

HP 3000 Series 900 Computers
HP 9000 Series 800 Computers

Online Diagnostics Subsystem Utilities Manual



Edition 4 E1289
Update 1

09740-90021
Printed in U.S.A. 06/90

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

Notice

The information contained in this document is subject to change without 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.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another language without the prior written consent of Hewlett-Packard Company.

© Copyright 1987, 1988, 1989, 1990 by HEWLETT-PACKARD COMPANY

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.

First Edition	February 1987
Second Edition	December 1987
Third Edition	December 1988
Update 1	March 1989
Fourth Edition	December 1989
Update 1	June 1990

List of Effective Pages

The List of Effective Pages gives the date of the most recent version of each page of the manual. To verify that your manual contains the most current information, check the dates printed at the bottom of each page with those listed below. The date on the bottom of each page reflects the edition or subsequent update in which that page was printed.

Effective Page	Date
All	December 1987
All	December 1988
Title Page	March 1989
2	March 1989
3	March 1989
4	March 1989
5-1 to 5-148	March 1989
All	December 1989
1-1 to 1-97	June 1990
5-113 to 5-148	June 1990

Safety Considerations

This product and related documentation must be reviewed for familiarization with safety markings and instructions before operation.

WARNING

The WARNING sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed 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 correctly performed or adhered to, could result in damage to or destruction of 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 Subsystem Utilities for the HP 3000 Series 900 and HP 9000 Series 800 computer systems. 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 described are focused primarily on the hardware troubleshooting environment and require specific training for correct and safe usage. Specifically, this manual describes the Online Diagnostics Subsystem Utility Programs currently supported and descriptions of the subsystem under MPE XL and HP-UX. Information concerning subsystem diagnostics, user interface, and general description can be found in the *Online Diagnostics Subsystem Manual, Volume I* (09740-90028).

Table of Contents

System and Memory Log Analysis Tool (LOGTOOL)

HP-UX Logging Facility (DELOG/DECODE)

System Map (SYSMAP)

Terminal Diagnostic Support Monitor (TERMDSM)

I/O Test Tool (IOTT)

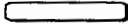
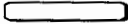
MPE-XL Online Diagnostic Installer (DIAGINST)



Conventions

NOTATION	DESCRIPTION
UPPERCASE	Within syntax statements, characters in uppercase must be entered in exactly the order shown, though you can enter them in either uppercase or lowercase. For example: SHOWJOB Valid entries: showjob ShowJob SHOWJOB Invalid entries: shojwob Shojob SHOW_JOB
<i>italics</i>	Within syntax statements, a word in italics represents a formal parameter or argument that you must replace with an actual value. In the following example, you must replace <i>filename</i> with the name of the file you want to release: RELEASE <i>filename</i>
punctuation	Within syntax statements, punctuation characters (other than brackets, braces, vertical parallel lines, and ellipses) must be entered exactly as shown.
[]	Within syntax statements, brackets enclose optional elements. In the following example, brackets around <i>.TEMP</i> indicate that the parameter and its delimiter are optional: PURGE <i>filename</i>[<i>.TEMP</i>] When several elements within brackets are stacked, you can select any one of the elements or none. In the following example, you can select <i>devicename</i> or <i>deviceclass</i> or neither: SHOWDEV [<i>devicename</i>] [<i>deviceclass</i>]
shading	Within an example of interactive dialog, shaded characters indicate user input or responses to prompts. In the following example, OMEGA is the user's response to the NEW NAME prompt: NEW NAME? OMEGA

Conventions (Continued)

NOTATION	DESCRIPTION
	The symbol  indicates a key on the terminal's keyboard. For example,  indicates the Control key.
 <i>char</i>	 <i>char</i> indicates a control character. For example,  <i>y</i> means you have to simultaneously press the Control key and the <i>y</i> key on the keyboard.
base prefixes	The prefixes %, #, and \$ specify the numerical base of the value that follows: % <i>num</i> specifies an octal number # <i>num</i> specifies a decimal number \$ <i>num</i> specifies a hexadecimal number When no base is specified, decimal is assumed.



Contents

1. System And Memory Log 06/90	
Introduction	1-1
Defects and Enhancements	1-2
Minimum Configuration	1-2
Operating Instructions	1-2
RUN Command	1-2
Test Execution	1-3
Definitions and Conventions	1-3
Parameters	1-5
Command Summary	1-8
Command Descriptions	1-9
DISPLAYLOG	1-10
EXIT	1-12
HELP	1-13
ALTFIELD	1-15
LAYOUT	1-17
LIST	1-19
MEMCLR	1-27
MEMRPT	1-28
MEMTIMER	1-30
PURGESYSLOG	1-31
PURGEWORK	1-32
REDO	1-33
SELECT	1-36
STATUS	1-39
SUSPEND	1-40
SWITCHLOG	1-41
TYPES	1-42
Record Definition Files	1-44
Record Definition Commands	1-45
COMMENT	1-46
RECORD'ID	1-47
TAG'ID	1-48
Record Description Commands	1-49
RECORD'LENGTH	1-50
N'FIELDS	1-51
RECORD'LABEL	1-52
FIELD'FORMAT	1-53
VAR'LABELS	1-56
VALUE'LABELS	1-57
VARIANT	1-58
Tag Usage Example	1-59

RECORD-ID Installation Example	1-62
Error and Warning Messages	1-70

System And Memory Log 06/90

Introduction

The LOGTOOL utility program provides the user with a complete environment in which to identify, purge/clear and display the various log files present within the MPE XL operating system. Further information concerning the MPE XL system tables can be found by referring to the MPE XL System Configuration manual (P/N 32650-90042). LOGTOOL is available in multi-user mode, but the user is required to execute at diagnostic security 1 for some functions.

The host operating system has two types of log files: system log files and the memory error log file. A set of commands is available to handle each type of log file.

System log files contain information generated by the MPE XL operating system. The **LIST** command will display the contents of the system log files based on selection criteria given by the user. The **ALTFIELD** command can be used to specify that only a portion of certain log records found within the system log files are to be displayed by the **LIST** command. Commands are also available to purge log files, create new log files, and to create LOGTOOL "work" files, which will contain a subset of log files. When new log entries are developed, the **LAYOUT** command with a record definition file will allow code developers to "tell" LOGTOOL how to format the information they have sent to the system log files.

The memory error log file contains memory error information gathered by the memory error logging process (MEMLOGP). The **MEMRPT** command will produce a summary report based on the information from the memory log file. The **MEMCLR** command will give the user the ability to clear all the information in the memory log file. In addition, the **MEMTIMER** command allows the user to change the time interval in which the MEMLOGP process will scan all the memory controllers to see if any memory errors occurred.

In addition to the various log file commands, LOGTOOL contains commands which aid the user during a LOGTOOL session.

HELP	Displays detailed and syntax summary information to the user upon request.
REDO	Allows the user to edit/recycle previous input.
SUSPEND	Allows the user to temporarily suspend LOGTOOL and return control 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 30600-10019.

Minimum Configuration

The System and Memory Log Analysis Tool executes on any HP Precision Architecture computer up and running on the MPE XL operating system. The Online Diagnostic subsystem must be supported.

Operating Instructions

RUN Command

To bring up the Online Diagnostic subsystem, enter the following command to the system prompt:

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 LOGTOOL
      |
      |
      | no parameters required to
      | load
```

Test Execution

The utility responds with the following header and welcome message:

```
*****
****                                ****
****                                LOGTOOL                                ****
****                                ****
****                                (C) Copyright Hewlett Packard Co. 1989 ****
****                                All Rights Reserved.                    ****
****                                Version A.02.00                          ****
****                                ****
*****
```

Welcome, Today is < Current date and time >

LOGTOOL initializing, please wait ...

At this point, if the user gave any of the following DUI parameters: LOOP, ERRPAUSE, ERRCOUNT, or ERRONLY,—a warning message will be displayed, indicating that the parameter will be ignored.

The LOGTOOL> prompt will then be displayed. The user is now free to enter commands.

To exit LOGTOOL, the user types EXIT. The message

LOGTOOL Exiting ...

will be displayed to confirm that LOGTOOL completed properly. Control will then be returned to the DUI.

Definitions and Conventions

All keywords, commands and options may be abbreviated to the shortest number of characters that make the reserved word unique. Delimiters for keywords and options include spaces and/or semicolons. The delimiter for items in a list is a comma.

The user may be prompted from time to time for certain data. If one of several specific responses is required (a YES or NO question, for example), the valid responses will be indicated in parentheses. In all instances, a default response is indicated in square brackets. This response may be selected simply by pressing **Return**. For example, a user who is deleting a system log file might be asked the question:

Purge LOG3251.pub.sys (Y/N) [N] ?

Valid responses include Y and N. Simply pressing the **Return** key selects the default response, N. If the text of a command is too long to fit on one line, the user may use continuation lines to complete entry of the command. To accomplish this task, the user simply places an ampersand (&) character as the last character of the line. LOGTOOL will then prompt the user for the rest of the input. A command may consist of numerous continuation lines provided that the total length of the text entered does not exceed 255 characters. For example:

```
LOGTOOL>select log=100/400, 234/599 ;type= 101, 105, 109 &
LOGTOOL>>name=hp7935, hp7933
```

The LOGTOOL> prompt appears slightly different to indicate that a continuation line is expected.

Command lines may include comments if desired. The beginning of a comment is indicated by the "{" character. A "}" or **Return** terminates the comment (i.e., For **Return** everything after the "{" is ignored). The continuation character "&" within a comment is ignored. For example:

```
LOGTOOL>status { This is a comment }
LOGTOOL>select log=1,2 { This is line 1's comment } &
LOGTOOL>>ldev=4    { Note the "&" above was recognized! }
```

Typing a **CTRL-Y** while LOGTOOL is performing a lengthy task (i.e., producing a report or selecting out a subset of error log records) will cause that task to be interrupted. The user may be asked in some cases to choose either to abort or to continue the current task. If the user decides to terminate the task and there was output being directed to a disc file (a report, for example), that file will be closed and preserved as is. Any appropriate cleanup is performed and the user will be returned to the LOGTOOL> prompt.

The following notation appears throughout the remainder of this document. This notation is also used when performing the HELP HELP command.

[]	An element inside brackets is optional.
<>	Items within angle brackets will be replaced with user-supplied information. Do NOT type the angle brackets.
{ }	Items within curly brackets are required. (Usually you must choose one of several options).
...	A horizontal ellipsis indicates that the previous element may be repeated, or that elements have been omitted.
	A vertical bar means "or." For example, (YES NO) means (YES or NO).
::=	Means "is defined as."

Parameters

The following list of parameters will be used to define the functions and options of the LOGTOOL package. The USER COMMANDS section of this document makes constant reference to the terms defined here.

log list	Specifies a list of log file ranges. All system log files are named "LOGxxxx" where "xxxx" is a 4-digit number. <log_list> ::= <log_range> <log_range> , <log_list> <log_range> ::= <log#> <log#> / <log#> <log#> ::= (Any 4 digit integer) Example: LOG=0/5, 253/1004
file list	Specifies a list of file names used as input or output to LOGTOOL. Input and output files will be read/created from the diagnostic group and account (diag.sys). LP can be specified if output is to be directed to the system printer. <file_list> ::= <file_name> <file_name> , <file_list> <file_name> ::= Alpha-numeric text (starting with an alpha) Example: FILE=file1,file8,junk,work

type list	The system error logs contain several different "types" of log records. These record types may be referred to by a type number. (The LOGTOOL command TYPES gives the user a list of the types known to LOGTOOL.) <type_list> ::= <type id> <type id> , <type_list> <type id> ::= Any integer from 0 .. 255 Example: TYPE=101, 104, 106
ldev list	A list of logical devices. <ldev_list> ::= <ldev> <ldev> , <ldev_list> <ldev> ::= (any positive integer) Example: LDEV=1,2,5,10
pdev list	A list of physical devices. Please note that bus converters are applicable to some HPPA systems only. <pdev_list> ::= <pdev> <pdev> , <pdev_list> <pdev> ::= <BUSCONV>/<CA>.<DA>.<device> <BUSCONV>/<CA>.<DA>.<device> <BUSCONV>/<CA>.<DA> <BUSCONV>/<CA> <CA>.<DA>.<device> <CA>.<DA>.<device> <CA>.<DA> <CA> <BUSCONV> ::= <BUSNUM> <BUSNUM>/<BUSNUM> <BUSNUM> ::= An integer giving the bus converter address. <CA> ::= An integer giving the channel address <DA> ::= An integer giving the DA address <device> ::= An integer giving the device address Example: PDEV=1.1, 1.2, 1.2.3, 1.2.1, 3.2.1

Field label keywords

Field label keywords allow the user to select log records based on the data present in the log record. Quotes are only required for field labels and data items which contain embedded blanks or LOGTOOL reserved words used as part of the command syntax. If a quote is needed as part of the field label, entering a quote twice ("") will be interpreted as a single quote as part of the field label. The index construct only applies to fields which are always displayed as hex arrays or can be displayed as hex arrays by using the HEXDUMP option. The selection criteria specified by a field label keyword will only be met if the data item and the data found in the log record match exactly. All field labels and text data items will be upshifted before the evaluation is performed.

<field label> ::= text | "text" | text[<index>] | "text"[<index>]
<index> ::= integer value starting at 1
<data list> ::= <data item> | <data item> , <data list>
<data item> ::= integer value | text | "text"

Ex: Ldev=1,7 "PRODUCT NAME"=HP7935 JSNAME=JREPORT
"Hardware status"[1]=4

DATE & TIME

Date and time formats are language dependent. Machines which support language localization may support numerous date and time formats at one time. The format expected will correspond to the language custom the program is run in.

Standard USA formats are:

<date> ::= <month#> / <day#> / <year>
<day#> ::= 1 .. 31
<month#> ::= 1 .. 12
<year> ::= 1800 .. 2000

<time> ::= <hours> : <minutes> AM | PM
<hours> ::= 0 .. 12
<minutes> ::= 0 .. 59

integer

Any integer value. Any time an integer is expected, that value can be entered in any of 4 bases by preceding the value with one of the base indication characters. If no base character is present, decimal is assumed.

\$ = Hex
% = Octal
B = Binary

Example:

%12 is 10 base ten.
B1001 is 9 base ten.
\$1A is 26 base ten.

Command Summary

Commands are inputs executed immediately after they are received. There are three categories of input commands available to the user:

- System Log File Commands (SFL)
- Memory Log File Commands (MLF)
- Miscellaneous Commands (MC)

The following is a list and brief description of commands available in LOGTOOL.

Command Name	Command Category	Description
ALTFIELD	(SFL)	Specifies which fields of a log record should be displayed in a report generated by LIST.
DISPLAYLOG	(SFL)	Displays I/O entries as information is logged.
EXIT	(MC)	Exits LOGTOOL and returns user to DUI.
HELP	(MC)	Gives help on running LOGTOOL.
LAYOUT	(SLF)	Reads in a layout file.
LIST	(SLF)	Lists contents of a system log file.
MEMCLR	(MLF)	Clears the memory logging process log files.
MEMRPT	(MLF)	Displays the contents of the memory log file.
MENTIMER	(MLF)	Alters the timer value of the memory error logging process.
PURGESYSLOG	(SLF)	Deletes the specified system log files from the disc.
PURGEWORK	(SLF)	Deletes the specified work files from the disc.
REDO	(MC)	Edits any of the last four lines of text entered.
SELECT	(SLF)	Selects specified records from the system log files.
STATUS	(SLF)	Reports on the status of all system log files.
SUSPEND	(MC)	Suspends LOGTOOL and returns control to the DUI.
SWITCHLOG	(SLF)	Causes the system to start a new system log file.
TYPES	(SLF)	Describes the system log file "types."

Command Descriptions

This section explains the standard set of LOGTOOL commands available to the user. (An additional set of commands used in connection with Record Definition Files is explained following this section.) Each description contains the command explanation, syntax definition, parameter definition and an example.

DISPLAYLOG

The DISPLAYLOG command displays I/O and diagnostic log entries as they are logged into the MPE XL log file. The user can terminate this command by typing control-Y. This command accepts a parameter which is a list of either LDEV or PDEV numbers of an I/O device(s). If this parameter is given, only the I/O and diagnostic log entries for those devices will be displayed. Otherwise, all I/O and diagnostic log entries will be displayed.

Syntax:

```
DISPLAYLOG [;LDEV=<LDEV_list>]
           [;PDEV=<PDEV_list>]
```

Parameters:

- LDEV= <LDEV_list> - This is a list of LDEV numbers for I/O devices for which the user would like to see log entries displayed.
- **Example:** LDEV=1,2,5,10
- PDEV= <PDEV_list> - This is a list of PDEV numbers for I/O devices for which the user would like to see log entries displayed.
- **Example:** PDEV=1.1, 1.2, 1.2.3, 1.2.1, 3.2.1.1

Command Example:

```
LOGT00L>displaylog
```

```
I/O ERROR
PRODUCT NAME:      HP7935H          PDEV:          8.0.1
LDEV:              2                DEVICE CLASS:   Hard Disc
I/O EVENT CLASS:   Software          LLIO STATUS:    $DFD700C6
```

```
MPE/XL I/O Status: Proc. Num. = -41, Error Num. = -33, Subsystem = 198
```

```
RETRY SCHEME:      Summarized Retries WILL RETRY:      NO
I/O RESULT:        I/O Failed      RUN AUTODIAG:       NO
RETRY COUNT:       5                MGR PORT NUM.:    -82
TRANS. NUM. :      1279             # HDWR BYTES:      20
HARDWARE STATUS:
```

```
Status =
Unit = 0   Volume = 0
No units with status pending
>>>>>>>> Access Errors <<<<<<<<<
Drive is not ready for access
New target address is :
Vector address =
Cylinder = 0   Head = 0   Sector = 0
```

Block address = 0

DATA LEN:	52	MGR CODE:	121
STATUS INFORMATION:	0	COMMAND BUFFER:	
	1 2 3 4 5 6 7 8 9 10 11 12		
	== == == == == == == == == == == == ==		
1:	40 01 00 0A 20 10 00 00 00 00 00 08 @.....		
13:	34 34 34 18 00 00 00 00 02 00 00 00 4 4 4.....		
MSG DESCRIPTOR:	4	MESSAGE ID:	-1000
TRANSACTION NUM.:	1277	PORT NUMBER:	-82
COMMAND RESIDUE:	0	DATA RESIDUE:	-1
STATUS RESIDUE:	0		

<hit control-y>
LOGTOOL>



EXIT

The **EXIT** command causes LOGTOOL to terminate returning control to the diagnostic user interface.

Syntax: EXIT

Example:

```
LOGTOOL >EXIT
LOGTOOL Exiting...
LOGTOOL (PIN 42) has just terminated
DUI >
```

HELP

Help for any of the LOGTOOL commands may be obtained by using the HELP command. Using the SYNTAX option prints only the syntax of the command. Typing HELP all by itself produces a general description and a list of commands. The symbols used in the HELP syntax output are defined on page 1-5 under the "Notation and Special Symbols" heading.

Syntax:

```
HELP [<command name> [;SYNTAX]]
```

Optional Parameters:

- | | |
|----------------|--|
| <command_name> | - Any valid LOGTOOL command for which help is desired. |
| SYNTAX | - This option causes only the syntax of the indicated to be printed. |

Example:

```
LOGTOOL>HELP
```

The System and Memory Log Analysis Tool (LOGTOOL) enables the user to perform numerous functions on the various system log files. These functions include deleting/clearing of log files and displaying the contents of log files. LOGTOOL will allow the user to manipulate two types of log files.

- o System log files: Contains information logged by the host operating system.
- o The Memory log file: contains information concerning memory errors obtained from memory controller hardware.

LOGTOOL is available in multi-user mode, but the user will be required to execute at diagnostic security 1 to perform some functions.

LOGTOOL command summary:

Command Name	Description
=====	=====
SUSPEND	Return control to the DUI - Suspend LOGTOOL
EXIT	Exit LOGTOOL - return to the DUI
REDO	Edit the last line of text entered
HELP	Gives help on running LOGTOOL
LIST	List contents of a system error log
DISPLAYLOG	Display I/O entries as information is logged
PURGESYSLOG	Delete from disc the specified system error logs
PURGEWORK	Delete from disc the specified "work" files
LAYOUT	Read in layout file
ALTFIELD	Specify which fields are to be displayed in report
SELECT	Select specified records from system log files
STATUS	Report on status of all system log files
SWITCHLOG	Cause the system to start a new system log file
MEMCLR	Clears MEMLOGP log file.
MENTIMER	Alter timer value of MEMLOGP process.
MEMRPT	Display contents of Memory Log File.
TYPES	Describes System Log File "types"
=====	=====

Any command, keyword, or option may be entered by specifying the shortest number of characters which makes the name unambiguous.

Keywords and options may be separated by spaces and/or semicolons.

LOGTOOL>

ALTFIELD

The ALTFIELD command allows the user to identify which fields of a log record are to be displayed in a report generated by the LIST command. LOGTOOL will assume that all fields of all record types are to be displayed unless this command is used to indicate otherwise. Disabling or enabling the display of a field will only effect the fields given in the field label list. All other fields within the record types given by the TYPE parameter will remain unchanged. If the ALL reserved word is used with either the DISABLE or ENABLE keywords, all the fields in the record types given by the TYPE parameter will be changed to the appropriate setting. If the format of a record type is redefined by using the LAYOUT command, all fields for that record type will be re-enabled.

Syntax:

```
ALTFIELD TYPE=<type list>
        ;DISABLE = <field label list> | ENABLE = <field label list>
```

Parameters:

TYPE=<type list> - The TYPE keyword is used to specify which log record types (id's) the fields given in the DISABLE or ENABLE parameter applies to. Log record Id's range from 0 to 255. If this TYPE is not used, all log record types will be considered.

```
<type list> ::= <type number range> | <type number range>,<type list>
<type number range> ::= <type number> | <type number>/<type number>
<type number> ::= integer value
```

Ex: TYPE=101,104,106

DISABLE = <field label list> - Specifies which fields are not to be displayed in a report generated by the LIST command. Quotes are only required for field labels which contain embedded blanks.

```
<field label list> ::= <field label range> |
        <field label range>,<field label list> | ALL
<field label range> ::= <field label> | <field label>/<field label>
<field label> ::= text | "text"
```

Ex: DISABLE="NL CHECKSUM"/"SL CHECKSUM"

ENABLE = <field label list> - Specifies which fields are to be displayed in a report generated by the LIST command. Quotes are only required

for field labels which contain embedded
blanks.

```
<field label list> ::= <field label range> |  
    <field label range>,<field label list> | ALL  
<field label range> ::= <field label> | <field label>/<field label>  
<field label> ::= text | "text"
```

Ex: ENABLE = PDEV/"LLIO STATUS"

Command Example:

```
LOGTOOL> altfield type=111 disable="retry scheme"/"mgr code"  
LOGTOOL>
```

LAYOUT

The LAYOUT command will allow the user to define the format of a particular log entry, which may be found in the system log file. This command requires special files containing the definition of some log file record entries defined in a special language. Please refer to the Record Definition File Formatting section for more detailed information.

Once the record definitions have been accepted, LOGTOOL will display the record entry as specified when performing a LIST command. If any syntax errors are detected by the LAYOUT command, the errors will be displayed to the user and the record definition will not be accepted. Upon completion of the record definition file processing, LAYOUT will display the number of errors detected, the number of record definition accepted and the number of tag definitions accepted. If the APPEND option is given, LOGTOOL will append the new record definitions to the record decoding files (LAYOUTF.DIAG.SYS and LAYMSG.DIAG.SYS). These decoding files are used as a data base of record definitions. The record definitions in the record decoding files will be used by all users when performing a LIST command. If the APPEND option is not given, the record definitions given will be used only by the current session of LOGTOOL and will be lost upon exiting LOGTOOL.

During the processing of a record definition file, if a record definition already exists, the user will be prompted to determine if the previous definition should be overwritten. If the answer YES is given, the previous definition will be destroyed. This prompting can be disabled by giving the NOVERIFY option and the overwrite case is assumed.

To execute this command, the user must possess at least diagnostic security 1. (It is assumed that the record definition files given are located in the DIAG group of the SYS account.)

Syntax:

```
LAYOUT FILES=<file_list>
          [;VERIFY | ;NOVERIFY]
          [;APPEND]
```

Optional Parameters:

FILES - <file_list> The FILES keyword is used to specify the names of the record definition files to be processed. These files must reside in the diagnostic group (DIAG) of the SYS account.

<file_list> ::= <file_name> | <file_name> , <file_list>

<file_name> ::= Alpha-numeric text (starting with an alpha)

Example: FILE=laympexl, layio

APPEND - If given, specifies that the record definitions given are to be saved permanently in LOGTOOL's record decoding file.

VERIFY|NOVERIFY - Specifies if the user is to be asked to overwrite record definitions which already exist.

Example:

```
LOGT00L>layout file=laytemp;noverify
Processing ID 101 ...
Processing ID 102 ...
Processing ID 103 ...
=====

End processing of layout file LAYTEMP.DIAG.SYS

0 Errors encountered
3 out of 3 record'IDs were accepted.
0 out of 0 tag'IDs were accepted.

LOGT00L>
```

LIST

The **LIST** command allows the user to generate a report which will display the contents of the system log files. The user can use the **LIST** command to display all of the records contained in the system log files, or can specify which individual log records are to be displayed using various command options. The report from the **LIST** command may be sent to a file, line printer, or to the terminal (default).

In order for a log record to be included in the report, it must match all of the selection criteria specified. The user can define the selection criteria by using the **TYPE** keyword, **DATE** or **TIME** options, or field label keywords.

TYPE keyword: The **TYPE** keyword can be used to indicate that only log records of the types specified are to be displayed. The **LOGTOOL** command **TYPES** can be used to obtain a list of the log record types available.

DATE or **TIME** options: The **DATE** or **TIME** options can be used to indicate that only the log records generated between the date and time entered are to be displayed.

Field Label Keywords: Field label keywords allow the user to select log records based on the data present in the log record. Whenever the **LIST** command displays a data item from a log record, a field label is shown before the data item to identify what the item represents. This field label is usually a text abbreviation. Field label keywords allow the user to use this text abbreviation as a selection keyword. For example, if the user wishes to see all log records with a field label of "LDEV" and a logged data value of 1, the parameter of "LDEV=1" can be entered as part of the **LIST** command.

If no selection criteria are given, all log records found will be included in the report. After all the selected log records have been displayed, this report will be terminated with a summary indicating the number of log records selected. Please note that if **LOGTOOL** cannot find a record definition for a log record type, the entire log record will be displayed in **HEX**.

To execute this command, the user must possess at least diagnostic security level 1.

Syntax:

```
LIST LOG=<log list> | FILES=<file list>
      [;OUTFILE=<file name>]
      [;TYPE=<type list>]
      [;<field label>=<data list>]
      [;DATE]
      [;TIME]
      [;WIDE]
      [;HEXDUMP]
      [;SUMMARY]
```

Parameters:

LOG=<log list> - The **LOG** keyword specifies which log files to base the report on. Use the **STATUS** command to obtain a list of the system log files available on the host system.

<log list> ::= <log number range> | <log number range> , <log list>

```

<log number range> ::= <log number> | <log number>/<log number>
<log number> ::= integer value

Ex: LOG=100/105, 150

FILES=<file list> - The FILES keyword is optionally used in place
                    of the LOG keyword. The file list must be a
                    list of "work" files (created by the SELECT
                    command). Note: All "work" files are created
                    in the diagnostic group and account (DIAG.SYS).

<file list> ::= <file name> | <file name> , <file list>
<file name> ::= Alpha-numeric text (starting with an alpha)

Ex: FILE=file1,file8,junk,work

OUTFILE=<file name> - The OUTFILE keyword lets the user direct the
                    report to a file or the line printer.
                    Note: All files created by using the OUTFILE
                    parameter will reside in the diagnostic group
                    and account (DIAG.SYS).

<file name> ::= Alpha-numeric text (starting with an alpha) |
                    LP

Ex: OUTFILE=listfile    or

    OUTFILE=lp          {Output to printer}

TYPE=<type list> - The TYPE keyword is used to specify which log
                 record types (id's) to select for the report.
                 Log record Id's range from 0 to 255. If this
                 TYPE is not used, all log record types will be
                 considered.

<type list> ::= <type number range> | <type number range>,<type list>
<type number range> ::= <type number> | <type number>/<type number>
<type number> ::= integer value

Ex: TYPE=101,104,106

<field label>=<data list> - Field label keywords allow the user to
                           select log records based on the data present
                           in the log record. Quotes are only required
                           for field labels and data items which contain
                           embedded blanks or reserved words used as
                           part of the LIST command syntax. If a quote
                           is needed as part of the field label, entering

```

a quote twice ("") will be interpreted as a single quote as part of the field label. The index construct only applies to fields which are always displayed as hex arrays or can be displayed as hex arrays by using the HEXDUMP option. The selection criteria specified by a field label keyword will only be met if the data item and the data found in the log record match exactly. All field labels and text data items will be upshifted before the evaluation is performed.

```
<field label> ::= text | "text" | text[<index>] | "text"[<index>]
<index> ::= integer value starting at 1
<data list> ::= <data item> | <data item> , <data list>
<data item> ::= integer value | text | "text"
```

```
Ex: Ldev=1,7 "PRODUCT NAME"=HP7935 JSNAME=JREPORT
    "Hardware status"[1]=4
```

DATE - The DATE option will cause the user to be prompted for a range of dates. Only those log records falling into one of the specified date ranges will be included in the report. Default is the current day.

