECTED DURING TRANSMIT.
 (LANDADERR 5011)
*** Check the following:
***    1) Transceiver TEE shorted.
***    2) NETWORK CABLE shorted.
***    3) INT/EXT transceiver jumper block defective.
***    4) LANIC broken.
```

CAUSE Loss of carrier experienced during transmission.

ACTION Be sure to check for the hardware error conditions in the order listed in the diagnostic message.

---

**5012**

```
*** ERROR -- LATE COLLISION DETECTED DURING TRANSMISSION.
 (LANDADERR 5012)
```

CAUSE This node was transmitting and detected another node on the network transmitting past the time when the other node should have detected that this node was transmitting.

ACTION Check the following: 1) The transceiver connections are loose or broken. 2) The (Thicklan or Thinlan) network cable is not terminated at both ends. 3) The network cable is shorted. 4) The transceiver that the LAN/Console card is connected to has jabbed the LANIC card. Cycle power to the transceiver to reset it. 5) Some other node on the network is sending data onto the network cable either continually or at inappropriate times. If other nodes on the network exhibit this same failure, this is probably the cause. Run diagnostics on all nodes on the network.

---

**5013** `*** ERROR -- LANIC FAILED SELFTEST. (LANDADERR 5013)`

```
*** REPLACE LANIC.
*** Internal Status:
***     Interface Exception Status = !.
***     Miscellaneous Status       = !.
***     Selftest/loopback Status   = ! !.
```

CAUSE LANIC selftest failed.

ACTION Replace the LANIC.

---

**5014** `*** ERROR -- LOOPBACK FRAME WAS TRANSMITTED SUCCESSFULLY`

```
*** BUT WAS NOT RECEIVED BEFORE THE TIMEOUT. (LANDADERR 5014)
*** This problem is probably a software routing problem.
```

CAUSE This is an internal error, probably a software routing problem.

ACTION Call your HP support representative.

---

**5015** `*** ERROR -- STATUS CALL FAILED. (LANDADERR 5015)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5016** `*** ERROR -- BIND CALL FAILED. (LANDADERR 5016)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5017** `*** ERROR -- UNBIND CALL FAILED. (LANDADERR 5017)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5018** `*** ERROR -- TRANSMIT CALL FAILED. (LANDADERR 5018)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5019** `*** ERROR -- RESET CALL FAILED. (LANDADERR 5019)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5020** `*** ERROR -- SELFTEST CALL FAILED. (LANDADERR 5020)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5021** `*** ERROR -- TRANSMIT ERROR, BUT CANNOT DETERMINE CAUSE.` (LANDADERR 5021)"

```
*** A transmission error has occurred which caused the transmitted
 frame
*** not to reach the NETWORK CABLE.  For some reason (probably "sticky"
*** statistic counters that have reached their maximum count) the
 actual
*** error can not be determined.
```

CAUSE Transmission error due to undetermined cause.

ACTION Reset statistics and re-run the test again to determine cause.

---

**5022** `*** ERROR -- EXTERNAL LOOPBACK CALL FAILED. (LANDADERR 5022)`

CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5023**

```
*** ERROR -- BABBLE ERROR DURING EXTERNAL LOOPBACK.
 (LANDADERR 5023)
*** This error is caused by out of date hardware.
*** Please replace LANIC.
```

CAUSE LANIC is out of date.

ACTION Replace LANIC.

---

**5024**

```
*** ERROR -- FRAMING ERROR DURING EXTERNAL LOOPBACK.
 (LANDADERR 5024)
```

CAUSE Either a bad transceiver or a bad card.

ACTION Replace the transceiver; if this does not fix the problem, replace the card.

---

**5025**

```
*** ERROR -- OVERFLOW ERROR DURING EXTERNAL LOOPBACK.
 (LANDADERR 5025)
```

CAUSE The LAN controller chip on the card has failed.

ACTION Replace the LANIC.

---

**5026** `*** ERROR -- CRC ERROR DURING EXTERNAL LOOPBACK. (LANDADERR 5026)`

CAUSE The transceiver has malfunctioned.

ACTION Replace the transceiver.

---

**5027**

```
*** ERROR -- UNDERFLOW ERROR DURING EXTERNAL LOOPBACK.
 (LANDADERR 5027)
```

CAUSE The LAN controller chip on the card has failed.

ACTION Replace the LANIC.

---

**5028**

```
*** ERROR -- INCORRECT DEVICE TYPE. (LANDADERR 5028)
*** The following may be the cause:
*** 1) The device specified is not a LANIC.
*** 2) There is a software configuration problem.
***    Even though the device is a LANIC, the system
***    thinks that it is some other type of device.
```

CAUSE Either the LANIC has not be configured correctly, or the card in the slot specified is not a LANIC.

ACTION Either reconfigure the LANIC or place a LANIC in the specified slot.

---

**5029**

```
*** ERROR -- EXTERNAL LOOPBACK FAILURE BUT CANNOT ISOLATE
 TO A CAUSE.
*** (LANDADERR 5029)
*** Check the following:
*** 1) The transceiver is connected to the transceiver TAP.
*** 2) The network cable is terminated at both ends.
*** 3) The network cable is shorted.
*** 4) The transceiver that the LANIC is connected to has jabbed the
LANIC.
***    Cycle power to the transceiver to reset it.
*** 5) Broken cabling to the transceiver (AUI cable or stub connector
cable).
*** 6) Some other node on the network is babbling (sending data
 onto the
***    network cable continually).  If other nodes on the network
exhibit
***    this same failure, this is probably the cause.
```

CAUSE One of the following:
1) The transceiver is not connected to the transceiver TAP.
2) The network cable is not terminated at both ends.
3) The network cable is shorted.
4) The transceiver that the LANIC is connected to has jabbed the LANIC.
Cycle power to the transceiver to reset it.
5) Broken cabling to the transceiver (AUI cable or stub connector cable).
6) Some other node on the network is babbling (sending data
onto the network cable continually). If other nodes on the
network exhibit this same failure, this is probably the cause.

ACTION Check for all of the above causes in the order specified.

---

| | |
|---|---|
| **5030** | `*** ERROR -- POWERFAIL MESSAGE RECEIVED FROM DAM. (LANDADERR 5030)`<br>`*** Please run diagnostic again.`<br>`*** LANDAD Aborting.` |
| CAUSE | Power has failed. |
| ACTION | Re-run LANDAD. |
| **5031** | `*** ERROR -- RECEIVE CALL FAILED. (LANDADERR 5031)` |
| CAUSE | Internal software error. |
| ACTION | Contact your HP support representative. |
| **5032** | `*** ERROR -- RESET STATISTICS CALL FAILED. (LANDADERR 5032)` |
| CAUSE | Internal software error. |
| ACTION | Contact your HP support representative. |
| **5033** | `*** ERROR -- THE NOVRAM IC ON THE LANIC HAS FAILED. (LANDADERR 5033)`<br>`*** Please replace either the NOVRAM or the LANIC.`<br>`*** The NOVRAM is HP part number ! or equivalent. It is the`<br>`*** socketed 18 pin Integrated Circuit (IC) on the LANIC.`<br>`*** IMPORTANT NOTE: When you replace the NOVRAM or replace the LANIC,`<br>`*** your station address will change.` |
| CAUSE | The NOVRAM on the LANIC has failed. |
| ACTION | Replace the NOVRAM and run selftest. |
| **5034** | `*** INTERNAL ERROR -- TRANSFER_COUNT WRONG ON GET_LAN_DA_STATUS CALL.`<br>`*** (LANDADERR 5034)`<br>`*** transfer_count = !; expecting between 96 and 480 inclusive.` |
| CAUSE | Internal software error. |
| ACTION | Contact your HP support representative. |
| **5035** | `*** INTERNAL ERROR -- TRANSFER_COUNT WRONG DN BIND_TO_DAM CALL.`<br>`*** (LANDADERR 5035)`<br>`*** transfer_count = !; expecting 12.` |
| CAUSE | Internal software error. |
| ACTION | Contact your HP support representative. |

**5036**
```
*** INTERNAL ERROR -- TRANSFER_COUNT WRONG ON
 TRANSMIT_FRAME CALL.
*** (LANDADERR 5036)
*** transfer_count = !; expecting !.
```
CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5037**
```
*** INTERNAL ERROR -- TRANSFER_COUNT WRONG ON
 RECEIVE_FRAME CALL.
*** (LANDADERR 5037)
*** transfer_count = !; expecting between 17 and 1514 inclusive.
```
CAUSE Internal software error.

ACTION Contact your HP support representative.

---

**5038**
```
 *** ERROR -- THE LANIC HAS ISSUED A PER EVENT AND TURNED
 ITSELF OFF.
*** (LANDADERR 5038)
*** The LANIC has sent an event to the driver indicating that a
*** Protocol ERror (PER) has occurred.  The actual ARQ status byte
*** received by the driver was !.
*** Please report this to your Hewlett-Packard Service Representative.
*** Reset the LANIC.
```
CAUSE The LANIC firmware has detected a discrepency in the protocol with the driver.

ACTION Reset the LANIC and try the test again. If the error still occurs, report the ARQ status byte to your Hewlett-Packard Service Representative.

---

**5039**
```
 *** ERROR -- THE LANIC HAS ISSUED A DOD EVENT AND TURNED
 ITSELF OFF.
*** (LANDADERR 5039)
*** The LANIC has sent an event to the driver indicating that it
*** was Dead or Dying (DOD).  The actual ARQ status byte
*** received by the driver was !.
*** Please report this to your Hewlett-Packard Service Representative.
*** Reset the LANIC.
```
CAUSE Hardware or firmware problem due to inconsistant state.

ACTION Reset the LANIC and re-run the test. If the same results are obtained, replace the LANIC.

---

**5040**