```
Ex: Enter the starting date [01/01/86] ? 02/02/86
    Enter the ending date [<starting date>]? 02/03/86
    Do you have more dates (Y/N) [N] ? n
```

TIME - The TIME option will cause the user to be prompted for a range of times. Only those log records falling into one of the specified time ranges will be included in the report. Default is the current time.

```
Ex: Enter the starting time [12:47 PM]? 1:00 PM
    Enter the ending time [<starting time>]? 5:00 PM
    Do you have more to enter (Y/N) [N] ? n
```

WIDE - The WIDE option is used when the output is directed to a device which has 120 or more columns. If the output device is LP, this option is automatically set.

- HEXDUMP - The HEXDUMP option is applicable only for I/O and diagnostic records. If used, the hardware status and diagnostic error stack is dumped in HEX rather than being decoded into text statements.
- SUMMARY - If given, only the log summary portion of the system error log report will be displayed. Display of individual log records are suppressed.

Command Examples:

```
LOGTOOL> list log=1 out=demo
Processing LOG0001.PUB.SYS ...
```

```
      Total records checked = 52.   Records selected = 52.
LOGTOOL>
```

LIST Report Example:

SYSTEM ERROR LOG REPORT
=====

Report Date/Time: TUE, MAR 31, 1987 - 10:34 AM

```
=====
TUE, MAR 31, 1987  10:31 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT)startspool epoc
=====
TUE, MAR 31, 1987  10:31 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(INPUT) STARTSPOOL epoc
=====
TUE, MAR 31, 1987  10:31 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT)streams 10
=====
TUE, MAR 31, 1987  10:31 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(INPUT) STREAMS 10
=====
TUE, MAR 31, 1987  10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT)outfence 14
=====
TUE, MAR 31, 1987  10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(INPUT) OUTFENCE 14
=====
TUE, MAR 31, 1987  10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT)jobfence 7
=====
TUE, MAR 31, 1987  10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 34)
FILE CLOSE
NUM. LOGICAL READS: 0          NUM. BYTES READ: 0
NUM. LOGICAL WRITES:0         NUM. BYTES WRITTEN: 0
FILE NAME:      JSMAIN        GROUP NAME:      PUB
ACCOUNT NAME:    SYS          CREATOR NAME:     MANAGER
USER NAME:       MANAGER      USER GROUP:      PUB
USER ACCOUNT:    SYS          UFID:
      1  2  3  4  5  6  7  8  9 10 11 12
      == == == == == == == == == == == ==
      1: 05 57 00 01 15 4A A8 67 00 00 DB AE . W . . J . g . . .
      13: 2A 82 30 74 02 51 BD CD * . 0 t . Q . .
USER I.D.:      0          CLOSE DISP.:      No Change
=====
```

```

OPEN DOMAIN:      Old Perm File      FILE SIZE (BYTES): 98304
FILE OPEN COUNT:  26
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(INPUT) jobfence 7
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT)stream jsetfoot.pub.sys
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
SPOOLING
USER NAME:        MANAGER          ACCOUNT NAME:      SYS
JOB NAME:         JSETFOOT         FILE NAME:         $STDIN
JOB TYPE:         -2               JOB NUMBER:        2
I/O:              Input Spooler    DEVICE FILE ID:    1
DEVICE TYPE:      0               SPOOLER #:        1
# COPIES:         0               OUT PRIORITY:      0
# REC PROCESSED:  2               # SECTORS USED:    8
SUBTYPE:         0               FUNCTION:          0
#LOG PAGE/PHYS PAGE:0           #PHYSICAL PAGES:  0
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
CONSOLE LOG
(OUTPUT) #J2
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 1)
FILE CLOSE
NUM. LOGICAL READS: 4             NUM. BYTES READ:   216
NUM. LOGICAL WRITES:0            NUM. BYTES WRITTEN: 0
FILE NAME:         JSETFOOT       GROUP NAME:        PUB
ACCOUNT NAME:      SYS            CREATOR NAME:      MANAGER
USER NAME:         MANAGER        USER GROUP:        PUB
USER ACCOUNT:      SYS            UFID:
      1  2  3  4  5  6  7  8  9  10 11 12
      == == == == == == == == == == == ==
1:  05 57 00 01 15 4A A8 67 00 00 D2 AE  . W ... J . g....
13: 2A 82 30 74 02 51 B6 0B      * . 0 t . Q . .
USER I.D.:         0              CLOSE DISP.:        255
OPEN DOMAIN:      Old Perm File    FILE SIZE (BYTES): 4096
FILE OPEN COUNT:  0
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 20)
CONSOLE LOG
(OUTPUT)10:32/#J2/20/LOGON FOR: "JSETFOOT,MANAGER.SYS,PUB" ON LDEV #10.
=====
TUE, MAR 31, 1987 10:32 AM      LOG0009.PUB.SYS      SYSTEM (PIN 26)
FILE CLOSE
NUM. LOGICAL READS: 0             NUM. BYTES READ:   0
NUM. LOGICAL WRITES:0            NUM. BYTES WRITTEN: 0

```

```

FILE NAME:          SETFOOT          GROUP NAME:          PUB
ACCOUNT NAME:       SYS              CREATOR NAME:         MANAGER
USER NAME:          MANAGER          USER GROUP:         PUB
USER ACCOUNT:       SYS              UFID:
      1  2  3  4  5  6  7  8  9  10 11 12
      == == == == == == == == == == == ==
      1: 05 57 00 01 15 4A A8 67 00 00 CF AE . W . . J . g . . .
      13: 2A 82 30 78 02 51 B3 8A * . 0 x . Q . .
USER I.D.:          0                CLOSE DISP.:         No Change
OPEN DOMAIN:        Old Perm File    FILE SIZE (BYTES):  86016
FILE OPEN COUNT:    0
=====
TUE, MAR 31, 1987  10:32 AM    LOG0009.PUB.SYS          SYSTEM (PIN 28)
I/O ERROR
PRODUCT NAME:      HP7935H          PDEV:              8.0.1
LDEV:              2                DEVICE CLASS:      Hard Disc
I/O EVENT CLASS:   Software         LLIO STATUS:       $DFD700C6

MPE/XL I/O Status: Proc. Num. = -41, Error Num. = -33, Subsystem = 198

RETRY SCHEME:      Summarized Retries WILL RETRY:        NO
I/O RESULT:        I/O Failed      RUN AUTODIAG:        NO
RETRY COUNT:       5                MGR PORT NUM.:    -82
TRANS. NUM.:       1279             # HDWR BYTES:      20
HARDWARE STATUS:

Status =
Unit = 0   Volume = 0
No units with status pending
>>>>>>>> Access Errors <<<<<<<<<
Drive is not ready for access
New target address is :
Vector address =
Cylinder = 0   Head = 0   Sector = 0
Block address = 0

DATA LEN:          52                MGR CODE:          121
STATUS INFORMATION: 0                COMMAND BUFFER:
      1  2  3  4  5  6  7  8  9  10 11 12
      == == == == == == == == == == == ==
      1: 40 01 00 0A 20 10 00 00 00 00 00 08 @. . . . .
      13: 34 34 34 18 00 00 00 00 02 00 00 00 4 4 4. . . . .
MSG DESCRIPTOR:     4                MESSAGE ID:       -1000
TRANSACTION NUM.:   1277             PORT NUMBER:      -82
COMMAND RESIDUE:    0                DATA RESIDUE:     -1
STATUS RESIDUE:     0
=====
TUE, MAR 31, 1987  10:32 AM    LOG0009.PUB.SYS          SYSTEM (PIN 20)
JOB TERMINATION
MAX. PRIORITY:      200              NUM PROC. CREATE:  2
CPU SEC.:           3                CONNECT MIN.:      1

```



```
=====
TUE, MAR 31, 1987   10:32 AM       LOG0009.PUB.SYS       SYSTEM (PIN 20)
CONSOLE LOG
(OUTPUT)10:32/#J2/20/LOGOFF ON LDEV #10.
=====
```

TYPE	DESCRIPTION	COUNT
103	JOB TERMINATION	1
105	FILE CLOSE	3
108	SPOOLING	1
111	I/O ERROR	1
115	CONSOLE LOG	12

MEMCLR

The **MEMCLR** command will clear the memory error log file of all data. It is recommended that the user generate a memory report using the **MEMRPT** command before entering this command. The user must possess at least diagnostic security 1.

Syntax:

```
MEMCLR
```

Limitations:

This command affects only the memory error log file.

Example:

```
LOGTOOL>memclr  
LOGTOOL>
```

MEMRPT

The **MEMRPT** command produces a summary report of the memory errors detected by the memory logging process (MEMLOGP). Error information will be displayed for each memory controller present on the system. To execute this command, the user must possess at least diagnostic security level 1.

Syntax:

```
MEMRPT [OUTFILE=<file name>]
      [,WIDE]
```

Optional Parameters:

OUTFILE= <file name> - The output from the command is directed to the user's terminal unless the OUTFILE keyword is present.

<file_name> ::= Alpha-numeric text (starting with an alpha) |

Example: OUTFILE=listfile

WIDE - The WIDE option is used when the output is directed to a device which has 120 or more columns. If the output device is LP this option is automatically set.

Limitations:

This command accesses only the memory error log file.

Example:

```
LOGT00L>memrpt outfile=lp
LOGT00L>
```

Memory Report Example (Series 3000/930 Only):

MEMORY ERROR LOGGING REPORT

SYSTEM START TIME -- TUE, JAN 1, 1985, 3:59 AM
LAST ERROR CHECK TIME -- TUE, JAN 1, 1985, 2:27 AM

LOGGING TIME INTERVAL -- 2000 seconds

MEMORY CONTROLLER # 1

```
-----  
| BIT | CHIP|    1 ST ERROR    |    LAST ERROR    | ERROR |  
| NUM | ID |    DETECTED AT    |    DETECTED AT    | COUNT |  
-----  
| 34 | u1603|01/01/85 1:40 AM |01/01/85 2:09 AM | 9     |  
-----  
| 39 | u1708|04/23/86 6:10 PM |04/23/86 6:10 PM | 1     |  
-----  
|          *** NO MULTI BIT ERRORS HAVE BEEN DETECTED ***          |  
-----  
|          *** NO SINGLE BIT ERRORS HAVE BEEN DETECTED ***          |  
-----
```

MEMORY CONTROLLER # 2

```
-----  
| BIT | CHIP|    1 ST ERROR    |    LAST ERROR    | ERROR |  
| NUM | ID |    DETECTED AT    |    DETECTED AT    | COUNT |  
-----  
|          *** NO MULTI BIT ERRORS HAVE BEEN DETECTED ***          |  
-----  
|          *** NO SINGLE BIT ERRORS HAVE BEEN DETECTED ***          |  
-----
```

MEMORY CONTROLLER # 3

```
-----  
| BIT | CHIP|    1 ST ERROR    |    LAST ERROR    | ERROR |  
| NUM | ID |    DETECTED AT    |    DETECTED AT    | COUNT |  
-----  
|          *** NO MULTI BIT ERRORS HAVE BEEN DETECTED ***          |  
-----  
|          *** NO SINGLE BIT ERRORS HAVE BEEN DETECTED ***          |  
-----
```

MENTIMER

The **MENTIMER** command alters the time interval between successive interrogation of the memory controllers to determine if any memory errors occurred. This error information, from the memory controllers, is logged into the memory log file. The default value (3600 seconds) is retained while the system is down. Altering this timer causes **MEMLOGP** to update the log file immediately and then periodically thereafter according to the new interval. The user must possess at least diagnostic security 1 to execute this command.

Syntax:

MENTIMER SECONDS=<#seconds>

Optional Parameters:

SECONDS= <#seconds> - The time between memory log scans

Example: SEC=3600

Example:

```
LOGTOOL>mentimer sec=9600
LOGTOOL>
```

PURGESYSLOG

The PURGESYSLOG command will delete from disc the specified system log file(s). Included is a verify mode (default) which will ask the user to confirm the deletion of each file specified before the program actually purges it from the system. The user must possess at least diagnostic security 1 to execute this instruction.

Syntax:

```
PURGESYSLOG LOG=<log_list>
                [;VERIFY | ;NOVERIFY]
```

Optional Parameters:

LOG= <log_list> - The LOG keyword specifies which log files to purge.

Example: LOG=1000/1400,3418/3590,3400,4

VERIFY|NOVERIFY - The VERIFY option is default for this command. It causes the user to be prompted before each log file is deleted. If NOVERIFY is selected, the specified log files will be purged without query.

Limitations:

This command affects only the system log files.

Example:

```
LOGT00L>purgesyslog log=4232/4235
Purge LOG4232.PUB.SYS (Y/N) [N] ? Y
...Deleted LOG4232.PUB.SYS
Purge LOG4233.PUB.SYS (Y/N) [N] ? Y
...Deleted LOG4233.PUB.SYS
LOG4234.PUB.SYS was missing
Purge LOG4235.PUB.SYS (Y/N) [N] ? N
...Kept
LOGT00L>
```



PURGEWORK

This command will delete from disc the specified "work" file(s). These working log files are created by using the SELECT command. There are no naming conventions for these files. LOGTOOL can, however, tell the difference between a "work" file and files of any other type. In otherwords, this is NOT a general file purge command. Included is a verify mode (default) which will ask the user to confirm the deletion of each file specified before the program actually purges it from the system. (Remember: All work files are currently created in the group DIAG of the SYS account.)

Syntax:

```
PURGEWORK FILES=<file list>
           [,VERIFY | ;NOVERIFY]
```

Optional Parameters:

FILES= <file_list> - The FILES keyword is used to specify the names of the work files to purge.

Example: FILE=stuff,junk,foo

VERIFY|NOVERIFY - The VERIFY option is default for this command. It causes the user to be prompted before each work file is deleted. If NOVERIFY is selected, the specified work files will be purged without query.

Limitations:

This command affects only work files.

Example:

```
LOGTOOL>purgework file=subset1,subset2,junkfile,logtool
Purge SUBSET1.DIAG.SYS (Y/N) [N] ? Y
...Deleted SUBSET1.DIAG.SYS
SUBSET2.DIAG.SYS does not exist
Purge JUNKFILE.DIAG.SYS (Y/N) [N] ? N
...Kept
LOGTOOL.DIAG.SYS is not a work file - skipped
LOGTOOL>
```

REDO

The REDO command allows you to edit any of the last 4 lines of input entered to LOGTOOL. If redo is given with no parameters, the last command given can be edited. If the “?” is given, a list of the last 4 commands given will be displayed. The user can also give a redo index number or an offset number to indicate which command on the redo stack is to be modified.

Syntax:

```
REDO [? | <index> | <offset>]
```

Parameters:

- | | |
|----------|--|
| ? | - Will display redo stack to user. |
| <index> | - A positive number of the command to be modified. |
| <offset> | - A negative number indicating the offset from the last command given.
“redo -1” is equivalent to “redo.” |

The valid edit characters are:

i	- INSERT: Text following this character is inserted at the location just prior to the location of the "I"
r	- REPLACE: Text following this character overlays existing text starting at the location of the "R"
d	- DELETE: Each "D" will cause the corresponding character to be deleted.
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: Change 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.
other	- Simple replacement: Any other character (not i, r, d, d>, >, >d, or c) 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 the line.

Command Example:

```
LOGTOOL>sel log=1,,4,5 type=11 ldev=1,2 outfile=myoutfile

LOG FILE NUMBER EXPECTED. (LTERR 614)
LOGTOOL>redo
>sel log=1,,4,5 type=11 ldev=1,2 outfile=myoutfile
>      d
>sel log=1,4,5 type=11 ldev=1,2 outfile=myoutfile
>      i,3
>sel log=1,4,5 type=11 ldev=1,2,3 outfile=myoutfile
>      ddd
>sel log=1,4,5 type=11 ldev=1,2,3 outfile=myfile
>      12
>sel log=1,4,5 type=12 ldev=1,2,3 outfile=myfile
>
LOGTOOL>sel log=1,4,5 type=12 ldev=1,2,3 outfile=myfile

( Logtool now processes the corrected select command ... )
```

SELECT

The SELECT command allows the user to create a "work" file containing a subset of the log records found in the system log files. Currently, the work file will be created in the DIAG group of the SYS account.

In order for a log record to be included in the work file, it must match all of the selection criteria specified. The user can define the selection criteria by using the TYPE keyword, DATE or TIME options, or field label keywords.

TYPE keyword: The TYPE keyword can be used to indicate that only log records of the types specified are to be included. The LOGTOOL command TYPES can be used to obtain a list of the log record types available.

DATE or TIME options: The DATE or TIME options can be used to indicate that only the log records generated between the date and time entered are to be included.

Field Label Keywords: Field label keywords allow the user to select log records based on the data present in the log record. Whenever the LIST command displays a data item from a log record, a field label is shown before the data item to identify what the item represents. This field label is usually a text abbreviation. Field label keywords allow the user to use this text abbreviation as a selection keyword. For example, if the user wishes to see all log records with a field label of "LDEV" and a logged data value of 1, the parameter of "LDEV=1" can be entered as part of the LIST command.

If no selection criteria are given, all log records found will be included in the work file.

To execute this command, the user must possess at least diagnostic security level 1.

Syntax:

```
SELECT LOG=<log list>
      ;OUTFILE= <file name>
      [;TYPE=<type list>]
      [;<field label>=<data list>]
      [;DATE]
      [;TIME]
      [;APPEND]
```

Parameters:

LOG=<log list> - The LOG keyword specifies which log files to base the report on. Use the STATUS command to obtain a list of the system log files available on the host system.

<log list> ::= <log number range> | <log number range> , <log list>
<log number range> ::= <log number> | <log number>/<log number>
<log number> ::= integer value

Ex: LOG=100/105, 150

OUTFILE=<file name> - The OUTFILE keyword lets the user specify the work file name.

<file_name> ::= Alpha-numeric text (starting with an alpha)

Ex: OUTFILE=listfile

TYPE=<type list> - The TYPE keyword is used to specify which log record types (id's) to select for the report. Log record Id's range from 0 to 255. If this TYPE is not used, all log record types will be considered.

<type list> ::= <type number range> | <type number range>,<type list>

<type number range> ::= <type number> | <type number>/<type number>

<type number> ::= integer value

Ex: TYPE=101,104,106

<field label>=<data list> - Field label keywords allow the user to select log records based on the data present in the log record. Quotes are only required for field labels and data items which contain embedded blanks or reserved words used as part of the SELECT command syntax. If a quote is needed as part of the field label, entering a quote twice ("") will be interpreted as a single quote as part of the field label. The index construct only applies to fields which are always displayed as hex arrays or can be displayed as hex arrays by using the HEXDUMP option. The selection criteria specified by a field label keyword will only be met if the data item and the data found in the log record match exactly. All field labels and text data items will be upshifted before the evaluation is performed.

<field label> ::= text | "text" | text[<index>] | "text"[<index>]

<index> ::= integer value starting at 1

<data list> ::= <data item> | <data item> , <data list>

<data item> ::= integer value | text | "text"

Ex: Ldev=1,7 "PRODUCT NAME"=HP7935 JSNAME=JREPORT
"Hardware status"[1]=4

DATE - The DATE option will cause the user to be prompted for a range of dates. Only those log records falling into one of the specified date ranges will be included in the report. Default is the current date.

```
Ex: Enter the starting date [01/01/86] ? 02/02/86
Enter the ending date [<starting date>]? 02/03/86
Do you have more dates (Y/N) [N] ? n
```

TIME - The TIME option will cause the user to be prompted for a range of times. Only those log records falling into one of the specified time ranges will be included in the report

```
Ex: Enter the starting time [12:47 PM]? 1:00 PM
Enter the ending time [<starting time>]? 5:00 PM
Do you have more to enter (Y/N) [N] ? n
```

APPEND - The APPEND option will append data to the file specified in the OUTFILE parameter.

Command Example:

```
LOGTOOL>select log=1,2 type=100,105 outf=subset1
Processing LOG0001.PUB.SYS ...
Processing LOG0002.PUB.SYS ...
```

```
Total records checked = 109.  Records selected = 2.
LOGTOOL>
```

STATUS

The STATUS command displays a list of the currently existing system log files. The system log file currently being used by the system is indicated by the presence of a "*" after the file name. If no options are given, only the log file names will be displayed. If the DETAIL option is given, the log file name, the number of records within that log file and the date/time of the first log entry will be displayed.

Syntax:

STATUS [DETAIL]

Optional Parameter:

DETAIL - Causes detailed information (# of records and date/time of first log entry) for all of the system log files to be displayed.

Example:

```
LOGT00L>status

System Log Files:
=====
LOG0001  LOG0002  LOG0003  LOG0004  LOG0005*
LOGT00L>status detail

Log File      Rec #1      Rec #1      Total
Name          Date        Time        #Records
=====
LOG0001      11/10/86    10:49 PM     52
LOG0002      11/17/86    12:46 PM     57
LOG0003                        0
LOG0004      11/22/86    7:22 AM      10
LOG0005*     12/01/86    8:30 AM      10

LOGT00L>
```

SUSPEND

The **SUSPEND** command returns control to the diagnostic user interface. LOGTOOL is temporarily suspended and may be resumed by typing **RESUME**.

Syntax:

```
SUSPEND
```

Example:

```
LOGTOOL>suspend
LOGTOOL SUSPENDED - [Type RESUME to continue]
DUI>(any DUI commands)
DUI>
DUI>resume
LOGTOOL resuming from SUSPEND mode ...
LOGTOOL>
```

SWITCHLOG

The SWITCHLOG command causes a new system log file to be opened and the old one closed. Display from the MPE XL SHOWLOG command and MPE XL SWITCHLOG command will appear on the system console. The LOGTOOL STATUS command may be used to obtain the new log file name. To execute this command, the user must possess at least diagnostic security 1.

Syntax:

```
SWITCHLOG
```

Limitations:

This command affects only the system log files.

Examples:

```
LOGTOOL>switchlog  
LOGTOOL>
```


TYPES

The **TYPES** command lists all the record identifiers currently defined within LOGTOOL. This command is useful for users who wish to display specific types of log records, or need specific information concerning how a log record will be formatted. When this command is executed without any options or keyword, the record ID number and the record label used when the log entry is displayed, will be listed. If the **DETAIL** option is given, the full definition of the record ID will be given. The information displayed will conform to the data given when the record definition file was processed. If the **SUMMARY** option is given, a list of all field label keywords, data field labels, and whether the field will be displayed in a **LIST** report, will be displayed for each log record. If the user needs information for a specific log record, the **TYPE=** keyword may be used to display information for a specify log record ID.

Syntax:

```
TYPES [TYPE=<type list>]
      [;DETAIL | ;SUMMARY]
```

Parameter:

- TYPE=<type list> - The TYPE keyword is used to specify which log record types (id's) to display information for. Log record Id's range from 0 to 255. If this TYPE is not used, all log record types will be displayed.
- <type list> ::= <type number range> | <type number range>,<type list>
<type number range> ::= <type number> | <type number>/<type number>
<type number> ::= integer value
- Ex: TYPE=101,104,106
- DETAIL - Causes detailed information for each record ID to be displayed.
- SUMMARY - Causes the field label keywords and data field labels to be displayed for each field of a record ID.

Command Example:

```
LOGTOOL> types

Type      Description
-----
100      LOG FAILURE
101      SYSTEM UP
102      JOB INITIATION
103      JOB TERMINATION
104      PROCESS TERMINATION
105      FILE CLOSE
106      SHUTDOWN
107      POWER FAILURE
108      SPOOLING
```

111 I/O ERROR
112 PHY. MOUNT/DISMOUNT
113 LOG. MOUNT/DISMOUNT
114 TAPE LABELS
115 CONSOLE LOG
116 PROGRAM FILE EVENT
143 CHGROUP
144 FILE OPEN
150 AUTO-DIAG/SUM
151 HPMC
152 LPMC
LOGTOOL>



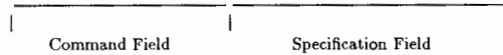
Record Definition Files

Many commands in the following sections are separated by the “” character. This is the “grave” character, which resembles a single opening quotation mark. This delimiter is important, and must be included in commands.

Each entry found in the system error log has an associated RECORD'ID number. In MPE XL, this is the first 16 bits of every log entry. Since these RECORD'ID numbers identify unique log entries, LOGTOOL will use these RECORD'ID numbers to match data from the log files with a RECORD'ID declaration. The RECORD'ID declarations specify the exact format of each log record and will be used during the translation of a log file. Therefore, the record definition file contains a series of Record'ID Declarations. To support PASCAL record variants, a second type of RECORD'ID may also appear in the file. These are referred to as TAG'IDs, since they describe a variant record tag field. Therefore, the record definition file would look as follows:

```
<record id X control statements>
<record id Y control statements>
<record id Z control statements>
```

The record definition file consists of control statements. Each statement has two fields: the “command field” that contains the record definition command that identifies the remainder of the data and the “specification field” that contains all of the parameters and arguments required by a particular control command. The specification field may span multiple lines. The terminator of a specification field is another command. Commands must be the first character of a line with no preceding white space (e.g., they must start at column 1). Comments can be embedded within the specifications field by using braces { }.



The following example should clarify these concepts:

COMMENT	The “COMMENT” is a record definition command. The body of the comment is called the specification field (that is what you are reading now). Notice that this field can span as many lines as desired. A new control command ends the specification field.
RECORD'ID	23 {This is a comment too}
COMMENT	The above line has “RECORD'ID” as the Command and “23” as the specification field.

The record definition file can be created with any editor, but must be kept as a Fixed ASCII 80-Byte Unnumbered file.

Record Definition Commands

The following section describes the various commands that can be used in a record definition file.

All commands are terminated by finding a new command (remember, all commands must start at column one). Command names may be abbreviated to the shortest unambiguous number of characters. For example, the `FIELD'FORMAT` command could be abbreviated to `F`, since no other commands begin with the letter `F`.

COMMENT

The **COMMENT** command allows one to separate record definitions and supply useful information to the maintainer of the file. Comments are ignored.

Syntax:

```
COMMENT      <anything may follow>
```

Example:

```
COMMENT      This is a comment.  
             It spans several lines.
```

RECORD'ID

This command must precede all commands which specify an attribute of a RECORD'ID. It identifies the numeric RECORD'ID that can be used to distinguish log entries in the system log file. All commands following (until the next RECORD'ID or TAG'ID) are assumed to belong to the current RECORD'ID.

Syntax:

```
RECORD'ID      <record id # allocated by MPE XL>
```

Notes:

RECORD'ID numbers must be between 0 and 255.

Examples:

```
RECORD'ID      1
RECORD'ID      123
```

TAG'ID

This command defines the variant tag portion of a previously defined RECORD'ID or TAG'ID. For example, refer to the following PASCAL record definition:

```
TYPE my_record = RECORD
    field1 : integer;
    CASE select : boolean OF
        true  : (true_case : some_other_type);
        false : (false_case : yet_another_type);
    END;
```

It has a tag field called "select" and two possible variant parts. Each of the two variant parts would need a TAG'ID entry (with additional record definition commands) to define the variant part. The specifications field indicates which RECORD'ID this TAG'ID is to be used with. For more details, see the section called "Tag Usage Example."

Syntax:

```
TAG'ID      <record id # this definition is used with>
```

Examples:

```
TAG'ID      100
```

Record Description Commands

The following set of commands are used to define the attributes of an individual log record (RECORD'ID) or tag variant of some record (TAG'ID). (From here on RECORD and TAG'IDs will not be differentiated unless required). If the LOGTOOL LAYOUT command sees any of the commands without a RECORD or TAG'ID command, it will generate an error. Additionally, only one of the possible definition commands may be provided for any one record. The following commands must be provided when defining a RECORD or TAG'ID:

```
RECORD'LENGTH  
N'FIELDS  
RECORD'LABEL  
FIELD'FORMATS
```

The RECORD'LENGTH and N'FIELDS commands must be provided before any of the other commands. For a TAG'ID definition, the VARIANT command is required.

RECORD'LENGTH

This command specifies the total length in bits for the maximum size of the current RECORD'ID. This value should not include the size of any data added by the system (i.e., the actual RECORD'ID, time stamp, etc.). This value and the number of bits specified in the FIELD'FORMAT must agree.

Syntax:

```
RECORD'LENGTH  <# of bits>
```

Notes:

The maximum number of bytes is 2028.

Example:

```
RECORD'LENGTH  220
```

N'FIELDS

This command specifies the number of fields (pieces of data) in the current RECORD'ID. Variant tags are considered as 1 field. The subfields are defined with subsequent TAG'ID commands. This value must be equal to the number of entries on the FIELD'FORMAT command.

Syntax:

```
N'FIELDS      <# of fields>
```

Example:

```
N'FIELDS      5
```

RECORD LABEL

This command assigns a label to the log record. The label is restricted to 20 characters and must be enclosed by either double or single quotes.

Syntax:

```
RECORD LABEL    "<name>"
```

Example:

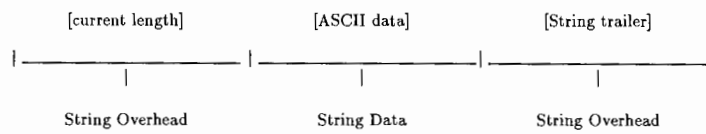
```
RECORD LABEL    "I/O"  
RECORD LABEL    ''
```

FIELD FORMAT

This command defines (bit for bit) what the record contains and the data types of each field. Several "types" are known, as follows:

<i>JUNK</i>	Typically a reserved field and is not formatted. (Don't include VALUE LABEL or VAR LABEL entries for this type).
<i>INTEGER</i>	Numeric data. It may be displayed in any of the popular bases (2,8,10,16). Currently a maximum of 64 bits is allowed.
<i>ASCII</i>	ASCII data. The full specified bit count will be displayed.
<i>STRING</i>	A PASCAL string. [Length information in the front]. Only the valid characters in the string will be printed.

For the present, assume PASCAL/MODCAL strings are of the following format:



The current length field requires 32 bits.
String trailer requires 32 bits.

<i>ARRAY</i>	An array of data. The specified number of bits (rounded up to the nearest byte count) will be displayed in HEX AND ASCII (if the character is valid) - something like the following: <pre> XX XX XX XX XX XX XX XX .hello. XX XX XX XX XX XX XX XX ..w... 4. </pre>
<i>MPEXLSTAT</i>	Specifies this field to be interpreted as an MPE XL Operating System status. LOGTOOL will display a text message from the system message catalog corresponding to the value obtained. If a message cannot be found in the catalog, the value will be displayed in the report. This type requires that the <#bits> parameter given is 32.
<i>LLIOSTAT</i>	Specifies this field is to be interpreted as an MPE XL Low Level I/O status. LOGTOOL will display a text message from the system message catalog corresponding to the value obtained. If a message cannot be found in the catalog, the value will be displayed in the report. This type requires that the <#bits> parameter given is 32.
<i>TIME</i>	Specifies this field is to be interpreted as a timestamp produced from the CLOCK intrinsic. The time obtained will be obtained in a native Language format. This type requires that the <#bits> parameter given is 32.
<i>DATE</i>	Specifies this field is to be interpreted as a datestamp produced from the CALENDAR intrinsic. The date will be displayed in a native Language format. This type requires that the <#bits> parameter given is 16.
<i>HARDSTAT</i>	This specifies that this field is to be interpreted as device hardware status. The hardware status bytes found in this field will be translated into text form. This type can only be successfully used for the I/O ERROR (111) record type. Use of this type for any other record id will produce unpredictable results.
<i>DIAGSTACK</i>	This specifies that this field is to be interpreted as a diagnostic error stack. The diagnostic error stack found in this field will be translated into text form. This type can only be successfully used for the DIAGNOSTICS (150) record type. Use of this type for any other record id will produce unpredictable results.



Syntax:

```
FIELD'FORMAT    <field list>

<field list>    ::= <repeat factor> (<field def>) |
                  <repeat factor> (<field def>) <field list>
<repeat factor> ::= an integer | <nil>
<field def>     ::= <#bits>, JUNK |
                  <#bits>, STRING |
                  <#bits>, ARRAY |
                  <#bits>, ASCII |
                  <#bits>, LLIOSTAT |
                  <#bits>, MPEXLSTAT |
                  <#bits>, TIME |
                  <#bits>, DATE |
                  <#bits>, INTEGER, <base> |
                  <#bits>, INTEGER, <base>, TAG, <#bits>

<#bits>         ::= 0 .. 16256
<base>          ::= 2 | 8 | 10 | 16
```

Notes:

- The total # of bits must match the record length specification.
- The number of fields must match the N'FIELDS specifications.
- ARRAY, ASCII, & STRING data must start on byte boundaries.
- ARRAY, ASCII, & STRING data must be multiples of 8 bits in length.
- ARRAY, ASCII, & STRING data can have a maximum of 132 bytes.
- ARRAY data will always print out in whole bytes.
- Integer data is at most 64 bits long.
- The length given for string data should include the overhead bytes value.

Examples:

```
FIELD'FORMAT    2(32,INTEGER,16) (160,ASCII)(3,INT,8)(13,JUNK)
                  (64,STRING) (64,ARRAY) (12,INTEGER,10,TAG,20)
```

- TWO (2) 32 bit integer fields printed in HEX
- 160 bits (32 chars) of printed in ascii.
- A 3 bit integer printed in octal
- 13 bits of junk which it not printed
- A 64 bit (8 char max) string - only valid chars printed.
- A 64 bit (8 bytes) array printed as HEX/ASCII
- A 12 bit integer tag field printed in decimal
- 20 additional bits [identified by the tag] follow.
- A TAG'ID must be used to describe the data.

VARIANT

This command must be provided for a TAG'ID definition. This command indicates which field of the RECORD'ID or TAG'ID the current TAG'ID definition is to be used. The RECORD'ID definition must contain a field declared as a tag. For example, a RECORD'ID must have a field similar to the following:

```
FIELD'FORMAT    (8,INT,10,TAG,64)
```

In this example, we have an 8-bit tag field (to be printed in decimal) followed by 64 bits of variant data. The VARIANT command will identify the RECORD'ID that the current TAG'ID is to be used with by a field number and value range. The VARIANT command will identify the TAG'ID containing a tag data type by a list of field number and value range pairs, which will specify the path to the current TAG'ID.

Error checking is performed to ensure that only TAG data types have variants. The value associated with a tag is not limited to a single value, but a range of values may be specified for a tag. All ranges must be unique and may not overlap with any other value or range.

Syntax:

```
VARIANT    <variant def>

<variant def>    ::= <label index> ( <variant list> ) <variant def> | <nil>
<label index>    ::= [ <field#> ]
<variant list>   ::= <value range>
<value range>    ::= <value> | <value>/<value>
<value>          ::= an integer
<field#>         ::= an integer
```

Examples:

```
VARIANT      [ 1 ] (1/100)
              [ 3 ] (-5 ) {This example is for a TAG'ID which
                           has a variant field}
```

Tag Usage Example

The system log file is expected to contain data structures as defined in PASCAL. In order to handle Pascal case variant records, the notion of TAGs has been provided. In the following example, a record definition file will be written for the PASCAL record "MY_RECORD."

```

TYPE
    some_other_type = integer;          { 32 bits }

    yet_another_type = RECORD
        count : integer;                { 32 bits }
        name   : string[4];             { (4 * 8 bits) + 64 bit overhead }
        CASE stuff : integer OF
            0 : ( f1 : integer );        { 32 bits }
            1 : ( f2 : bit8 );           { 8 bits }
        END;                            { SIZEOF = 192 bits }

YET_ANOTHER_TYPE                        MAX RECORD SIZE = x18 BYTES
COUNT                                x0.0 @ 4.0
NAME                                  x4.0 @ C.0 <- Note compiler added
                                         32 bits for string
                                         header and 32 bits
                                         for string trailer.

STUFF                                  x10.0 @ 4.0
F1                                    x14.0 @ 4.0
F2                                    x14.0 @ 1.0

my_record = RECORD
    field1 : integer;                  { 32 bits }
    CASE select : boolean OF
        true  : (true_case : some_other_type); { 32 bits }
        false : (false_case : yet_another_type); { 192 bits (max) }
    END;                                { SIZEOF = 256 bits }

MY_RECORD                              MAX RECORD SIZE = x20 BYTES
FIELD1                                x0.0 @ 4.0
SELECT                                x4.0 @ 1.0 <- Note: compiler padded
                                         to next word.

TRUE_CASE                             x8.0 @ 4.0
FALSE_CASE                             x8.0 @ 18.0

```


The following would define the above structures:

```

COMMENT          This defines MY_RECORD "

RECORD'ID        1
RECORD'LENGTH    256
RECORD'LABEL     "MY_REC"
N'FIELDS        2

FIELD'FORMAT     {field1} (32,INT,10)
                 {select} (8,INT,10,TAG,216)

VAR'LABELS       [ 1 ] "field 1 :"
                 [ 2 ] "select code : "

VALUE'LABELS     [ 2 ] (0,"False"; 1, "True" )

COMMENT          " This defines the case variant "TRUE"
                 (SOME_OTHER_TYPE) from the above record

TAG'ID           1      {This tag is to be used with RECORD'ID 1}
RECORD'LENGTH    216    {Must match length given in FIELD'FORMAT}
RECORD'LABEL
VARIANT          [ 2 ] (1)
N'FIELDS        3

FIELD'FORMAT     {compiler filler} (24,junk)
                 {some_other_type} (32,INT,10)
                 (160, junk)      {Total # bits must match RECORD'LENGTH}
VAR'LABELS       [ 1 ] "True Case"

COMMENT          " This defines the case variant "FALSE"
                 (YET_ANOTHER_TYPE) from the above record "

TAG'ID           1      {This tag is to be used with RECORD'ID 1}
RECORD'LENGTH    216    {Must match length given in FIELD'FORMAT}
RECORD'LABEL
VARIANT          [ 2 ] (0)  {Use this TAG'ID for RECORD'ID 1, when}
                   {the second field has a value of zero}
N'FIELDS        4

FIELD'FORMAT     (24,junk) (32,INT,10) (96,STRING) (32,INT,16,TAG,32)
VAR'LABELS       [ 1 ] "Count : "
                 [ 2 ] "Name : "
                 [ 3 ] "Stuff tag : "

COMMENT          " This defines the case variant "0" of
                 (YET_ANOTHER_TYPE) from the above record "

TAG'ID           1      {This tag is to be used with RECORD'ID 1}
RECORD'LENGTH    32      {Must match length given in FIELD'FORMAT}

```

```

RECORD'LABEL

VARIANT          [ 2 ] (0)
                  [ 4 ] (0)      {Use this TAG'ID when field 3 of}
                                  {previous TAG'ID has a value of zero}

N'FIELDS          1

FIELD'FORMAT      (32,INT,10)
VAR'LABELS        [ 1 ] "F1:"

COMMENT           " This defines the case variant "1" of
                  (YET_ANOTHER_TYPE) from the above record "

TAG'ID            1              {This tag is to be used with RECORD'ID 1}
RECORD'LENGTH      32           {Must match length given in FIELD'FORMAT}
RECORD'LABEL
VARIANT          [ 2 ] (0)
                  [ 4 ] (1)      {Use this TAG'ID when field 3 of}
                                  {previous TAG'ID has a value of zero}

N'FIELDS          2

FIELD'FORMAT      (8,INT,10) (24,junk)
VAR'LABELS        [ 1 ] "F2:"

```

As in the above example, TAG'IDs may be nested. Variants without tags are not supported. Consider the following:

```

TYPE
  some_record = RECORD
    case integer of
      0 : (f1 : type1);
      1 : (f2 : type2);
      2 : (f3 : type3);
    END;

```

LOGTOOL does not know which of the variants to print, since it has no idea which variant the data matches.

RECORD'ID Installation Example

This section describes in detail the process required for LOGTOOL to decode a log entry. Normally, when LOGTOOL is instructed to display the contents of a system log file, it searches through the LOGTOOL data structures for a RECORD'ID definition. If a RECORD'ID definition cannot be found, the log entry is displayed based on the definition located.

For the following example, the log entry will be that of the MPE XL low-level I/O (LLIO) system which consists of many software modules called "I/O managers." This log entry has the unique attribute that the record definition is written in parts. The log entry itself is written in two parts. A main header which displays information common to all "I/O managers" and a LLIO software dependent portion, which contains information specific to one software module.

The PASCAL declaration for the log entry header information is as follows:

```
type
  io_fixed_rec_type = record
    product_name   : dac_prodnun_type; { Name: i.e. "HP7933" }
    pdev           : pdev_str;         { Physical path i.e. "1.1.3" }
    ldev           : ldev_str;         { Logical device name }
    dev_class_code : shortint;         { 'ID for DISC, TAPE, etc }
    diag_msg_class : bit8;             { Indicates HW/SW/OTHER I/O err }
    lllo_stat      : integer;          { HPE status }

    log_all_retries : boolean;         { TRUE if retries not summarized }
    retry_again     : boolean;         { TRUE if log_all_entries will retry }
    io_worked       : boolean;         { Did I/O finally work ? }
    run_autodiag    : boolean;         { Were Auto diagnostics run? }
    retry_count     : shortint;        { N'th retry OR Total retries }
    mgr_port_num    : integer;         { port number of manager }
    tran_num        : integer;         { transaction number }
    hdwr_len        : shortint;        { # of valid bytes in the HDWR_STAT }
    hdwr_stat       : hdwr_stat_type;  { Hardware status bytes }
  end; {io_fixed_rec_type}

  io_event_array = packed array [ 1..max_buff_bytes ] of char;

  io_var_rec_type = record
    data_len : shortint; { # of valid bytes in DATA }
    mgr_code : shortint; { Uniquely IDs the mgr }
    case boolean of
      true : (data : packed array [1..max_event_count] of io_event_array);
      false : (buff : packed array [1..io_var_len-4] of char);
    end; {io_var_rec_type}
```

The previous record structure is used when logging information to the system log file. This record must be translated into a LOGTOOL record definition file which will be processed by LOGTOOL and incorporated into its data base. The following is a listing of a file named LAYLLIO, which contains the record definition for the above PASCAL record.