```
*** ERROR -- THE CIO LANIC HAS ISSUED A DOD EVENT AND TURNED
 ITSELF OFF
*** INDICATING THAT IT TRIED 128 CONSECUTIVE TIMES TO TRANSMIT
*** A FRAME ONTO THE NETWORK MEDIA BUT COULD NOT BECAUSE IT WAS
*** DEFERRING TO CARRIER. (LANDADERR 5040)
*** Check that both terminators are connected to the
*** LAN cable and run LANDAD Section 5 (Selftest).
*** Also, one of the following may be at fault:
*** 1) The transceiver that the LANIC is connected to has jabbed the
LANIC.
***    Cycle power to the transceiver to reset it.
*** 2) Some other transceiver on the network is jabbering.
*** 3) Broken cabling to the transceiver (AUI cable or stub connector
cable).
*** 4) There is a DC voltage level on the LAN cable that is causing
***    the transceiver to assert its collision signal.
*** 5) The LANIC is broken.
```

CAUSE One of the hardware malfunctions listed in the error message has occurred.

ACTION Check for all of the possible hardware malfunctions listed in the error message. If the failure is still not found, contract your HP representative.

---

**5041**

```
*** ERROR -- THE DAM REPORTS LOSS DF COMMUNICATION WITH LANIC.
*** (LANDADERR 5041)
*** The actual ARQ status byte reported by the DAM was !.
*** If this byte is $00 or $10, this indicates an I/O system error.
*** Any other value indicates a LANIC error.
*** Please report the state of the LEDs on the LANIC card and the
*** ARQ status value to your Hewlett-Packard Service Representative.
```

CAUSE I/O System error.

ACTION Please report the state of the LEDs on the LANIC card and the ARQ status value to your Hewlett-Packard Service Representative.

---

**5042**

```
*** ERROR -- THE LANIC INDICATES THAT INFINITE DEFERRAL HAS
*** BEEN DETECTED.  (LANDADERR 5042)
*** Check that both terminators are connected to the
*** LAN cable and run LANDAD Section 5 (Selftest).
*** Otherwise, one of the following may be at fault:
*** 1) The transceiver that the LANIC is connected to has jabbed the
LANIC.
***    Cycle power to the transceiver to reset it.
*** 2) Some other transceiver on the network is jabbering.
*** 3) Broken cabling to the transceiver (AUI cable or stub connector
cable).
*** 4) There is a DC voltage level on the LAN cable that is causing
***    the transceiver to assert its collision signal.
*** 5) If the transceiver is not an HP transceiver, both terminators are
not connected
***    to the LAN cable.
*** 6) The LANIC is broken.
```

CAUSE One of the hardware malfunctions listed in the error message has occurred.

ACTION Check for all of the hardware malfunctions in the error message, fix the problem, and re-run selftest.

---

**5043**

```
*** INSUFFICIENT SECURITY LEVEL.  (LANDADERR 5043)
```

CAUSE Your user security level does not give you the ability to run destructive tests.

ACTION Contact your system administrator the give you the ability to run destructive tests, or only run non-destructive tests.

---

**5044**

```
*** INT/EXT TRANSCEIVER JUMPER IS MISSING OR MISALIGNED (LANDADERR
5044)
*** The jumper that determines if you are using the internal Thin
transceiver
*** or external AUI connector is either missing or misaligned.
```

CAUSE The transceiver jumper is missing or is misaligned.

ACTION Determine whether you are using an internal or an external transceiver and set the jumper accordingly.

---

**5045**

```
*** ERROR -- THE LANIC HAS ISSUED A STF EVENT AND TURNED
 ITSELF OFF.
*** (LANDADERR 5045)
*** The LANIC has sent an event to the driver indicating that its
*** Selftest Failed (STF).  The actual ARQ status byte
*** received by the driver was !.
*** Please report this to your Hewlett-Packard Service Representative.
*** Please replace the LANIC.
```

CAUSE LANIC selftest failed.

ACTION Report the ARQ status byte to your Hewlett-Packard Service Representative and replace the LANIC.

---

**5046**
```
*** ERROR -- THE LANIC HAS ISSUED A WTF EVENT AND GONE OFFLINE.
*** (LANDADERR 5046)
*** The LANIC has sent an event to the driver indicating that a
*** Warning Test Failed (WTF) occurred.  The actual ARQ status byte
*** received by the driver was !.
```
CAUSE — An error has occurred which affects some portion of the LANIC. This message is usually followed by a message to replace a specific part.

ACTION — Replace the part as directed in the message following this message. If there is no additional message, run Selftest to determine the failure.

---

**5047**
```
*** ERROR -- CREATE_MANAGER CALL FAILED. (LANDADERR 5047)
```
CAUSE — This can occur on an MPE/XL system only. The Device Access Routine could not create the driver for LANDAD, reason unknown.

ACTION — Reboot the system and try the test again. If the same results occur, contact your HP Service Representative.

---

**5048**
```
*** ERROR -- DELETE_MANAGER CALL FAILED. (LANDADERR 5048)
```
CAUSE — This can occur on an MPE/XL system only. The Device Access Routine could not delete the driver for LANDAD, reason unknown.

ACTION — Reboot the system and try the test again. If the same results occur, contact your HP Service Representative.

---

**6000**
```
*** WARNING -- DESTRUCTIVE SECTIONS CAN NOT BE RUN FROM YOUR
 TERMINAL.
*** DESTRUCTIVE SECTIONS MAY ONLY BE RUN FROM A TERMINAL THAT IS
 NOT CONNECTED
*** THROUGH THE LANIC TO BE DIAGNOSED.
*** THE FOLLOWING SECTIONS CAN BE PERFORMED FROM YOUR TERMINAL:
1,3,4,6,7,9,10.
*** NO OTHER SECTIONS CAN BE SPECIFIED. (LANDADWARN 6000)
```
CAUSE — Wrong terminal type.

ACTION — Change to a terminal which is directly connected to the LANIC or do not run destructive tests.

---

6002

```
*** WARNING -- NO SQE HEARTBEAT DETECTED WHEN USING
*** "SQE" JUMPER CONFIGURATION. (LANDADWARN 6002)
*** The No SQE Heartbeat condition was detected.
*** Run Section 7 Step 72 (Reset Link Statistics). Then
 run
*** Section 4 (Local Loopback) with LOOPCOUNT=10. Next run
 Section 7
*** (Link Statistics). If six or more of the transmit attempts
 have caused
*** the "No heartbeat detected after transmission" statistic to
 increment,
*** a hardware fault exists.
*** Possible sources are:
*** 1) If you are using an HP28664A Starlan10 transceiver, make certain
***    that the SQE enable switch in in the ENABLED position.
*** 2) An Ethernet 1.0 transceiver is connected. This is an illegal
***    configuration. Set the SQE option to "no SQE".
*** 3) The transceiver is broken.
*** 4) The AUI cable is broken.
*** 5) The stub connector cable is broken.
*** 6) The LANIC is broken.
```

CAUSE Possible sources are: 1) If you are using an HP28664A Starlan10 transceiver, make certain that the SQE enable switch in in the ENABLED position. 2) An Ethernet 1.0 transceiver is connected. This is an illegal configuration. Set the SQE option to "no SQE". 3) The transceiver is broken. 4) The AUI cable is broken. 5) The stub connector cable is broken. 6) The LANIC is broken.

ACTION Run Section 7 Step 72 (Reset Link Statistics). Then run Section 4 (Local Loopback) with LOOPCOUNT=10. Next run Section 7 (Link Statistics). If six or more of the transmit attempts have caused the "No heartbeat detected after transmission" statistic to increment, a hardware fault exists. Check the following:

1) If you are using an HP28664A Starlan10 transceiver, make certain that the SQE enable switch in in the ENABLED position. 2) An Ethernet 1.0 transceiver is connected. This is an illegal configuration. Set the SQE option to "no SQE". an IEEE 802.3 compatible transceiver (HP30241A,HP28641A or HP28664A). 3) Swap the transceiver with a transceiver which is known to be good. Re-run the test. If the results are correct, then replace transceiver. 4) Swap the AUI cable with an AUI cable which is known to be good. Re-run the test. If the results are correct, then replace AUI cable. 5) Swap the stub connector cable with a stub connector cable which is known to be good. Re-run the test. If the results are correct, then replace stub connector cable. 6) If all other actions fail, the LANIC is broken. Replace the LANIC.

| | |
|---|---|
| **6003** | `*** WARNING -- LANDAD CAN ONLY TEST HP TRANSCEIVERS. (LANDADWARN 6003)`<br>`*** The diagnostic does not know how non HP transceivers react to stimuli`<br>`*** and results may not be as expected. If you really want to run`<br>`*** this section on a non HP transceiver, answer 1 to the question`<br>`*** and proceed at your own risk.` |
| CAUSE | The user input OTHER type transceiver. This indicates to the diagnostic that the LANIC is attached to a transceiver which has not been tested with the diagnostic. |
| ACTION | If you wish to proceed, answer 1 to the question. If you wish to indicate another transceiver type, then input the correct type. |
| **6004** | `*** WARNING -- DIAGNOSTIC SSAP IN USE BY ANOTHER PROCESS. (LANDADWARN 6004)`<br>`*** Another process has bound to the diagnostic SSAP.`<br>`*** This section can only be run after the other process finishes.` |
| CAUSE | You have exhausted all tries to obtain a SSAP. |
| ACTION | Try again. If it still fails, reboot the system. |
| **6005** | `*** WARNING -- LANIC IS OFFLINE. (LANDADWARN 6005)`<br>`*** The LANIC must be online to perform this test.`<br>`*** This indicates that you may have a bad LANIC.`<br>`*** Please run section 5 (selftest) to determine if your`<br>`*** LANIC is defective. If selftest passes, the LANIC`<br>`*** will be returned to the online state.` |
| CAUSE | LANIC is offline. LANIC may be defective. |
| ACTION | This indicates that you may have a bad LANIC. Please run section 5 (selftest) to determine if your LANIC is defective. If selftest passes, the LANIC will be returned to the online state. If selftest fails, replace the LANIC. |
| **6006** | `*** WARNING -- SSAP ! ALREADY IN USE BY ANOTHER PROCESS. (LANDADWARN 6006)` |
| CAUSE | LANDAD has probably been previously aborted and the SSAP specified has not been unbound. |
| ACTION | Enter another SSAP. |
| **6007** | `*** WARNING -- NO VALID SECTIONS WERE SPECIFIED. LANDADWARN 6007)`<br>`*** Valid sections are 1-12 inclusive.` |
| CAUSE | An invalid section was specified to be run. |
| ACTION | Input a valid section number. |

**6008**