```

RECORD'ID      111
RECORD'LENGTH  16224
N'FIELDS      25
RECORD'LABEL   "I/O ERROR"
FIELD'FORMAT   {1 product_name}      (320,string)
                {2 pdev}              (320,string)
                {3 ldev}              (320,string)
                {4 dev_class_code}     (16,int,10)
                {5 diag_msg_class}     (8,int,10) (8,junk)
                {7 lllo_stat}          (0,int, 10 ) (32,lliostat)
                {9 log_all_retries}    (7,junk) (1,int,10)
                {11 retry_again}       (7,junk) (1,int,10)
                {13 io_worked}        (7,junk) (1,int,10)
                {15 run_autodiag}      (7,junk) (1,int,10)
                {17 retry_count}       (16,int,10)
                {18 mgr_port_num }     (16,junk) (32,int,16)
                {20 tran_num}          (32,int,16)
                {21 hdwr_len}         (16,int,10)
                {22 reserved}         (16,junk)
                {23 hdwr }             (320,hardstat)
                {24 datalen}          (16,int,10)
                {25 mgrcode}          (16,int,10,tag,14688)
                {---IO_VAR portion }

VAR'LABELS     [ 1 ] "PRODUCT NAME:"
                [ 2 ] "PDEV:"
                [ 3 ] "LDEV:"
                [ 4 ] "DEVICE CLASS:"
                [ 5 ] "I/O EVENT CLASS:"
                [ 8 ] "LLIO STATUS:"
                [ 10 ] "RETRY SCHEME:"
                [ 12 ] "WILL RETRY:"
                [ 14 ] "I/O RESULT:"
                [ 16 ] "RUN AUTODIAG:"
                [ 17 ] "RETRY COUNT:"
                [ 19 ] "MGR PORT NUM.:"
                [ 20 ] "TRANS. NUM. :"
                [ 21 ] "# HDWR BYTES:"
                [ 23 ] "HARDWARE STATUS:"
                [ 24 ] "DATA LEN:"
                [ 25 ] "MGR CODE:"

```

VALUE LABELS

```
[ 5 ] (0,"Hardware";      1,"Software";    2,"Other")
[ 4 ] (0,"Hard Disc";     1,"Mag Tape";     2,"Terminal";
      4,"Plotter";        7,"Bus Converter" 13,"Crd Rdr";
      16,"DTC";           19,"Term. Mux.";  21,"Printer";
      22,"CA";            23,"DA")
[ 10 ] (1,"Log Each Retry"; 0,"Summarized Retries")
[ 12 ] (1,"YES"; 0,"NO" )
[ 14 ] (1,"I/O Successful"; 0,"I/O Failed")
[ 16 ] (1,"YES"; 0,"NO" )
```

Now that the record definition file for the header is complete, it can be installed using LOGTOOL. The user must enter the Diagnostic User Interface (DUI) to access LOGTOOL. The following LOGTOOL session will perform this task:

:SYSDIAG

```
*****
*****                               *****
*****      ONLINE DIAGNOSTIC SYSTEM      *****
*****                               *****
*****      (C) Copyright Hewlett Packard Co. 1987 *****
*****      All Rights Reserved.          *****
*****                               *****
*****  DUI  Version A.00.00      Monitor Version A.00.00 *****
*****                               *****
*****      *****
```

Welcome, Today is MON, August 10, 1987 at 12:30 PM

Type HELP for assistance.

DUI >run logtool

```
*****
*****                               *****
*****      LOGTOOL                               *****
*****                               *****
*****      (C) Copyright Hewlett Packard Co. 1987 *****
*****      All Rights Reserved.          *****
*****      Version A.00.00                *****
*****                               *****
*****      *****
```

Welcome, Today is MON, August 10, 1987 at 12:30 PM

LOGTOOL>list log=23;type=111 {Display the log entry
without a record definition}

SYSTEM ERROR LOG REPORT =====

Report Date/Time: TUE, JAN 1, 1987 - 2:25 AM

=====

TUE, JAN 1, 1987 2:22 AM LOG0023.PUB.SYS SYSTEM (PIN 27)

RECORD TYPE 111

	1	2	3	4	5	6	7	8	9	10	11	12	
	==	==	==	==	==	==	==	==	==	==	==	==	
1:	00	00	00	07	48	50	49	42	44	41	4D	00	 H P I B D A M .
13:	00	00	00	00	00	00	00	00	00	00	00	00	
25:	00	00	00	00	00	00	00	00	00	00	00	00	
37:	00	00	00	00	00	00	00	03	38	2E	32	00	 8 . 2 .
49:	00	00	00	00	00	00	00	00	00	00	00	00	
61:	00	00	00	00	00	00	00	00	00	00	00	00	
73:	00	00	00	00	00	00	00	00	00	00	00	00	
85:	00	00	00	00	00	00	00	00	00	00	00	00	
97:	00	00	00	00	00	00	00	00	00	00	00	00	
109:	00	00	00	00	00	00	00	00	00	00	00	00	
121:	00	06	00	00	5A	00	00	74	00	00	00	00	 Z . . t
133:	00	00	00	00	FF	FF	FF	AE	00	00	00	00	
145:	00	01	00	00	01	02	00	00	00	00	00	00	
157:	00	00	00	00	00	00	00	00	00	00	00	00	
169:	00	00	00	00	00	00	00	00	00	00	00	00	
181:	00	00	00	00	00	00	00	00	00	18	00	74	 t
193:	02	00	00	00	00	00	00	0C	00	45	45	45	 E E E
205:	00	00	00	0C	00	67	46	94	00	00	00	09	 g F

DEVICE HARDWARE STATUS:

	1	2	3	4	5	6	7	8	9	10	11	12
	==	==	==	==	==	==	==	==	==	==	==	==
1:	FF											

TYPE	DESCRIPTION	COUNT
=====	=====	=====
111		1
=====	=====	=====

LOGTOOL>layout file=layllio; append {Update data base file}
Processing ID 111 ...

End processing of layout file LAYLLIO.DIAG.SYS
0 Errors encountered
1 out of 1 record ID's were accepted.
0 out of 0 tag ID's were accepted.

Number of messages in LAYMSG = 287
Header records in LAYOUTF = 25

Total record in LAYOUTF = 799

LOGTOOL> list log=23; type=111 {Display log entry with header}

SYSTEM ERROR LOG REPORT
=====

Report Date/Time: TUE, JAN 1, 1987 - 2:26 AM

=====

TUE, JAN 1, 1987	2:22 AM	LOG0023.PUB.SYS	SYSTEM (PIN 27)
------------------	---------	-----------------	-----------------

I/O ERROR

PRODUCT NAME:	HPIBDAM	PDEV:	8.2
LDEV:		DEVICE CLASS:	6
I/O EVENT CLASS:	Hardware	LLIO STATUS:	\$5A000074

MPE/XL I/O Status: Proc. Num. = 0, Error Num. = 90, Subsystem = 116

RETRY SCHEME:	Summarized Retries	WILL RETRY:	NO
I/O RESULT:	I/O Failed	RUN AUTODIAG:	NO
RETRY COUNT:	0	MGR PORT NUM.:	-82
TRANS. NUM. :	0	# HDWR BYTES:	1

HARDWARE STATUS:

1	2	3	4	5	6	7	8	9	10	11	12
==	==	==	==	==	==	==	==	==	==	==	==

1: FF

DATA LEN: 24 MGR CODE: 116

TAG DEFINITION NOT GIVEN - FIELD WILL BE DISPLAYED IN HEX:

1	2	3	4	5	6	7	8	9	10	11	12
==	==	==	==	==	==	==	==	==	==	==	==

1: 02 00 00 00 00 00 00 0C 00 45 45 45.... . E E E

13: 00 00 00 0C 00 67 46 94 00 00 00 09.... . g F.... .

=====

TYPE	DESCRIPTION	COUNT
=====	=====	=====
111	I/O ERROR	1
=====	=====	=====

LOGTOOL>

Because the record definition file only contains declarations for the header, the "I/O manager" dependent portion of the log entry is displayed as a hex dump. The "I/O manager" dependent portion of the record definition will be given in the format of TAG'ID declarations. As an example, the following PASCAL record defines an "I/O manager" dependent portion of the log entry:

```

TYPE
  dam_log_info_type = PACKED RECORD
    hw_status      : bit8;
    dam_mgr_ptr    : srv_ptr_type;
    dam_msg_ptr    : msg_ptr_type;
    dam_uid_num    : integer;
  END;

```



DAM_LOG_INFO_TYPE	MAX_RECORD_SIZE = x18 BYTES
HW_TSTATUS	x0.0 @ 1.0
DAM_MGR_PTR	x4.0 @ 8.0
DAM_MSG_PTR	xC.0 @ 8.0
DAM_UID_NUM	x14.0 @ 4.0

The TAG'ID definition for the above declaration is the following, which will be stored in the file LAYPORT:

```

TAG'ID 111                                {HP-IB device adapter manager}
RECORD'LENGTH 192
N'FIELD 7
RECORD'LABEL
VARIANT [25] (116)
FIELD'FORMAT (8,int,16)                   {Hardware status}
              (24,junk)                   {Compiler filler}
              (32,int,16)                 {Port data area address}
              (32,int,16)
              (32,int,16)                 {Message pointer}
              (32,int,16)
              (32,int,10)                 {UID}

VAR'LABEL [ 1 ] "HDWR STATUS:"
          [ 3 ] "PDA SPACE'ID:"
          [ 4 ] "PDA OFFSET:"
          [ 5 ] "MSG SPACE'ID:"
          [ 6 ] "MSG OFFSET:"
          [ 7 ] "UID:"

```


Now, the TAG'ID can be installed through LOGTOOL:

```
:SYSDIAG

*****
****
****          ONLINE DIAGNOSTIC SYSTEM          ****
****
****          (C) Copyright Hewlett Packard Co. 1987 ****
****          All Rights Reserved.              ****
****
**** DUI Version A.00.00      Monitor Version A.00.00 ****
****
*****

Welcome, Today is MON, August 10, 1987 at 12:30 PM

Type "HELP" for assistance.

DUI >run logtool

*****
****
****          LOGTOOL                          ****
****
****          (C) Copyright Hewlett Packard Co.  ****
****          All Rights Reserved.              ****
****          Version A.00.00                   ****
****
*****

Welcome, Today is MON, August 10, 1987 at 12:30 PM

LOGTOOL>layout file=layport;append
      Processing TAG'ID 111 ...
=====

End processing of layout file LAYPORT.DIAG.SYS

0 Errors encountered
0 out of 0 record'IDs were accepted.
1 out of 1 tag'IDs were accepted.

Number of messages in LAYMSG = 292
Header records in LAYOUTF=26
Total records in LAYOUTF = 819

LOGTOOL>list log=14;type=111      {Display log entry}
```

SYSTEM ERROR LOG REPORT
=====

Report Date/Time: TUE, JAN 1, 1987 - 2:27 AM

```
=====
TUE, JAN 1, 1987  2:22 AM      LOG0023.PUB.SYS          SYSTEM (PIN 27)
I/O ERROR
PRODUCT NAME:      HPIBDAM          PDEV:              8.2
LDEV:              DEVICE CLASS:    6
I/O EVENT CLASS:   Hardware         LLIO STATUS:      $5A000074

MPE/XL I/O Status: Proc. Num. = 0, Error Num. = 90, Subsystem = 116

RETRY SCHEME:      Summarized Retries WILL RETRY:      NO
I/O RESULT:        I/O Failed      RUN AUTODIAG:      NO
RETRY COUNT:       0                MGR PORT NUM.:    -82
TRANS. NUM. :      0                # HDWR BYTES:      1
HARDWARE STATUS:
  1  2  3  4  5  6  7  8  9  10 11 12
  == == == == == == == == == == == ==
  1: FF

DATA LEN:          24                MGR CODE:         116
HDWR STATUS:       $2                PDA SPACE ID:      $C
PDA OFFSET:        $454545          MSG SPACE ID:      $C
MSG OFFSET:        $674694          UID:              9
=====
```

TYPE	DESCRIPTION	COUNT
111	I/O ERROR	1

LOGTOOL>

LOGTOOL is not able to display all the information for the log entry. Since the append option was used in the LAYOUT command, all subsequent users have access to the definition given. To shorten this process, the LAYLLIO and LAYPORT record definition files can be merged together. Then LOGTOOL will process the RECORD'ID, followed by the TAG'ID.

Error and Warning Messages

The following are general error messages which may be encountered during the execution of LOGTOOL. Please note that system dependent error messages may be displayed from the On-Line diagnostic system before any error message generated by this diagnostic. All error messages without the (LTERR #) trailer are generated by the online diagnostics subsystem. Each of the error messages is followed by CAUSE/ACTION explanations. These explanations will not be displayed to the user.

401	*** PASCAL CASE STATEMENT OVERFLOW IN ! (LTERR 401)
CAUSE	LOGTOOL has hit the otherwise clause of an CASE statement. This is a software error.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of LOGTOOL commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the LOGTOOL being executed.
404	*** UNABLE TO OBTAIN MESSAGE ! FROM DIAG. CATALOG (LTERR 404)
CAUSE	LOGTOOL was not able to obtain a text message from LOGTOOL's message set in the diagnostic message catalog.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of LOGTOOL commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the LOGTOOL being executed.
405	*** ERROR ATTEMPTING TO DELETE FILE !. (LTERR 405)
CAUSE	LOGTOOL encountered an error while attempting to purge a file. The file name will be displayed in this error message.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
406	*** ERROR ATTEMPTING TO OPEN FILE !. (LTERR 406)
CAUSE	LOGTOOL encountered an error while attempting to open a file. The file name will be displayed in this error message.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
407	*** ERROR ATTEMPTING TO CLOSE FILE !. (LTERR 407)
CAUSE	LOGTOOL encountered an error while attempting to close a file. The file name will be displayed in this error message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.



408	*** ERROR ATTEMPTING TO READ FROM FILE !. (LTERR 408)
CAUSE	LOGTOOL encountered an error while attempting to read from a file. The file name will be displayed in this error message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
409	*** ERROR ATTEMPTING TO WRITE TO FILE !. (LTERR 409)
CAUSE	LOGTOOL encountered an error while attempting to write to a file. The file name will be displayed in this error message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
410	*** ERROR ATTEMPTING TO CREATE FILE !. (LTERR 410)
CAUSE	LOGTOOL encountered an error while attempting to create a file. The file name will be displayed in this error message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
414	*** FILE "LAYOUTF" CORRUPTED. ID !. EXPECTED !. REC# !. (LTERR 414)
CAUSE	LOGTOOL discovered some inconsistencies while it was reading its data base file LAYOUTF.DIAG.SYS.
ACTION	Contact support personnel to get a new LAYOUTF.DIAG.SYS and LAYMSG.DIAG.SYS files.
417	*** COULD NOT DECODE STATUS FOR ! DEVICE (LTERR 417)
CAUSE	LOGTOOL issued a request to the diagnostic system to have the appropriate diagnostic program initiated for decode of hardware status. That request was not granted. Therefore, the hardware status is displayed as a HEX dump.
ACTION	Messages indicating why the request was not granted will be displayed with this error message. Please refer to the ACTION clause for those messages for further information.
418	*** COULD NOT KILL ! AFTER STATUS DECODE FINISHED (LTERR 418)
CAUSE	LOGTOOL was not able to abort a diagnostic program which was initiated for decoding of hardware status.
ACTION	Messages indicating why the diagnostic program could not be aborted will be displayed with this error message. Please refer to the ACTION clause for those messages for further information.
421	*** ERROR ATTEMPTING TO READ RECORD ! FROM "!" (LTERR 421)
CAUSE	LOGTOOL encountered an error while attempting to read from a file. The file name will be displayed in this error message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.

429	*** SWITCHLOG COMMAND WAS NOT PERFORMED (LTERR 429)
CAUSE	LOGTOOL requested that a switchlog command be performed by the operating system. That request returned an error.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages.
430	*** UNABLE TO DETERMINE CURRENT LOG FILE. (LTERR 430)
CAUSE	LOGTOOL requested that a showlog command be performed by the operating system. That request returned an error.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages.
446	*** UNABLE TO OBTAIN PDEV FOR LDEV !. (LTERR 446)
CAUSE	LOGTOOL was not able to convert a physical device specifier into a logical device number.
ACTION	Error messages giving further detail as to the type of problem encountered will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
452	*** WARNING: SYSTEM LOGGING IS SUSPENDED (LTWARN 452)
CAUSE	The MPE XL operating system indicates that the system logging facility has been suspended. Therefore, the SWITCHLOG request was not performed.
ACTION	Resume the system logging facility.
453	*** WARNING: SYSTEM LOGGING IS DISABLED (LTWARN 453)
CAUSE	The MPE XL operating system indicates that the system logging facility has been disabled. Therefore, the SWITCHLOG request was not performed.
ACTION	Start the system logging facility.
455	*** UNABLE TO COMMUNICATE WITH MEMORY LOGGING PROCESS (LTERR 455)
CAUSE	The user has issued either a MEMTIMER, MEMCLR, or MEMRPT command to LOGTOOL. To perform these commands, LOGTOOL must communicate with the memory logging process. LOGTOOL is not able to locate MEMLOGP.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
456	*** UNABLE TO OBTAIN MEMORY REPORT DATA (LTERR 456)
CAUSE	The user has issued a MEMRPT command to LOGTOOL. LOGTOOL was not able to successfully obtain memory report information from the memory logging process (MEMLOGP).
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.

457	*** UNABLE TO START DISPLAYLOG COMMAND (LTERR 457)
CAUSE	LOGTOOL was not able to notify the diagnostic system that DISPLAYLOG was to start.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
458	*** UNABLE TO STOP DISPLAYLOG COMMAND (LTERR 458)
CAUSE	LOGTOOL was not able to notify the diagnostic system that DISPLAYLOG was to stop.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the LOGTOOL command again.
459	*** LOGTOOL ABORTING DUE TO AN UNRECOVERABLE ERROR. FURTHER ERROR INFORMATION HAS BEEN LOGGED IN THE SYSTEM LOG FILE. (LTERR 459)
CAUSE	LOGTOOL encountered an error it could not recover from. Further error information has been logged into the system log file as a diagnostic log record. Depending on the error, this message may not make it to the users terminal, but LOGTOOL is try to tell the user that it is aborting.
ACTION	Please find the diagnostic log record containing the detailed error information on why LOGTOOL aborted. In most cases, a service request may need to be submitted.
460	*** LOGTOOL ERROR NUMBER IS !. (LTERR 460)
CAUSE	LOGTOOL encountered an error it could not recover from. The number given will identify the type of error encountered.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the LOGTOOL command again.
461	*** LOGTOOL ABORTED DUE TO AN ERROR IN PROCEDURE ! (LTERR 461)
CAUSE	LOGTOOL encountered an error it could not recover from. The procedure name given will identify the location in LOGTOOL code where the error occurred.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of LOGTOOL commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the LOGTOOL being executed.

462	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO CONVERT A LOG NUMBER TO A PASCAL STRING (LTERR 462)
CAUSE	After obtaining a log number from the user, LOGTOOL needs to convert that number from an integer to a PASCAL string. This string is later used to form a log file name.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
463	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO FULLY QUALIFY A LOG FILE NAME (LTERR 463)
CAUSE	After LOGTOOL has formed a log file name from the log file number given, it must fully qualify the file name to access the file. An unexpected error occurred while trying to fully qualify the log file name with the directory where log files are stored.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
464	*** UNABLE TO START PROGRAM TRACE (LTERR 464)
CAUSE	LOGTOOL was not able to notify the diagnostic system that tracing of program procedures is to begin.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
465	*** UNABLE TO START PROGRAM AND LIBRARY TRACE (LTERR 465)
CAUSE	LOGTOOL was not able to notify the diagnostic system that tracing of program and diagnostic library procedures is to begin.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
466	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO WRITE THE LOGTOOL FILE HEADER TO THE WORK FILE (LTERR 466)
CAUSE	The first record of each work file create by LOGTOOL contains a unique header. LOGTOOL uses this header to identify work files from other disk files found in the diagnostic directory. In this situation, LOGTOOL encountered an unexpected error while trying to write the header information to the work file.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.

467	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO APPEND FILE PATH TO LOG FILE NAME (LTERR 467)
CAUSE	After LOGTOOL has formed a log file name from the log file number given, it must fully qualify the file name to access the file. An unexpected error occurred while trying to fully qualify the log file name with the directory where log files are stored.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
468	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO OBTAIN LIST OF LOG FILES (LTERR 468)
CAUSE	The user entered the STATUS command in LOGTOOL. LOGTOOL encountered an unexpected error while trying to obtain the list of log files available on the system.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
470	*** UNABLE TO OBTAIN THE SIZE OF A LOG FILE (LTERR 470)
CAUSE	LOGTOOL encountered an unexpected error while trying to determine how many records were in a log file.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
471	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO ABORT A MEMORY REPORT (LTERR 471)
CAUSE	The user entered a user interrupt while a memory report was being generated. In response to the user interrupt, LOGTOOL tried to abort the memory report and encountered an error.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
472	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO DETERMINE IF A USER INTERRUPT WAS ENTERED (LTERR 472)
CAUSE	An unexpected error status was returned from a diagnostic procedure which determines if a user interrupt was entered.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
473	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO DECODE A SECOND HARDWARE STATUS (LTERR 473)
CAUSE	LOGTOOL encountered an unexpected error while trying to request that a diagnostic program decode a second hardware status.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.

474	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO OBTAIN HARDWARE STATUS TEXT (LTERR 474)
CAUSE	LOGTOOL issued a request to the diagnostic system that a diagnostic program be launched to decode hardware status. LOGTOOL encountered an unexpected error while trying to obtain the text from the diagnostic program.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
475	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO SEND DATA TO THE SYSTEM PRINTER (LTERR 475)
CAUSE	LOGTOOL encountered an unexpected error while trying to send one line of a report to the system line printer.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
476	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO END THE PRINT JOB (LTERR 476)
CAUSE	LOGTOOL encountered an unexpected error while trying to end the print job, which contained either a log report or a memory report.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
477	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO AN OBTAIN THE NEXT ENTRY FROM THE DIAGNOSTIC ERROR STACK (LTERR 477)
CAUSE	While translating a diagnostic record from the system log file, LOGTOOL was not able to obtain an entry from the logged diagnostic error stack.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
478	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO TRANSLATE AN ENTRY FROM THE DIAGNOSTIC ERROR STACK TO TEXT (LTERR 478)
CAUSE	While translating a diagnostic record from the system log file, LOGTOOL was not able to obtain text for the entry obtained from the logged diagnostic error stack.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
479	*** UNEXPECTED ERROR OCCURRED WHILE TRYING TO FULLY QUALIFY A DIAGNOSTIC FILE NAME (LTERR 479)
CAUSE	LOGTOOL is required to fully qualify the names of all files it needs to access in the diagnostic directory. In this situation, it was not able to successfully perform that task.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.

502	*** INVALID LOG RECORD TYPE ! RECORD SKIPPED (LTWARN 502)
CAUSE	LOGTOOL encountered a log record with an invalid record type. This log record will be skipped and will not appear in the report.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of LOGTOOL commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the LOGTOOL being executed.
503	*** USER INTERRUPTS (CONTROL-Y) COULD NOT BE ENABLED. (LTWARN 503)
CAUSE	LOGTOOL will not able to enable CONTROL-Y during initialization. LOGTOOL will execute but CONTROL-Y will be ignored.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information.
506	*** LAYOUT FILE EMPTY OR HAS FEWER RECORDS THAN HEADER SAYS. (LTWARN 506)
CAUSE	LOGTOOL has encountered an unexpected end of file while reading its data base file LAYOUTF.DIAG.SYS.
ACTION	Contact support personnel to get a new LAYOUTF.DIAG.SYS and LAYMSG.DIAG.SYS files.
507	*** NO LAYOUT ENTRY FOR LOG TYPE ! WITH VALUE ! (LTWARN 507)
CAUSE	During the generation of a log report, LOGTOOL was not able to locate a "TAG" definition for the log record type indicated.
ACTION	Since LOGTOOL cannot find the definition, the data will be displayed as a HEX dump, and the report will continue.
508	*** NO VALUE LABEL FOR LOG TYPE ! [VALUE= !] (LTWARN 508)
CAUSE	During the generation of a log report, LOGTOOL was not able to locate a "TAG" definition for the log record type indicated.
ACTION	Since LOGTOOL cannot find the definition, the data will be displayed as a HEX dump, and the report will continue.
510	*** STRING IN A LOG TYPE ! WITH LENGTH ! IS LARGER THAN MAX SIZE IN LAYOUT (SKIPPING DECODE OF THIS FIELD) . (LTWARN 510)
CAUSE	During the generation of a log report, LOGTOOL encountered a string to be displayed which was larger than the buffer allocated within LOGTOOL.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameters. b) Any other error messages displayed with this error message. c) The version (A.nn.nn) of the LOGTOOL being executed. A copy of the log file containing the log record producing this error should also be submitted with the service request.

512	*** CONSOLE TEXT LENGTH OF ! IS INVALID. TEXT NOT DISPLAYED (LTWARN 512)
CAUSE	LOGTOOL was not able to display the console logging record because the text was larger than the buffer allocated within LOGTOOL.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameters. b) Any other error messages displayed with this error message. c) The version (A.nn.nn) of the LOGTOOL being executed. A copy of the log file containing the log record producing this error should also be submitted with the service request.
601	*** ILLEGAL INPUT - UNEXPECTED CHARACTER(S) OR NON UNIQUE TOKEN (LTERR 601)
CAUSE	LOGTOOL has encountered a problem while parsing the input text given by the user. The problem can be any of the following: o Unknown token or a token which did not have enough characters specified to make it unique. o Extra characters on a line. o User entered non-printable characters.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
602	*** VALUE INDICATING NUMBER OF SECONDS EXPECTED (LTERR 602)
CAUSE	The user did not specify a mandatory parameter which is the number of seconds to be used.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
603	*** THIS OPTION HAS ALREADY BEEN SPECIFIED OR CONFLICTS WITH ANOTHER (LTERR 603)
CAUSE	The user has specified an option more than once. This includes the situation where one of two types of parameters are needed but not both.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
604	*** OPTION INVALID FOR THIS COMMAND (LTERR 604)
CAUSE	The option given by the user is not appropriate for the command.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
605	*** NO SUCH TOPIC. TYPE HELP FOR LIST OF COMMANDS AND TOPICS (LTERR 605)
CAUSE	The command given by the user does not exist within LOGTOOL.
ACTION	Entering HELP alone will cause LOGTOOL to display a list of all commands within LOGTOOL.

606	*** KEYWORD PARAMETER IS INVALID FOR THIS COMMAND (LTERR 606)
CAUSE	The keyword parameter given by the user is not appropriate for the command.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
607	*** OUTPUT FILE NAME DESIGNATOR EXPECTED (LTERR 607)
CAUSE	The user has given the OUTFILE keyword, but did not specify a file name, or gave an invalid file name.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
609	*** FORMAT OF THE DATE IS INVALID (LTERR 609)
CAUSE	The date given by the user was not in the expected format.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
610	*** FILE IDENTIFIER EXPECTED (LTERR 610)
CAUSE	User has given the keyword FILE, but did not give a file name, or gave an incorrectly formatted file name.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
612	*** LDEV IDENTIFIER EXPECTED (LTERR 612)
CAUSE	User has given the keyword LDEV, but did not give a logical device number.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
613	*** LOG FILE NUMBERS MUST BE BETWEEN 0 AND 9999 (LTERR 613)
CAUSE	User has given the keyword LOG, but gave a log file number which was not between 0 and 9999.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
614	*** LOG FILE NUMBER EXPECTED (LTERR 614)
CAUSE	User has given the keyword LOG, but did not give a log number.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
616	*** PDEV IDENTIFIER EXPECTED (LTERR 616)
CAUSE	User has given the keyword PDEV, but did not give physical device specifier.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.

617	*** INVALID RECORD ID (TYPE) (LTERR 617)
CAUSE	User specified a type number which was outside the valid range of 0 to 255.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
618	*** LOG RECORD TYPE IDENTIFIER EXPECTED (LTERR 618)
CAUSE	The user specified the keyword TYPE, but did not give a type number.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
619	*** THIS KEYWORD HAS ALREADY BEEN SPECIFIED (LTERR 619)