```
*** WARNING -- DO NOT HIT THE BREAK KEY.
(LANDADWARN 6008)
*** This instance of the driver was created by LANDAD.
*** If you hit the BREAK key and abort LANDAD, system
*** resources will not be recovered and you will have
*** to reboot the system in order to use networking.
```

CAUSE: This warning message is seen on MPE/XL only. It is a temporary solution to a driver problem which will be fixed in MPE/XL Version 4.0.

ACTION: Do not hit the BREAK key and select the ABORT option or you will have to reboot the system to use networking.

---

**6009**

```
*** WARNING -- ERROR IN IO PATH.
(LANDADWARN 6009)
*** Any subsequent errors may be associated with
*** a malfunction in a hardware module which is in
*** the path of the LANIC being tested.
```

CAUSE: Something in the path between LANDAD and the LANIC is not present or is malfunctioning.

ACTION: Verify that you have the LANIC in the correct slot of the card cage and that you are using the correct parameters in the RUN command line.

---

# Contents

# 6

# SCSI Disk Diagnostic

## Introduction

The SCSI Disk Diagnostic (SCSIDSK2) is designed to provide a means of testing SCSI disks on any system that supports the online diagnostics subsystem. The disks that will be tested using this diagnostic are those disks that utilize the Small Computer System Interface (SCSI) message protocol for communication with the SPU. Tests provided can:

- Perform an extensive fault isolating diagnostic trouble tree on the disk and it's system interface. Defects encountered will be relayed to the user.
- Verify the integrity of the SCSI data path to the selected disk.
- Identify the product type of the selected disk.
- Perform the internal selftest on the disk.
- Obtain and decode status messages from the disk.
- Test common transactions that are used in communication between the disk and CPU to determine the integrity of these transactions.

In addition, an interactive external exerciser is provided. This exerciser provides access to internal disk diagnostics, logs, and utilities.

## Defects and Enhancements

Submit defect reports and enhancement requests for this diagnostic through the STARS database referencing Product Number 30600-10054.

## Auto-Diagnostics

If SCSIDSK2 is run as an auto-diagnostic, the following default section will be run:

**Section 10** Diagnostic Trouble Tree

## Minimum Configuration

The hardware running beneath SCSIDSK2 is of no concern to the diagnostic, as long as the online diagnostics subsystem is supported and there is at least one SCSI disk configured on the system.

## Operating Instructions

The SCSI Disk Diagnostic is part of the total online diagnostics subsystem package. It is designed to provide the user with both an on-line and off-line means of thoroughly testing any SCSI disk on the system. With the on-line version, the functionality of system disks is limited to Non-Exclusive/Non-Destructive commands.

SCSIDSK2 can be run in two different modes as described below:

**Non-Exclusive/Non-Destructive** indicates that the user can only run tests on the device that are non-destructive to data on that device as well as other commands that may be executing. An example of a test that would not be allowed in this mode is the WTR Media Test.

**Exclusive/Destructive** - indicates that the user may run any test on the device. There are virtually no restrictions placed on the user in this mode and, therefore, extreme caution should be exercised by anyone running tests in Exclusive/Destructive Mode.

## Default Tests

If no sections or steps are designated by the user in the RUN command the following default section will be run:

**Section 10** Diagnostic Trouble Tree

## RUN Command

The SCSI disk diagnostic can be accessed by the user via the Diagnostic User Interface. It is initiated using the **run** SCSIDSK2 command. Please refer to the DUI chapter for details concerning this command and its parameters. All parameters available in the run command are acceptable as parameters when running this diagnostic. Note that if the ERRONLY parameter is set "on", only error messages will be output by this diagnostic. Error messages can be distinguished from other messages by three "*"'s preceding the text of the message (i.e., '*** MESSAGE' is an error message whereas 'Message' is not). Also note that error messages are in all capital letters and other messages use some lower case. As noted in the previous section, certain modes are required in order to do certain tests. The mode needed to execute each command will be noted in the description of that command.

## Test Execution

When SCSIDSK2 is run, the following header and welcome message will be displayed:

```
************************************************************************
******                                                           ******
******                  SCSI DISK DIAGNOSTIC                     ******
******                                                           ******
******        (C) Copyright Hewlett Packard Co.  1990            ******
******                  All Rights Reserved                      ******
******                    Version  n.nn.nn                       ******
******                                                           ******
************************************************************************

Welcome, Today is FRI, Aug 5, 1982  7:39 AM
```

At this point, the diagnostic calls IO_Path_Test, which is a diagnostic procedure that tests the I/O path to the device. If the status returned from this procedure call is "fail", the following message will be output:

```
*** WARNING -- THE I/O PATH TO THE DISK MAY NOT BE FUNCTIONING
    PROPERLY (SCD2ERR 100)
```

Otherwise,the diagnostic issues an inquiry to the specified device to determine whether or not it is a SCSI disk. If the device does not respond to the inquiry command, the following message will be output:

```
*** DEVICE FAILED TO RESPOND TO INQUIRY COMMAND IN ALLOTTED TIME
(SCD2ERR 101)
```

A second inquiry command is then issued. If this command fails, the following messages are displayed:

```
*** DEVICE FAILED TO RESPOND TO INQUIRY COMMAND IN ALLOTTED TIME
(SCD2ERR 101)
```

```
*** SCSI INQUIRY COMMAND FAILED.  DEVICE MAY NOT BE SCSI OR INQUIRY
INFORMATION RETURNED MAY BE ERRONEOUS.  SCSIDSK2 MAY NOT FUNCTION
PROPERLY (SCD2ERR 207)

Do you with to continue (Y/N)[N]:
```

If the user answers no the diagnostic will terminate immediately.

If a response was obtained, then the returned status is examined to determine if the device is a SCSI disk. If not, the following message will be displayed:

```
*** SCSIDSK2 IS UNABLE TO DIAGNOSE THE SELECTED DEVICE (SCD2ERR 102)
```

If this message is generated, the diagnostic will terminate immediately after outputting it.

At this point, the sections specified by the user will be executed and the results output. If the user did not specify any sections the default section will be executed. If at any time, the number of errors generated reaches the limit specified by the user in the ERRCOUNT parameter of the run command, the following message will be output:

```
*** THE MAXIMUM NUMBER OF ERROR MESSAGES HAS BEEN EXCEEDED (SCD2ERR 110)
```

The diagnostic will then terminate execution. If the ERRPAUSE parameter of the RUN command was assigned a value of "on", then this diagnostic will stop after each error is generated and ask the user if the test should continue:

```
Do you wish to continue (Y/N)[Y]?
```

If the response is Y, then the test will be resumed (if possible); if the response is N, this diagnostic will terminate. the [Y] indicates that Y will be the default response if the user simply hits <CR> in response to the prompt. If the sections specified by the user were executed the number of times specified in the LOOP parameter of the run command without the number of errors exceeding the ERRNUM value, the diagnostic will terminate normally and the following message output:

```
SCSI Disk Diagnostic Exiting . . .
```

Upon termination of this diagnostic, control will return to the diagnostics subsystem.

## Test Section Descriptions

This section is devoted to explaining each section in SCSIDSK2. For each section, this explanation will consist of a description of the section, including the actions performed therein, the expected output from that section, and any error messages that may be generated that are worth noting. Please note that in regard to the error messages, all possible error messages that may be generated are not listed. The only error messages that are listed are those that are considered to be of special significance. For a complete list of error messages that may be generated while running SCSIDSK2, please refer to the final section of this chapter.

## Section 10—DIAGNOSTIC TROUBLE TREE

This section will execute the fault isolating diagnostic trouble tree. The algorithm follows:

1. Issue an internal power-on SELFTEST to the device.
2. Read block twice and compare the data.

**OUTPUT:**

```
Section 10 -- Diagnostic Trouble Tree

Device Selftest Completed

Read Block Completed

End of Section 10 -- Diagnostic Trouble Tree
```

**POSSIBLE ERROR MESSAGES**

```
*** DEVICE ENCOUNTERED AN ERROR WHILE EXECUTING THE
    SCSI INITIATE DIAGNOSTIC COMMAND (SCD2ERR 109)

    SENSE = { sense }

{Note: -- This sense printout will include the failing field
          replaceable unit(s) as specified by the device}
```

## Section 17—EXTERNAL EXERCISER

The SCSI External Exerciser is an interactive program which provides the user with access to the set of internal diagnostics, logs and utilities within a SCSI disk. The purpose of the exerciser is to aid service-trained personnel in troubleshooting SCSI disks to a field replaceable unit level.

The External Exerciser, as discussed earlier, is an interactive program that provides the user with access to the set of internal diagnostics, logs and utilities within a SCSI disk.

This section is divided into two sections. The first contains general information concerning the types of information the exerciser can provide and how that information is generated. The second section contains complete descriptions of the commands available in the exerciser.

### Media Testing

Media tests are powerful tools used to determine media integrity within a SCSI device. These tests can find recoverable and unrecoverable read errors and provide information concerning each error, such as the address where the error occurred, the type of error, and the number of times it has occurred. This information can be logged on the disk maintenance tracks, which are reserved for such use. These tracks provide non-volatile storage, not only for media test errors, but also for spare track addresses, drive faults, and special worst case data patterns which are written on the disk in certain media tests.

All media tests allow the user to input a loop count when requesting the test. Each time an error is detected during the test, the test will stop, log the error, and then resume testing until the loop count has been satisfied. Note that the loop count is not a count of the number of errors, but rather the number of passes the device will execute during the media test. The following information will be reported for each error that occurs:

- The current logical address.
- The error type.
- The loop count when the error occurred.
- An error information byte.

There are two general types of media tests that can be performed. The first type, called a read only media test, is a non-destructive test which reads data from the disk and attempts to detect any read errors that occur. There are two tests available in this category:

1. **Selected Area Read Only** Media Test—Sequentially reads the current data on the disk in a specified area of the media.
2. **Random Read Only** Media Test—Reads random sectors of random length data. This allows read errors to be detected on a large portion of the media in a minimum amount of time.

The second type of media test, called a write-then-read media test, is a destructive test which writes data onto the media and subsequently reads it back, attempting to detect such things as sensitive bit patterns, read errors, and media defects. When requesting a media test of this type, the user is allowed to specify two types of tests. The first type of test will test the complete data channel to the device. The second type will only test the device media. This second type is not available on some drives. Two options exist for each type of write-then-read media test:

1. **Selected Area Write-Then-Read Media Test**—Sequentially writes a specified data pattern over a specified area of the media and then reads back all of the data that was written.
2. **Random Write-Then-Read Media Test**—Writes then reads randomly generated data patterns of random length at random locations on the media. This test locates errors that occur over a large area of the media in a minimum amount of time.

### Error Logging

The ACCESS LOG command allows the user to read the entries contained in the disk drive's maintenance log. This information is available for maintenance purposes. Log information is maintained in a RAM table which is initialized from the disk maintenance log on power-on or reset. It is only posted to the disk when an error entry is added. The ACCESS LOG command will always return the information from the RAM log; there is no disk access.

The addresses and block counts may return physical addresses as defined by the PRINT PHYSICAL command.

The ACCESS LOG command returns the contents of the Data Log, which includes the Usage Log and the Data Error Log, or the Hardware Error Log.

The maintenance log may be cleared by the CLEAR LOGS command. When this command is used, ALL logs are cleared.

Note that some drives do not support error logging.

### Sparing

All SCSI disk drives provide the means of replacing defective sectors or tracks with good ones. This operation is referred to as "sparing". Each SCSI drive contains extra tracks which do not appear in the user's data space. These tracks are set aside as spare tracks to be used in the event that a bad block is found somewhere on another track and needs to be removed. By using the SCSI REASSIGN BLOCK command, the spare track can be substituted for the bad track.

The SCSIDSK2 EXTERNAL EXERCISER REASSIGN BLOCK command attempts to perform pseudo-intelligent sparing. Following is the algorithm:

1. Attempt to recover the data from the target block.
2. If data was recovered, perform a reassign block retaining all other track data and rewrite target block data.
3. If data was not recovered from the target block, warn of data loss and prompt to continue.
4. If a single track contains multiple defective blocks the user must select the option of reassigning multiple blocks for the selected track.

### Exerciser Command Descriptions

When the external exerciser is invoked, the following prompt will be displayed to the user:

```
SCSIDSK2>
```

When the prompt appears, the exerciser is waiting for a command from the user. The available commands are listed in this section in alphabetical order accompanied by descriptions. Each command description is in the following format:

**COMMAND NAME**

**SHORT DEFINITION**

Explanation of what the command does and when it should be used.

**INPUT FORMAT:**

**SCSIDSK2> [COMMAND NAME]**

Note that the prompt for this exerciser is SCSIDSK2>. The appearance of this prompt indicates that the exerciser is waiting for the user to input a command.

**OUTPUT FORMAT:**

Information printed as a result of this command being executed

**POSSIBLE ERROR MESSAGES:**

Any error messages worth noting that may be generated as a result of this command. Note that this does not imply that all possible error messages that may be generated are listed here, but rather, only those that are especially worth mentioning in relation to the command.

There are several conventions used throughout this section in the command formats. They are as follows:

| | |
|---|---|
| **nnnn** | - refers to a decimal number of any magnitude that is output by the diagnostic. |
| **vvvv** | - refers to a decimal number of variable magnitude that must be input by the user. |
| **xxxx** | - refers to text displayed or output by the program |
| **H** | - refers to a hexadecimal digit (0-F). |
| **O** | - refers to an octal digit (0-8). |
| **B** | - refers to a binary digit (0-1). |

Any text enclosed in parenthesis indicates that the user is expected to input that text in response to a query from the diagnostic. Text enclosed in square brackets indicates the default response that will be assumed if the user simply hits <CR> in response to the query. For example, the user may be asked if the program should continue by the following question:

```
Do you wish to continue (Y/N)[N]?
```

The text enclosed in the parentheses (i.e., Y / N) indicates that the user is to type either a Y or an N in response to the question. Each option is separated by a /. The default response in this case would be N as indicated by the [N] prior to the question mark. If there is no default response listed (enclosed in square brackets), the user will not be allowed to "default" the input by only entering a <CR>.

Comments in the input and output sequences are enclosed in curly brackets and are not output by the program. For example, following the question in the previous example, the input sequence would probably look something like the following:

```
{if response was N this command will terminate}
```

Error messages will be preceded by three asterisks (***) and will be in all upper case letters. Other messages will not be all upper case and will not be preceded by the asterisks.

If the user enters a command in response to a prompt and that command is not recognized as an exerciser command, the following message will be displayed:

```
*** YOUR RESPONSE WAS INVALID

SCSIDSK2>
```

This message simply means that the command entered is not part of the command set for the external exerciser.

When the user enters a command that is recognized by the external exerciser, that command will then be processed according to the corresponding command description given in one of the following sections of this document. If an error is encountered as a result of issuing a SCSI command to the device, the hardware sense that is returned by the device will be displayed to the user. The status display will consist of the following format:

```
SENSE =

HH HH HH HH
HH HH HH HH
HH HH HH HH
HH HH HH HH
HH HH HH HH
HH HH

Sense Key = xxxx
Sense Code = xxxx

{One or more of the following sense fields may be printed}

Address of event:
   Block address = nnnn
         {or}
   Vector address =
      cylinder = nnnn     head = nnnn     sector = nnnn

Failed field replaceable unit: nnnn

Drive error numbers = HH, HH, HH, HH
```

Note that only the portions of the above status display that correspond to errors indicated by the hardware status variable will be output. This means, for example, that if no Drive Errors were indicated by the status, none would be output.

In order to exit the exerciser, the EXIT command should be entered (see EXIT command description).

## ACCESS LOG

*Non-Exclusive/Non-Destructive Command*

This command is used to retrieve information from the drive's maintenance log. The maintenance log is used by the drive to record the occurrence of various events, such as data errors and hardware faults. Detailed information concerning how the drive maintains the maintenance log is product specific. Logs are not available on all SCSI disk drives. The `ACCESS LOG` command returns the contents of the Data log (which includes the Usage log and the Data Error log) or the Hardware Error log. A description of each follows:

**Data Log Contents:**

Area indicates what portion of the drive's media is covered by the log data.

Access Count indicates the number of media positionings since the last hardware error.

Blocks Accessed indicates the number of blocks read over the entire disk drive.

First Retry Count indicates the number of times data recovery recovered data with one retry.

Multiple Retry Count indicates the number of times data was not recovered on the first retry.

Address field contains the address of the data block that encountered multiple read retries.

Error Type field indicates the type of data error the block encountered.

Count field indicates the number of times the specified block required multiple retries.

Error byte encodes specific data error information. Content is product specific.

**Hardware Error Log Contents:**

Address field contains the address of the data block accessed when the error occurred.

Internal Device Status field contains an error code corresponding to the Additional Sense code returned by the `REQUEST SENSE` command.

Access Count field indicates the number of media accesses.

**INPUT:**

```
SCSIDSK2> ACCESS LOG

   Types of logs:
      D - Data Log
      H - Hardware Error Log
      B - Both Logs
   Which log (D/H/B)[B]?
```

**OUTPUT:**

```
ACCESS LOG UTILITY
PDEV XXXX is an HPXXXX disk drive

{If response was D}
{Print Physical is disabled}
   Usage and Data Error Log
   ========================

   Area = xxxx
   Access Count = nnnn
   Blocks Accessed = nnnn
   First Retry Count = nnnn
   Multiple Retry Count = nnnn

   There are no data log entries
                    {or}
        Logical         Error
      Block Address     Type      Count     Error
   =================    =====     =====     ========
        nnnn            xxxx      nnnn      BBBBBBBB
          .
          .   {repeat for each entry}
          .
        nnnn            xxxx      nnnn      BBBBBBBB
        nnnn            xxxx      nnnn      BBBBBBBB

   Do you wish to see error byte decoding information (Y/N)[N]?
{If response was Y}
   The error types are:
       M-RET = recovered with retries
       M-ECC = recovered with ECC
       UNR   = unrecoverable error
       ***   = decode error manually

   The Error values are :
      XXXXXXX1  unused
      XXXXXX1X  unused
      XXXXX1XX  error recovered with retries
      XXXX1XXX  error recovered with ECC
      XXX1XXXX  unrecoverable error
      XX1XXXXX  sector error
      X1XXXXXX  header error
      1XXXXXXX  unused

{If response was H}
{Print Physical is disabled}

   There are no hardware error log entries
           {or}
```

```
   Hardware Error Log
   ==================

        Logical           (HEX) Internal    Access
      Block Address       Device Status     Count
   =================      =============     ======
         nnnn                 nnnn          nnnn
           .
           .  {repeat for each entry}
           .
         nnnn                 nnnn          nnnn
         nnnn                 nnnn          nnnn

   Do you wish to see the access count decoding information(Y/N)[N]?
{If response was Y}
   Access count values:
     0 = no seeks
     1 = 1 seek
     2 = 2 seeks
     3 = 3 seeks
     4 = 4 seeks
     5 = 5 - 7 seeks                      (1 sec)
     6 = 8 - 200 seeks                    (1-30 sec)
     7 = 201 - 2,000 seeks                (30 sec - 5 min)
     8 = 2,001 - 12,000 seeks             (5-30 mins)
     9 = 12,001 - 25,000 seeks            (30-60 mins)
    10 = 25,001 - 150,000 seeks           (1-6 hours)
    11 = 150,001 - 600,000 seeks          (6-24 hours)
    12 = 600,001 - 4,000,000 seeks        (1-7 days)
    13 = 4,000,001 - 16,000,000 seeks     (1-4 weeks)
    14 = 16,000,001 - 100,000,000 seeks   (1-6 months)
    15 = > 100,000,000 seeks              (> 6 months)

{If response was N}
{Print Physical is enabled}

   There are no hardware error log entries
           {or}
   Hardware Error Log
   ==================

      Physical          Internal      Access
    Block Address     Device Status   Count
   ===============    =============   ======
       nnnn               nnnn        nnnn
           .
           .  {repeat for each entry}
           .
       nnnn               nnnn        nnnn
       nnnn               nnnn        nnnn
```

```
   Do you wish to see the access count decoding information(Y/N)[N]?
{If response was N}

ACCESS LOG UTILITY COMPLETED
```

## ADDRESS

*Non-Exclusive/Non-Destructive Command*

This command allows the user to convert block addresses to 3-vector addresses and visa versa. This conversion will be made using the maximum address for the device that is currently selected. This maximum address will be obtained via the SCSI READ CAPACITY command. Some drives do not support 3-vector addressing. For these drives, the address command will be rejected.

The ADDRESS command is also used to set a global program flag to either three-vector or block addressing mode.

**INPUT:**

```
SCSIDSK2> ADDRESS

   Convert from block or 3-vector addresses (B/V)[V]?

{If response was B}
   Enter block address (<cr> to keep current value) --
      Block address(nnnn - nnnn) = nnnn?

{If response was V}
   Enter 3-vector address (<cr> to keep current values) --
      Cylinder address(nnnn - nnnn) = nnnn?
      Head address(nnnn - nnnn)     = nnnn?
      Sector address(nnnn - nnnn)   = nnnn?
```

**OUTPUT:**

```
ADDRESS UTILITY
PDEV XXXX is an HPXXXX disk drive

{For block addresses to 3-vector addresses}
   Block address nnnn is equivalent to 3-vector address:
      Cylinder = nnnn   Head = nnnn   Sector = nnnn

{For 3-vector addresses to block addresses}
   3-vector address:
      Cylinder = nnnn   Head = nnnn   Sector = nnnn
      is equivalent to block address nnnn

ADDRESS UTILITY COMPLETED
```

## CAPACITY

*Non-Exclusive/Non-Destructive Command*

This command allows the user to determine the maximum capacity, block address of the last addressable block and the current block size.

**INPUT:**

```
SCSIDSK2> CAPACITY
```

**OUTPUT:**

```
CAPACITY UTILITY
PDEV XXXX is an HPXXXX disk drive

   Max Sector Address = nnnn
   Max Head Address = nnnn
   Max Cylinder Address = nnnn
   Max Block Address = nnnn
   Current Block Size = nnnn
   Drive Capacity = nnnn bytes

CAPACITY UTILITY COMPLETED
```

## CLEAR LOGS

*Non-Exclusive/Non-Destructive Command*

This command is used to clear the drive's maintenance log. This consists of the Data log and Hardware Error log. For drives which do not support logs, this command will be rejected.

**INPUT:**

```
SCSIDSK2> CLEAR LOGS

   ****************************************************
   *                    CAUTION                       *
   *     This command will destroy service related    *
   *     information.                                 *
   *                                                  *
   *     ALL logs will be cleared.                    *
   ****************************************************

   Do you wish to continue(Y/N)[N]?
```

**OUTPUT:**

```
CLEAR LOGS UTILITY
PDEV XXXX is an HPXXXX disk drive

   Logs were cleared successfully

CLEAR LOGS UTILITY COMPLETED
```

## DEFECT LIST

*Non-Exclusive/Non-Destructive Command*

This command displays the drives primary defect list and/or the growing defect list. DEFECT LIST is similar to the CS/80 SPARE TABLE command.

The primary defect list contains the permanent flaws detected on the media by the original manufacturer. The growing defect list includes all defects identified after the drive leaves the manufacturer.

**INPUT:**

```
SCSIDSK2> DEFECT LIST

   Defect types:
     G - growing defect list
     P - primary defect list
     B - both lists
   Which list (G/P/B)[G]?
```

**OUTPUT:**

```
   DEFECT LIST UTILITY
   PDEV XXXX is an HPXXXX disk drive

{If response was G}
   Growing Defect List
   ===================

     Physical                      Logical
   Cyl  Head  Sect      Type     Cyl   Head  Sect
   ===============   =========   ===============
   nnnn  nnnn nnnn   User Data   nnnn  nnnn  nnnn
   nnnn  nnnn nnnn   Spare Trk   nnnn  nnnn  nnnn

{If response was P}
   Primary Defect List
   ===================

     Physical                      Logical
   Cyl  Head  Sect      Type     Cyl   Head  Sect
   ===============   =========   ===============
   nnnn  nnnn nnnn   User Data   nnnn  nnnn  nnnn
   nnnn  nnnn nnnn   User Data   nnnn  nnnn  nnnn

   DEFECT LIST UTILITY COMPLETED
```

## DEVICE RESET

*Exclusive/Non-Destructive Command*

This command is used to reset the device. The following will be performed as a result of this command:

- Finish any logical block write in progress
- Abort any command in progress
- Controller initialization
- Initialize spare table
- Initialize saved pages information
- Initialize logs

### INPUT:

```
SCSIDSK2> DEVICE RESET
```

### OUTPUT:

```
DEVICE RESET UTILITY
PDEV XXXX is an HPXXXX disk drive

DEVICE RESET UTILITY COMPLETED
```

## DIAG

*Non-Exclusive/Non-Destructive Command*

This command will initiate the internal power-on self-test. A loop option allows the diagnostic to be repeated a specified number of times. Tests are device dependent, and are full described in the support documentation for each drive.

**INPUT:**

```
SCSIDSK2> DIAG

   * * * * * * * * * * * * * * * * * * * * * * * * *
   *               CAUTION                         *
   *   This command MAY impact system performance  *
   * * * * * * * * * * * * * * * * * * * * * * * * *

   Do you wish to continue (Y/N)[Y]?

   Input the loop count (nnnn <= count <= nnnn)[nnnn]?
```

**OUTPUT:**

```
   INITIATE DIAGNOSTIC UTILITY
   PDEV XXXX is an HPXXXX disk drive

   Diagnostic loop = nnnn

   INITIATE DIAGNOSTIC UTILITY COMPLETED
```

**POSSIBLE ERROR MESSAGES:**

```
*** DEVICE ENCOUNTERED AN ERROR WHILE EXECUTING THE
    SCSI INITIATE DIAGNOSTIC COMMAND

    SENSE = { sense }
```

## EXIT

*Non-Exclusive/Non-Destructive Command*

This command terminates execution of the External Exerciser. It may be entered any time the SCSIDSK2> prompt appears.

**INPUT:**

```
SCSIDSK2> EXIT
```

**OUTPUT:**

```
End of Section 17 - External Exerciser
```

**FORMAT UNIT**

*Exclusive/Destructive Command*

This command allows the user to format the disk's media. The user will be given the option to retain all spares that have been made on the disk or retain only factory spares. All data on the disk will be destroyed by this command. It is essential that extensive media testing be performed and all questionable blocks reassigned after executing this command.

**INPUT:**

```
SCSIDSK2> FORMAT UNIT

   ****************************************************
   *                     CAUTION                      *
   *       This command may destroy user data         *
   ****************************************************

   Do you wish to continue (Y/N)[N]?

   Do you want to:
      A = retain all spares
      P = retain only primary spares
   Which option would you like (A/P)[A]?
```

**OUTPUT:**

```
   FORMAT UNIT UTILITY
   PDEV XXXX is an HPXXXX disk drive

{If response was A}
   format - retain all spares
          {or}
{If response was B}
   format - retain primary spares

   Media is being initialized -- may take several minutes

   Media has been successfully initialized

   FORMAT UNIT UTILITY COMPLETED
```

## HELP

*Non-Exclusive/Non-Destructive Command*

This command provides the user with access to information concerning the commands that are available in the external exerciser. The user may request a list of the available commands accompanied by a brief description of each, or individual command descriptions. For individual commands, the user can request a description and syntax or just the syntax.

**INPUT:**

```
SCSIDSK2> HELP [command name or <cr>]
```

**OUTPUT:**

```
{If no command name was given (i.e. <cr>)}
   The following commands are available:


   ACCESS LOG - Retrieve information from the maintenance log.
   ADDRESS    - Converts block addresses to 3-vector and
                visa versa.
      .
      .
      .
   WTR MT     - Performs a write-then-read media test on the disk.
                The user is given the option of testing the complete
                data path or disk media only.

{If a command was given}
   Do you want a description or just syntax (D/S)[D]?

{If response was D}
   COMMAND DESCRIPTION:
   Description of the command
   Syntax of the command

{If response was S}
   COMMAND SYNTAX:
   Syntax of the command
```

### INQUIRY

*Non-Exclusive/Non-Destructive Command*

This command allows the user to obtain the drive type, media type and firmware revision.

**INPUT:**

```
SCSIDSK2> INQUIRY
```

**OUTPUT:**

```
    INQUIRY UTILITY
    PDEV XXXX is an HPXXXX disk drive

    Inquiry Information:
    ====================

      Product revision number = nnnn
{If HPC221XB}
      Product serial number = xxxx
{If HPC221XB}
      Firmware ID number = nnnn
{If HPC221XB}
      HDA serial number = nnnn
{If HPC221XB}
      SCSI firmware revision = nnnn
{If HPC221XB}
      ESDI firmware revision = nnnn
      Physical addressing is enabled
                {or}
      Physical addressing is disabled
      Three-Vector address mode
                {or}
      Block address mode

    INQUIRY UTILITY COMPLETED
```

## LDEV

*Non-Exclusive/Non-Destructive Command*

The LDEV command allows the user to select a new device to be tested. The user can select the new device by LDEV or PDEV.

**INPUT:**

```
CS80DIAG> LDEV
```

**OUTPUT:**

```
   LDEV UTILITY
   PDEV XXXX is an HPXXXX disk drive

   Input LDEV or PDEV (L/P)[P]?

{If response was L}
   New LDEV>

{If response was P}
   New PDEV>

   PDEV XXXX is an HPXXXX disk drive

   LDEV UTILITY COMPLETED
```

### PRINT PHYSICAL

*Non-Exclusive/Non-Destructive Command*

This command is used to enable the printing of PHYSICAL addresses in the DATA LOG and HARDWARE ERROR LOG.

**INPUT:**

```
SCSIDSK2> PRINT PHYSICAL
```

**OUTPUT:**

```
Print physical address enabled
            {or}
Print physical address disabled
```

## READ

*Non-Exclusive/Non-Destructive Command*

This command allows the user to access any data block on the selected device. Due to potential security compromise, the user will need to possess Level 0 security to use this command.

### INPUT:

```
SCSIDSK2> READ

{If the drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or the drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn)  = nnnn?
```

### OUTPUT:

```
   READ UTILITY
   PDEV XXXX is an HPXXXX disk drive

   The data in hex follows:

         0   1   2   3   4   5   6   7   8   9

  nnnn:  HH  HH  HH  HH  HH  HH  HH  HH  HH  HH   . . . . . . . . .
  nnnn:  HH  HH  HH  HH  HH  HH  HH  HH  HH  HH   . . . . . . . . .
      .
      .
      .
  nnnn:  HH  HH  HH  HH  HH  HH                   . . . . . .


{Note: -- the dots in the preceding table represent non-alpha-numeric
 ASCII characters. Alpha-numeric characters will be printed}

   READ UTILITY COMPLETED
```

## RFBLOCK

*Non-Exclusive/Non-Destructive Command*

This command allows the user to request all available information fields for the specified logical or physical block. This information includes header, data, and ECC field contents. The drive returns a complete image of one physical block. This command is not supported on all drives.

Due to potential security compromise, the user will need to possess Level 0 security to use this command.

**INPUT:**

```
SCSIDSK2> RFBLOCK

   Do you want to access PHYSICAL addresses (Y/N)[N]?

{If response was Y}
   Enter new address (<cr> to keep current value) --
      PHYSICAL cylinder address (nnnn - nnnn) = nnnn?
      PHYSICAL head address (nnnn - nnnn) = nnnn?
      PHYSICAL sector address (nnnn - nnnn) = nnnn?

{If response was N}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn) = nnnn?
```

**OUTPUT:**

```
READ FULL BLOCK UTILITY
PDEV XXXX is an HPXXXX disk drive

==============================================================
Read full block of Cylinder = nnnn, Head = nnnn, Sector = nnnn
==============================================================

Physical Cylinder   = nnnn
Physical Head       = nnnn
Logical Sector      = nnnn
Spare Sector        = nnnn

Header values (Hex):
      0   1   2   3

  0:  HH  HH  HH  HH
```

```
The data follows (Hex):

       0  1  2  3  4  5  6  7  8  9

nnnn:  HH HH HH HH HH HH HH HH HH HH   . . . . . . . . .
nnnn:  HH HH HH HH HH HH HH HH HH HH   . . . . . . . . .
      .
      .
      .
nnnn:  HH HH HH HH HH HH               . . . . . .


{Note: -- the dots in the preceding table represent non-alpha-numeric
 ASCII characters. Alpha-numeric characters will be printed}

  READ FULL BLOCK UTILITY COMPLETED
```

**REASSIGN BLOCK**

*Exclusive/Destructive Command*

This command will allow the user to reassign a block to an address reserved for sparing. Refer to the subsection above entitled "SPARING" for details on the REASSIGN BLOCK operation. Note that this command should NOT be used unless the block being reassigned is known to be defective. This implies that media tests have been run on the suspected area and it consistently generates errors.

**INPUT:**

```
SCSIDSK2> REASSIGN BLOCK

   *****************************************************
   *                     CAUTION                       *
   *       This command may destroy user data          *
   *****************************************************

   Do you wish to continue (Y/N)[N]?

   * * * * * * * * * * * * * * * * * * * * * * * * * *
   * The REASSIGN BLOCK command is intended to be  *
   * used to reassign a single block defect.  The  *
   * provision to handle multiple defects in a     *
   * single command is made to allow recovery from *
   * a situation where multiple defects occur on a *
   * single track.                                 *
   * * * * * * * * * * * * * * * * * * * * * * * * * *

   Defects to reassign on this track (nnnn <= defects <= nnnn)[nnnn]?

{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses(B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current values) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Block address (nnnn - nnnn) = nnnn?
```

```
   Number of read retries (0 <= retries <= 9)[0]?

{If target block data not recovered}
   ************************************************************
   *                          WARNING                         *
   * Data from the above block(s) could not be recovered.     *
   * If you continue this data will be lost and DATA          *
   * RECOVERY PROCEDURES MUST BE PERFORMED.                   *
   ************************************************************

   Do you wish to continue (Y/N)[N]?

{If reassign block failed}
   REASSIGN BLOCK FAILED --
   ************************************************************
   *  The REASSIGN BLOCK command failed.  Perform a MEDIA     *
   *  TEST on the target track to verify that all defective   *
   *  blocks on the track were specified for reassignment.    *
   ************************************************************
```

**OUTPUT:**

```
   REASSIGN BLOCK UTILITY
   PDEV XXXX is an HPXXXX disk drive

   Attempting to read data of target block . . .
{If data was recovered}
   Data was successfully recovered from target block

{If continue with target block data loss or target block data
 was recovered}
   Attempting to REASSIGN BLOCK(S)
{If sucessful}
   Reassign Block(s) Succeeded

{If target block data recovered}
   Attempting to re-write target block(s) data
   All recovered data was successfully re-written to target block(s)

{If reassign block was performed}
   Reassign block (retain track data option) was successful

{If no reassign block performed}
   No reassign block was performed

   REASSIGN BLOCK UTILITY COMPLETED
```

## RO MT

*Non-Exclusive/Non-Destructive Command*

This command is used to initiate a read only media test. Two types of tests are available through this command. The first read only media test allows the user to specify the address at which the test is to start. This test will sequentially read data starting at this address in an attempt to detect any read errors. The second type of test is a random read only media test which uses random addresses and lengths of reads in attempt to detect any read errors.

**INPUT:**

```
SCSIDSK2> RO MT

   * * * * * * * * * * * * * * * * * * * * * * * * *
   *                 CAUTION                       *
   *   This command MAY tie up the system for      *
   *   as long as it takes to finish the test.     *
   * * * * * * * * * * * * * * * * * * * * * * * * *

   Do you wish to continue (Y/N)[Y]?

   Clear ALL logs (Y/N)[N]?

   Types of RO MT's:
     S = selected area
     R = random area
   Enter the test type (S/R)[S]?

{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn) = nnnn?

   Test Area:
     V = volume
     H = head
     C = cylinder
     T = track
     S = sector
   Enter the test area (V/H/C/T/S)[T]?

{If block address mode}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?
```

```
{If 3-vector address mode}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn) = nnnn?

   Input the loop count (nnnn<= count <=nnnn)[nnnn]?
```

**OUTPUT:**

```
   RO MT UTILITY
   PDEV XXXX is an HPXXXX disk drive

   Test Area = Volume starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Head starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Cyl starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Track starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Sec starting at Cyl = nnnn, Head = nnnn, Sec = nnnn

{If no errors detected}
   No errors were detected by the media test
             {or}
   * * * * * * * * * * * * * * * * * *
   *    MEDIA ERRORS DETECTED        *
   * Read the Access Log's Data Log  *
   * * * * * * * * * * * * * * * * * *

   RO MT UTILITY COMPLETED
```

## SEEK

*Non-Exclusive/Non-Destructive Command*

This command causes the drive to seek to a specified address or series of addresses. The command is useful for testing the servo circuitry. Three types of seeks may be performed: Random Seek seeks to a random address; Alternate Seek seeks between two specified addresses; and Butterfly Seek performs a series of seeks over the entire disk surface.

**INPUT:**

```
SCSIDSK2> SEEK

   * * * * * * * * * * * * * * * * * * * * * * * * * * *
   *                CAUTION                            *
   *   This command MAY impact system performance      *
   * * * * * * * * * * * * * * * * * * * * * * * * * * *

   Do you wish to continue (Y/N)[Y]?

   Types of seeks:
     A = alternate seek
     B = butterfly seek
     R = random seek
   Select the seek type (A/B/R)[R]?

{If response was Alternate}
   Input Address #1:

{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn)  = nnnn?

   Input Address #2:

{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
```

```
        Cylinder address (nnnn - nnnn) = nnnn?
        Head address (nnnn - nnnn) = nnnn?
        Sector address (nnnn - nnnn)  = nnnn?

  {If response was Butterfly}
  {If drive supports 3-vector addressing}
     Test all heads (Y/N)[N]?
  {If response was N}
     Input head number (nnnn <= head <= nnnn)[nnnn] ?

     Input the loop count (nnnn <= count <= nnnn)[nnnn]?Λ
```

**OUTPUT:**

```
SERVO TEST UTILITY
PDEV XXXX is an HPXXXX disk drive

Pass nnnn started
Pass nnnn completed

SERVO TEST UTILITY COMPLETED
```

## SUSPEND

*Non-Exclusive/Non-Destructive Command*

The SUSPEND command allows the user to suspend SCSIDSK2 and return to the DUI. The user enters RESUME at the DUI prompt to return to SCSIDSK2.

**INPUT:**

```
SCSIDSK2> SUSPEND
```

**OUTPUT:**

```
DUI>
```

## VERIFY

*Non-Exclusive/Non-Destructive Command*

This command requests that data on the selected or random area of the drive's media be verified by ECC check only. A compare is not performed. The logs can be cleared before performing the verify.

**INPUT:**

```
SCSIDSK2> VERIFY

   * * * * * * * * * * * * * * * * * * * * * * * * *
   *                  CAUTION                      *
   *   This command MAY impact system performance  *
   * * * * * * * * * * * * * * * * * * * * * * * * *

   Do you wish to continue (Y/N)[Y]?

   Clear ALL logs (Y/N)[N]?

   Types of VERIFY's:
     S = selected area
     R = random area
   Enter the test type (S/R)[S]?

{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn) = nnnn?

   Test Area:
     V = volume
     H = head
     C = cylinder
     T = track
     S = sector
   Enter the test area (V/H/C/T/S)[T]?

{If response was U}
   Enter ending address:

{If block address mode}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?
```

```
{If 3-vector address mode}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Sector address (nnnn - nnnn) = nnnn?

   Input the loop count (nnnn<= count <=nnnn)[nnnn]?
```

**OUTPUT:**

```
   VERIFY UTILITY
   PDEV XXXX is an HPXXXX disk drive

   Test Area = Volume starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Head starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Cyl starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Track starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                       {or}
   Test Area = Sec starting at Cyl = nnnn, Head = nnnn, Sec = nnnn

{If no errors detected}
   No errors were detected by the verify
             {or}
   * * * * * * * * * * * * * * * * * *
   *    MEDIA ERRORS DETECTED        *
   * Read the Access Log's Data Log  *
   * * * * * * * * * * * * * * * * * *

   VERIFY UTILITY COMPLETED
```

## WTR MT

*Exclusive/Destructive Command*

This command is used to initiate a write-then-read media test. Two types of tests are available through this command. The first write-then-read media test will test the data channel to the device. The second type will only test the device media. Two options exist for each of these types of tests, the first option performs incremental writes followed by reads across a data area specified by the user. The pattern of data that is to be written and read may be specified by the user or determined randomly. The second option is a random write-then-read media test which does random length writes followed by reads at random locations on the disk. These tests will destroy data on the disk.

**INPUT:**

```
SCSIDSK2> WTR MT

   *****************************************************
   *                     CAUTION                       *
   *        This command may destroy user data         *
   *****************************************************

   Do you wish to continue (Y/N)[N]?

   Clear All logs (Y/N)[N]?

   Levels of media test:
     C = disk media and channel test
     D = disk media only
   Enter the test method (C/D)[D]?

   Types of WTR MT's:
     S = selected area
     R = random area
   Enter the test type (S/R)[S]?

{If selected area}
{If drive supports 3-vector addressing}
   Do you want block or 3-vector addresses (B/V)[V]?

{If response was B or if drive does not support 3-vector addressing}
   Enter new address (<cr> to keep current value) --
      Block address (nnnn - nnnn) = nnnn?

{If response was V}
   Enter new address (<cr> to keep current value) --
      Cylinder address (nnnn - nnnn) = nnnn?
      Head address (nnnn - nnnn) = nnnn?
      Block address (nnnn - nnnn) = nnnn?

{If selected area}
   Test Area:
     V = volume
```

```
      H = head
      C = cylinder
      T = track
      S = sector
    Enter the test area (V/H/C/T/S)[T]?

    Transfer length types:
      R = random transfer length
      U = user input transfer length
    Enter the transfer length type (R/U)[U]?

 {If response was U}
    Input transfer length (nnnn <= sectors <= nnnn)[nnnn]?

    The sources for the data pattern to be used are:
      I - Internal pattern
      U - User input pattern
      R - Random pattern
    Which pattern source would you like (I/U/R)[I]?

 {If response was U}
    Input the pattern in hex:

    Input the loop count (nnnn<= count <=nnnn)[nnnn]?
```

**OUTPUT:**

```
    WTR MT UTILITY
    PDEV XXXX is an HPXXXX disk drive

 {If clear all logs}
    All logs cleared

    Test Area = Volume starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                        {or}
    Test Area = Head starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                        {or}
    Test Area = Cyl starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                        i{or}
    Test Area = Track starting at Cyl = nnnn, Head = nnnn, Sec = nnnn
                        {or}
    Test Area = Sec starting at Cyl = nnnn, Head = nnnn, Sec = nnnn

 {If no errors detected}
    No errors were detected by the media test
              {or}
    * * * * * * * * * * * * * * * * * *
    *    MEDIA ERRORS DETECTED        *
    * Read the Access Log's Data Log  *
    * * * * * * * * * * * * * * * * * *
    WTR MT UTILITY COMPLETED
```

## Error Messages

This section gives a complete list of the error messages that may be generated by SCSIDSK2 along with brief explanations of the meaning of the messages. The messages will be listed in numerical order and are exactly as they appear in the message catalog. Thus, an "!" indicates that a parameter of some sort will be placed in the location marked by the exclamation point.

| | |
|---|---|
| **100** | `*** WARNING -- THE I/O PATH MAY NOT BE FUNCTIONING PROPERLY (SCD2ERR 100)` |
| CAUSE | An error was detected by the Io_Path_Test service while testing the modules on the i/o path preceding the selected device. |
| ACTION | Execute the appropriate diagnostics on the modules preceding the selected device on the i/o path, especially on those that may have been reported as faulty in error messages immediately preceding this message. Note that the results of the execution of this instance of SCSIDSK2 may be invalid. |

| | |
|---|---|
| **101** | `*** DEVICE FAILED TO RESPOND TO ! COMMAND IN ALLOTTED TIME (SCD2ERR 101)` |
| CAUSE | No response to an i/o was received prior to the expiration of the allotted time. |
| ACTION | Verify that the selected disk drive is actually connected to the system. Run SYSMAP, if available, to confirm the presence of the device. |

| | |
|---|---|
| **102** | `*** SCSIDSK2 IS UNABLE TO DIAGNOSE THE SELECTED DEVICE (SCD2ERR 102)` |
| CAUSE | The selected device identified itself as something other than a SCSI drive. |
| ACTION | Determine type of selected device and run the appropriate diagnostic on it. |

| | |
|---|---|
| **103** | `*** SCSI DISK DIAGNOSTIC TERMINATING (SCD2ERR 103)` |
| CAUSE | A fatal error has been encountered. |
| ACTION | The specific error that was encountered should have been reported immediately prior to this message. Follow the action instructions for that error message. |

| | |
|---|---|
| **104** | `*** A RESERVE AREA IS NOT AVAILABLE ON THIS DRIVE -- NO OPERATION WAS PERFORMED. (SCD2WARN 104)` |
| CAUSE | There is no available area on the disk that can be written to without corrupting user data. This can happen if no such area is provided by the system, or if the diagnostic is running in destructive mode, in which the location of the reserve area is unknown to the system since the drive is locked for diagnostics. |
| ACTION | For systems which do not support a reserve area (e.g. HP-UX), reserve area operations are not supported. |

| | |
|---|---|
| **105** | `*** EXCLUSIVE MODE REQUIRED TO EXECUTE THIS COMMAND (SCD2ERR 105)` |
| CAUSE | The diagnostic does not have access to the drive in the mode necessary to execute the selected command. |
| ACTION | The mode granted to a diagnostic for a selected device is system dependent. Use the HELP LDEV command at the DUI for more information. |
| **106** | `*** DESTRUCTIVE MODE REQUIRED TO EXECUTE THIS COMMAND (SCD2ERR 106)` |
| CAUSE | The diagnostic does not have access to the drive in the mode necessary to execute the selected command. |
| ACTION | The mode granted to a diagnostic for a selected device is system dependent. Use the HELP LDEV command at the DUI for more information. |
| **107** | `*** INVALID DEVICE WAS SELECTED (SCD2WARN 107)` |
| CAUSE | The device selected with the LDEV command is not a valid device. |
| ACTION | Verify the PDEV or LDEV input and retry the command. |
| **109** | `*** DEVICE ENCOUNTERED AN ERROR WHILE EXECUTING THE SCSI ! COMMAND (SCD2ERR 109)` |
| CAUSE | The drive reported an error as a result of executing the selected operation. |
| ACTION | The hardware status that is displayed immediately following this message should indicate what sort of problem occurred, including a failing FRU if one is reported. If an FRU is reported, replace it and re-execute this diagnostic. |
| **110** | `*** THE MAXIMUM NUMBER OF ERRORS HAS BEEN EXCEEDED (SCD2ERR 110)` |
| CAUSE | The user specified error limit has been reached. |
| ACTION | If more errors are desired, re-run the diagnostic assigning a larger value to the ERRCOUNT parameter of the run command. |

**111** `*** UNRECOGNIZED COMMAND -- TYPE "HELP" FOR A LIST OF VALID COMMANDS (SCD2ERR 111)`

CAUSE The specified command is not a valid command.

ACTION Use the help facility to obtain a list of the commands that are valid and enter the desired command.

---

**112** `*** UNRECOGNIZED REPLY WAS FOUND (SCD2ERR 112)`

CAUSE The reply that was entered in response to a prompt by the diagnostic is not valid.

ACTION Refer to the prompt that was displayed and enter a response that is within the specified list of valid responses.

---

**113** `*** A NUMERICAL INPUT WAS EXPECTED BUT NOT RECEIVED (SCD2ERR 113)`

CAUSE The reply that was entered in response to a prompt by the diagnostic is not a valid number.

ACTION Re-enter number using only numeric characters and valid special characters (e.g. +, -, , etc.).

---

**114** `*** AN UNEXPECTED ERROR OCCURRED IN THE IO_SCSI DAR (SCD2ERR 114)`

CAUSE A call to the SCSI device access routine resulted in an unexpected status return.

ACTION The specific status generated by Io_Cs80 should have been displayed immediately prior to this error message. Report this set of error messages to support personnel.

---

**115** `*** ERROR IN TRANSMISSION DETECTED DURING READ LOOPBACK TEST: (SCD2ERR 115)`

```
                                        Bit Positions
            Octal Value   Octal Value   In Error
   Byte #   Transmitted   Received        01234567
   ======   ===========   ===========   =============
```

CAUSE One or more bytes of data that were received from the disk as a result of a loopback operation did not contain the expected value(s).

ACTION Data is most likely being corrupted along the data path between the host and the drive. Check all cable connections along the path and re-execute the diagnostic. If errors persist, execute appropriate diagnostics against the modules that lay in the path between the host and the device.

---

| | |
|---|---|
| **118** | `*** ERROR -- EXPECTED ! BYTES FROM THE DEVICE AND RECEIVED ! BYTES (SCD2ERR 118)` |
| CAUSE | The number of bytes in the reply from the device was not what was expected. This is most likely a result of executing the diagnostic on a drive which is not supported by it. |
| ACTION | Verify that the selected device is in the list of supported devices for the diagnostic (LIST ALL from the DUI). If it is, report the problem to support personnel. |
| **119** | `*** THIS COMMAND IS NOT SUPPORTED ON THIS DEVICE (SCD2ERR 119)` |
| CAUSE | The specified operation is not supported by the selected device. |
| ACTION | While other devices supported by the diagnostic may support the specified command, the selected device does not. |
| **120** | `*** FILE SYSTEM ERROR ENCOUNTERED WHILE RETRIEVING A MESSAGE (SCD2ERR 120)` |
| CAUSE | An error was returned while attempting to obtain a message from the catalog. The actual error will have been displayed prior to this message. |
| ACTION | This is a software error. Report to support personnel. |
| **121** | `*** IDENTICAL READS FROM DISK RETURNED NON-IDENTICAL DATA (SCD2ERR 121)` |
| CAUSE | A read operation was performed on the same sector twice and the data from both reads did not match exactly. |
| ACTION | This is most likely a hardware problem with the drive. If loopback executes correctly, the read/write board in the drive is the probable cause of the error. |
| **122** | `*** DATA READ FROM DISK DID NOT MATCH DATA PREVIOUSLY WRITTEN (SCD2ERR 122)` |
| CAUSE | A sector was written to the disk and immediately read back. The data read did not match exactly the data written. |
| ACTION | This is most likely a hardware problem with the drive. If loopback executes correctly, the read/write board in the drive is the probable cause of the error. |

**123** *** SELECTED DEVICE IS NOT A RECOGNIZED HP DEVICE (SCD2ERR 123)

CAUSE
The selected device does not identify itself with a recognized Hewlett-Packard product number or the version of diagnostic does not recognize or support the product.

ACTION
Verify visually the type of product being diagnosed. If the product is a valid Hewlett-Packard product, update the diagnostic to a version that supports that product.

---

**124** *** ERROR IN TRANSMISSION DETECTED DURING READ/WRITE CHANNEL TEST: (SCD2ERR 124)

```
                               Bit
                 Hex    Hex    Positions
                 Value  Value  In Error
Cyl  Hd Sect Byte Trns  Recd   01234567   Time Error Occurred
=================== ====  ====   ========   ==================
```

CAUSE
Data written to the disk does not agree with the data read.

ACTION
Verify the data path to the device using the LOOPBACK command and WTR ERT on the target sector.

---

**140** *** NO OPERATION WAS PERFORMED (SCD2ERR 140)

CAUSE
Due to a previous error, which has already been reported, no operation was performed.

ACTION
Refer to action instructions for previously reported error.

---

**200** *** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO SEND/RECEIVE INFORMATION FROM THE USER (SCD2ERR 200)

CAUSE
Due to a previous error, which has already been reported, the diagnostic was unable to communicate with the user interface process.

ACTION
Refer to action instructions for previously reported error.

---

**201** *** AN ERROR WAS ENCOUNTERED IN ATTEMPTING TO RETRIEVE A MESSAGE FROM THE CATALOG (SCD2ERR 201)

CAUSE
Due to a previous error, which has already been reported, the diagnostic was unable to extract a message from its message catalog.

ACTION
Refer to action instructions for previously reported error.

---

| | |
|---|---|
| **202** | `*** AN ERROR WAS ENCOUNTERED IN ATTEMPTING TO CONVERT A NUMBER TO A STRING (SCD2ERR 202)` |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable to convert a number to a string. |
| ACTION | Refer to action instructions for previously reported error. |
| **203** | `*** AN ERROR WAS ENCOUNTERED IN ATTEMPTING A BIT EXTRACTION OPERATION (SCD2ERR 203)` |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable to extract one or more bits from a number. |
| ACTION | Refer to action instructions for previously reported error. |
| **204** | `*** THE SELECTED DEVICE COULD NOT BE OBTAINED FOR TESTING (SCD2ERR 204)` |
| CAUSE | The DUI was unable to obtain access to the device. Device may be held exclusively by another process. |
| ACTION | Determine the access status of the device and make the corrective requirements. |
| **207** | `*** SCSI DISK INITIALIZATION FAILED. DEVICE MAY NOT BE SCSI OR INFORMATION RETURNED MAY BE ERRONEOUS. SCSIDSK2 MAY NOT FUNCTION PROPERLY (SCD2ERR 207)` |
| CAUSE | The diagnostic was unable to successfully obtain describe data from the drive. Since the diagnostic needs this information to function correctly, the user must make the determination to continue or terminate. |
| ACTION | Verify that the disk is infact a SCSI device and that the hardware path specified is valid. |
| **208** | `*** AN ERROR WAS ENCOUNTERED IN ATTEMPTING TO COMMUNICATE WITH THE DIAGNOSTIC SYSTEM (SCD2ERR 208)` |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable to successfully perform a function which requires communication with the diagnostic system. |
| ACTION | Refer to action instructions for previously reported error. |

| | |
|---|---|
| **209** | ***** YOUR RESPONSE WAS INVALID (SCD2ERR 209)** |
| CAUSE | The data entered in response to a prompt was not valid. |
| ACTION | Refer to the prompt to determine the valid responses for the particular situation and enter one of the specified valid responses. |
| **210** | ***** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO OBTAIN DATA FROM AN I/O BUFFER (SCD2ERR 210)** |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable to get data from its i/o buffer and, therefore cannot obtain data from the device. |
| ACTION | Refer to action instructions for previously reported error. |
| **211** | ***** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO PLACE DATA INTO AN I/O BUFFER (SCD2ERR 211)** |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable place data into its i/o buffer and, therefore, cannot send data to the device. |
| ACTION | Refer to action instructions for previously reported error. |
| **212** | ***** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO OBTAIN AN I/O BUFFER (SCD2ERR 212)** |
| CAUSE | Due to a previous error, which has already been reported, the diagnostic was unable obtain an i/o buffer and therefore, cannot send/receive data to/from the device. |
| ACTION | Refer to action instructions for previously reported error. |
| **213** | ***** SECTION NUMBER ! IS NOT A VALID SECTION (SCD2ERR 213)** |
| CAUSE | The section number input was not a valid section. |
| ACTION | Verify the valid sections and input your selection. |

| | | |
|---|---|---|
| **214** | | `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO INITIALIZE THE PROGRAM (SCD2ERR 214)` |
| CAUSE | | Due to a previous error, which has already been reported, the diagnostic was unable to start. |
| ACTION | | Refer to action instructions for previously reported error. |

| | | |
|---|---|---|
| **215** | | `*** HIGHER SECURITY IS NEEDED TO PERFORM THIS OPERATION (SCD2ERR 215)` |
| CAUSE | | The user requested an operation which is restricted to users with higher security than the user possesses. |
| ACTION | | Contact system administrator if higher security level is desired. |

| | | |
|---|---|---|
| **216** | | `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO RELINQUISH ACCESS TO THE CURRENT DEVICE. PROGRAM TERMINATING. (SCD2ERR 216)` |
| CAUSE | | Due to a previous error, which has already been reported, the diagnostic was unable to relinquish access to the current device. |
| ACTION | | Refer to action instructions for previously reported error. |

| | | |
|---|---|---|
| **217** | | `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO OBTAIN ACCESS TO THE NEW DEVICE. ACCESS WAS NOT OBTAINED. (SCD2ERR 217)` |
| CAUSE | | Due to a previous error, which has already been reported, the diagnostic was unable to obtain access to the current device. |
| ACTION | | Refer to action instructions for previously reported error. |

| | | |
|---|---|---|
| **218** | | `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO OBTAIN ACCESS TO THE OLD DEVICE. PROGRAM TERMINATING. (SCD2WARN 218)` |
| CAUSE | | Due to a previous error, which has already been reported, the diagnostic was unable to regain access to the previous device under test. Access to this device was attempted after access to the new device was not obtained. |
| ACTION | | The device is currently being accessed exclusively by another process. |

**219** `*** THE OPTION TO OBTAIN DISK SPACE IS NOT IMPLEMENTED FOR THIS OPERATING SYSTEM. (SCD2WARN 219)`

CAUSE — The operating system does not currently provide the functionality necessary to execute this command.

ACTION — Communicate the need for this functionality to the appropriate division.

---

**220** `*** AN ERROR OCCURRED WHILE ATTEMPTING TO OBTAIN DISK SPACE. (SCD2ERR 220)`

CAUSE — Due to a previous error, which has already been reported, the diagnostic was unable to obtain disk space.

ACTION — Refer to action instructions for previously reported error.

---

**221** `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO OBTAIN DISK SPACE. DISK SPACE RETURNED IS NOT ADEQUATE TO PERFORM TESTS. (SCD2ERR 221)`

CAUSE — The area obtained from the disk device for test was not adequate to perform any substantial test. (SCD2WARN 221)

ACTION — Contiguous free space on the disk device is fragmented or there does not exist any free space large enough to test.

---

**222** `*** AN ERROR WAS ENCOUNTERED WHILE ATTEMPTING TO RELINQUISH DISK SPACE. (SCD2ERR 222)`

CAUSE — Due to a previous error, which has already been reported, the diagnostic was unable to relinquish disk space.

ACTION — Refer to action instructions for previously reported error.

---

**223** `*** LOGICAL BLOCK SIZE MUST EQUAL PHYSICAL BLOCK SIZE TO INITIATE THIS COMMAND. (SCD2ERR 223)`

CAUSE — The SCSI MODE SELECT parameter set the device block size to a value which differs from the physical block size.

ACTION — Command cannot be executed until the physical block size is the same as the logical block size.

---

**224** `*** SORRY, THREE VECTOR ADDRESSING IS NOT SUPPORTED BY THIS DEVICE. (SCD2ERR 224)`

CAUSE — A request was made to convert address format, but the device being diagnosed does not support 3-vector addressing.

ACTION — This command cannot be executed using the current device.

---