CAUSE	The keyword parameter specified by the user has already been found in the input text.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
620	*** INVALID RANGE - ENDING VALUE IS LESS THAN STARTING VALUE (LTERR 620)
CAUSE	User has specified a range of values where the ending value is less than the starting value.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
622	*** VALUE MUST BE BETWEEN ! AND ! (LTERR 622)
CAUSE	The user specified a numeric parameter which was not between the values indicated in the error message.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
623	*** LDEV NUMBER ALREADY GIVEN (LTERR 623)
CAUSE	In the LDEV= parameter, the user has specified a logical device number more than once. LOGTOOL will display a carat indicating the duplicate data.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
624	*** PDEV ALREADY GIVEN (LTERR 624)
CAUSE	In the PDEV= parameter, the user has specified a physical device specifier more than once. LOGTOOL will display a carat indicating the duplicate data.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.

626	*** WORK FILE ALREADY GIVEN (LTERR 626)
CAUSE	In the FILE= parameter, the user has specified a work file name more than once. LOGTOOL will display a carat indicating the duplicate data.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
628	*** TYPE ID ALREADY GIVEN (LTERR 628)
CAUSE	In the TYPE= parameter, the user has specified a type number more than once. LOGTOOL will display a carat indicating the duplicate data.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
629	*** LOG NUMBER OR LOG RANGE ALREADY GIVEN OR OVERLAP (LTERR 629)
CAUSE	In the LOG= parameter, the user has specified a log number more than once. LOGTOOL will display a carat indicating the duplicate data.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
630	*** FORMAT OF THE TIME IS INVALID (LTERR 630)
CAUSE	The user gave the ;TIME option on the input text. When the user was prompted for the time, it was entered in an incorrect format.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
632	*** CLOSE BRACKET EXPECTED (LTERR 632)
CAUSE	The user did not give a close bracket character when giving an index with a field label keyword.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
633	*** INDEX FOR FIELD LABEL KEYWORDS MUST BE GREATER THAN ZERO (LTERR 633)
CAUSE	The index given in the field label keyword was less than one, which is illegal.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
634	*** INDEX FOR FIELD LABEL KEYWORD EXPECTED (LTERR 634)
CAUSE	The user did not given a value to be used as the index for a field label keyword.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.

635	*** FIELD LABEL GIVEN IN EXCEEDS MAXIMUM LENGTH OF 20 CHARACTERS (LTERR 635)
CAUSE	The user give a field label which exceeded 20 character.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
636	*** DATA ITEM ALREADY GIVEN. (LTERR 636)
CAUSE	The user gave the same data item in a field label keyword parameter twice.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
637	*** DATA ITEM FOR FIELD LABEL KEYWORD EXPECTED. (LTERR 637)
CAUSE	The user did not give a data item for the field label keyword.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
638	*** INVALID TO GIVE BOTH 'DETAIL' and 'SUMMARY' OPTIONS (LTERR 638)
CAUSE	The user entered both the detail and summary options in the TYPES command, which is illegal.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the TYPES command.
639	*** THIS KEYWORD PARAMETER HAS ALREADY BEEN GIVEN OR CONFLICTS WITH ANOTHER (LTERR 603)
CAUSE	The user has specified an keyword parameter more than once, or specified a second keyword which is not allowed with a keyword parameter already given.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
640	*** THE FIELD LABEL RANGE HAS ALREADY BEEN GIVEN (LTERR 640)
CAUSE	The user has entered the same field label range twice.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
641	*** EXPECTING "ALL" RESERVED WORD (LTERR 641)
CAUSE	A reserved word other than "ALL" was found. This is invalid for the command entered.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.

642	*** FIELD LABEL RANGE EXPECTED (LTERR 642)
CAUSE	The user entered the DISABLE= or ENABLE= keyword, but the parameters are not valid field label ranges.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
643	*** INVALID FIELD LABEL GIVEN (LTERR 643)
CAUSE	The user entered an invalid field label. Field labels must be in the format of "text" or text, where text cannot be a empty string or a number.
ACTION	Please consult the LOGTOOL help facility for the correct syntax of the desired command.
651	*** REQUIRED "LOG=<LOG RANGE LIST>" PARAMETER WAS NOT FOUND (LTERR 651)
CAUSE	The user did not give a mandatory parameter of LOG= in the input text.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
652	*** REQUIRED "FILES=<WORK FILE LIST>" PARAMETER WAS NOT FOUND (LTERR 652)
CAUSE	The user did not give a mandatory parameter of FILES= in the input text.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
653	*** REQUIRED "OUTFILE=<WORK FILE NAME>" PARAMETER WAS NOT FOUND (LTERR 653)
CAUSE	The user did not give a mandatory parameter of OUTFILE= in the input text.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
655	*** THE LINE PRINTER IS AN INVALID OUTPUT FILE FOR THIS COMMAND (LTERR 655)
CAUSE	The user has specified the line printer with the OUTFILE= parameter. But the product of the command is not a printable file.
ACTION	Please consult the LOGTOOL help facility for further information
656	*** COMMAND NAME EXPECTED. (LTERR 656)
CAUSE	The first token given on the input text was not a command.
ACTION	Please consult the LOGTOOL help facility for a list of commands available within LOGTOOL.

657	*** REQUIRED "SECONDS=<#SECONDS>" PARAMETER WAS NOT FOUND (LTERR 657)
CAUSE	The command given by the user requires a SECONDS= parameter. This parameter indicates the number of seconds that the operation should be performed, or the number of seconds between operations.
ACTION	Please consult the LOGTOOL help facility for further information.
658	*** REQUIRED "TYPE=<TYPE LIST>" PARAMETER WAS NOT FOUND (LTERR 658)
CAUSE	The user did not give a mandatory parameter of TYPE= in the input text.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
659	*** REQUIRED "ENABLE=<FIELD LABEL LIST>" OR "DISABLE=<FIELD LABEL LIST>" PARAMETER WAS NOT FOUND (LTERR 659)
CAUSE	The user did not give a mandatory parameter of ENABLE= or DISABLE= in the input text.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
702	*** INSUFFICIENT CAPABILITIES TO PERFORM THIS COMMAND (LTERR 702)
CAUSE	The user does not have a high enough diagnostic security level to execute the requested command.
ACTION	Please consult the diagnostic user interface manual for detail concerning diagnostic security levels.
713	*** FILE ! HAS NO MORE ROOM FOR DATA (LTERR 713)
CAUSE	The file specified in the OUTFILE= parameter has been filled to capacity.
ACTION	Select a smaller set of data to be placed in the outfile.
720	*** THE LINE PRINTER MAY NOT BE ACCESSED IN APPEND MODE (LTERR 720)
CAUSE	The user has selected the system printer as the outfile, and also gave the APPEND option. This is not allowed.
ACTION	Please consult the LOGTOOL help facility for further information on the command.
731	*** UNABLE TO FIND FIELD LABEL "!"
	*** ALTFIELD FOR "!"/"!" WAS NOT PERFORMED (LTERR 731)
CAUSE	LOGTOOL was not able to locate the field label give in the format for the record type given in the TYPE= parameter. Therefore, the ALTFIELD command could not be performed.
ACTION	Please consult the TYPES SUMMARY command to verify that the field label is appropriate for the desired record type.



732	*** UNABLE TO FIND FIELD LABEL "!". ALTFIELD WAS NOT PERFORMED (LTERR 732)
CAUSE	LOGTOOL was not able to locate the field label give in the format for the record type given in the TYPE= parameter. Therefore, the ALTFIELD command could not be performed.
ACTION	Please consult the TYPES SUMMARY command to verify that the field label is appropriate for the desired record type.
900	*** THIS ID HAS ALREADY BEEN DEFINED (LTERR 900)
CAUSE	LOGTOOL has already encountered a RECORD'ID command for the current record definition.
ACTION	Please consult the LOGTOOL manual for the correct usage of the RECORD'ID command.
903	*** ONLY 16 BIT VALUES ARE VALID (LTERR 903)
CAUSE	The record id number given is outside the range of a 16 bit value.
ACTION	Please consult the LOGTOOL manual for the correct usage of the RECORD'ID command.
904	*** CLOSING PARENTHESIS EXPECTED (LTERR 904)
CAUSE	User did not give the closing parenthesis. The carat will indicate that expected location of the parenthesis.
ACTION	Please consult the LOGTOOL manual for the correct format of the command.
905	*** CLOSING BRACKET EXPECTED (LTERR 905)
CAUSE	User did not give the closing bracket. The carat will indicate that expected location of the bracket.
ACTION	Please consult the LOGTOOL manual for the correct format of the command.
906	*** INDEX IS GREATER THAN NUMBER OF FIELDS (LTERR 906)
CAUSE	The field index given by the user is greater than the number of fields given in the FIELD'FORMAT command of the current record definition.
ACTION	Please consult the LOGTOOL manual for the correct format of the VAR'LABEL command.
907	*** INDEX FOR FIELD NUMBER EXPECTED (LTERR 907)
CAUSE	The expected field number was not found. LOGTOOL will display a carat indicate where it was expecting to find the field number.
ACTION	Please consult the LOGTOOL manual for the correct format of the VAR'LABEL or VARIANT command.

908		*** OPENING BRACKET EXPECTED (LTERR 908)
CAUSE	LOGTOOL is encountered a token other than a open bracket. LOGTOOL will display a carat to indicate where the bracket was expected.	
ACTION	Please consult the LOGTOOL manual for further information on the VAR'LABEL or VARIANT command.	
909		*** ENDING RANGE VALUE EXPECTED HERE (LTERR 909)
CAUSE	LOGTOOL encountered a value range structure, but the ending value was not in the correct format. LOGTOOL will display a carat indicating where it expecting the ending range value to be.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL command.	
910		*** STARTING VALUE MUST BE LESS THAN ENDING VALUE (LTERR 910)
CAUSE	LOGTOOL encountered a value range specifier, but the ending value was greater than the starting value. This is incorrect.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL command.	
911		*** STARTING RANGE VALUE EXPECTED HERE (LTERR 911)
CAUSE	LOGTOOL was expecting the next token to be a value range specifier. LOGTOOL will display a carat to indicate where it was expecting the string range value to be.	
ACTION	Please consult the LOGTOOL manual for further information. on the VALUE'LABEL command.	
912		*** RECORD OR TAG ID IS OUT OF RANGE (LTERR 912)
CAUSE	LOGTOOL encountered a record id number which was outside the range of 0 to 255.	
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'ID or TAG'ID command.	
913		*** RECORD OR TAG ID IS EXPECTED HERE (LTERR 913)
CAUSE	LOGTOOL is expecting a record id number if the command is RECORD'ID, or a tag id number if the command is TAG'ID. This error message will be displayed is the if number is not present or is in an incorrect format.	
ACTION	Please consult the LOGTOOL manual for further information on the appropriate command.	
914		*** RECORD OR TAG ID STATEMENT NOT ALLOWED HERE. COMMAND IGNORED (LTERR 914)
CAUSE	The user has already specified a RECORD'ID or TAG'ID command for the current record definition. The RECORD'ID or TAG'ID command encountered will be ignored.	
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'ID or TAG'ID command.	

915	*** UNRECOGNIZED TOKEN IN COMMAND FIELD (LTERR 915)
CAUSE	LOGTOOL was expecting a command name. LOGTOOL will display a carat indicating where it was expecting the command name.
ACTION	Please consult the LOGTOOL manual for further information on record definition files.
916	*** NUMBER OF BITS TOO LARGE FOR THIS TYPE OF FIELD (LTERR 916)
CAUSE	The number of bits specified in the field format is too large for the type of field.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
917	*** ARRAY DATA MUST START ON A 8 BIT BYTE BOUNDARY (LTERR 917)
CAUSE	LOGTOOL requires array data to start on a 8 bit byte boundary.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
918	*** ARRAY DATA MUST BE A MULTIPLE OF 8 BITS (LTERR 918)
CAUSE	LOGTOOL requires that array data be defined in multiples on 8 bits.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
919	*** ASCII DATA MUST START ON AN 8 BIT BOUNDARY (LTERR 919)
CAUSE	LOGTOOL requires that all ascii data start on a 8 bit byte boundary.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
920	*** ASCII DATA MUST BE A MULTIPLE OF 8 BITS (LTERR 920)
CAUSE	LOGTOOL required that all ascii data be defined in multiples of 8 bits.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
921	*** TAB FIELD TOO LARGE FOR SPECIFIED RECORD LENGTH (LTERR 921)
CAUSE	The tab field given along with the other field defined will exceed the bit count given in the RECORD'LENGTH command of the current record definition.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.

922	*** TAG LENGTH IN BITS EXPECTED (LTERR 922)
CAUSE	LOGTOOL encountered a token other than a value which is to be used as the tag length. LOGTOOL will display a carat where it expected the tag length value.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
923	*** COMMA EXPECTED HERE (LTERR 923)
CAUSE	LOGTOOL encountered a token other than a comma. LOGTOOL will display a carat at the location where it was expecting the comma.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
924	*** "TAG" EXPECTED HERE (LTERR 924)
CAUSE	LOGTOOL encountered a token other than the reserved word "TAG". LOGTOOL will display a carat where it expected the "TAG".
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
925	*** FIELD BASE MUST BE IN [2,8,10,16] (LTERR 925)
CAUSE	LOGTOOL encountered a value other than a valid base value. LOGTOOL will display a carat indicating where the incorrect base value was found.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
926	*** FIELD BASE EXPECTED HERE [2,8,10,16] (LTERR 926)
CAUSE	LOGTOOL encountered a token other than a valid base value. LOGTOOL will display a carat indicating where the base value was expected.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
927	*** COMMA FOLLOWED BY FIELD BASE EXPECTED HERE (LTERR 927)
CAUSE	LOGTOOL encountered a token other than a comma followed by a field base. LOGTOOL will display a carat indicating where the comma was expected.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
928	*** STRING DATA MUST START ON A 8 BIT BYTE BOUNDARY (LTERR 928)
CAUSE	LOGTOOL requires that string data start on a 8 bit byte boundary. Fields declared before this field must end on a 8 bit byte boundary.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.

929	*** STRING DATA MUST A MULTIPLE OF 8 BITS (LTERR 929)
CAUSE	LOGTOOL requires that string data be declared in multiples of 8 bits.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
930	*** FIELD FORMAT #BITS SUM EXCEEDS RECORD LENGTH (LTERR 930)
CAUSE	The number of bits specified in the FIELD'FORMAT command exceeds the number of bits specified in the RECORD'LENGTH command for the current record definition.
ACTION	LOGTOOL will display the field format of the entry which exceeds the bit length. Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
931	*** FIELD TYPE EXPECTED (LTERR 931)
CAUSE	LOGTOOL was expecting a field type identifier. LOGTOOL will display a carat indicating where it was expecting the field type to be.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
932	*** FIELD BIT COUNT EXPECTED HERE (LTERR 932)
CAUSE	LOGTOOL encountered a token other than a field bit count value. LOGTOOL will display a carat at the location where it is expecting the field bit count.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
933	*** MORE FIELDS IN FIELD FORMAT THEN IN N FIELDS (LTERR 933)
CAUSE	The number of fields specified in the FIELD'FORMAT is larger that the value specified in the N'FIELDS command for the current record definition.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
934	*** INVALID REPEAT FACTOR SPECIFIED (LTERR 934)
CAUSE	The value given for the repeat factor is greater the than the value given in the N'FIELDS command.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
935	*** EXPECTED A REPEAT FACTOR OR "(" (LTERR 935)
CAUSE	LOGTOOL is expecting the next token to be a repeat factor, which is represented as a number, or a "(".
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.

936	*** FIELD FORMAT IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 936)
CAUSE	A FIELD'FORMAT command has already been given for the current record definition, or commands which must proceed the FIELD'FORMAT command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
937	*** NUMBER OF FIELD FORMATS DOES NOT MATCH N'FIELDS VALUE (LTERR 937)
CAUSE	The number of fields given in the FIELD'FORMAT command does not match the value given in the N'FIELD.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
938	*** TOTAL BITS IN FIELD FORMAT DOES NOT MATCH RECORD LENGTH VALUE (LTERR 938)
CAUSE	The total number of bits defined in the FIELD'FORMAT command does not match the RECORD'LENGTH value for the current record definition.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
939	*** N'FIELDS VALUE EXPECTED HERE (LTERR 939)
CAUSE	LOGTOOL encountered the N'FIELD command but did not find the number of fields parameter. Either the user did not give a value, or the value was given in an incorrect format.
ACTION	Please consult the LOGTOOL manual for further information on the N'FIELDS command.
940	*** N'FIELDS IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 939)
CAUSE	A N'FIELD command has already been given for the current record definition, or the commands which must proceed the N'FIELD command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the N'FIELD command.
945	*** QUOTED RECORD LABEL EXPECTED HERE (LTERR 945)
CAUSE	LOGTOOL did not encounter a quoted string, which is to be used as the record label, or the string was not given in the proper format.
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'LABEL command.
946	*** RECORD LABEL IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 946)
CAUSE	The RECORD'LABEL command has already been given for the current record definition, or commands which must proceed the RECORD'LABEL command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'LABEL command.

947	*** RECORD LENGTH VALUE (# BITS) IS OUT OF RANGE (LTERR 947)	
CAUSE	A RECORD'LENGTH command has been encountered, but the user specified a record length value outside of the valid range.	
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'LENGTH command.	
948	*** RECORD LENGTH VALUE (# BITS) EXPECTED HERE (LTERR 948)	
CAUSE	A RECORD'LENGTH command has been encountered, but the user either did not give the number of bits parameter, or gave an invalid token for the number of bits parameter.	
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'LENGTH command.	
949	*** RECORD LENGTH IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 949)	
CAUSE	The RECORD'LENGTH command has already been given for the current record definition, or the commands which must proceed the RECORD'LENGTH command were not given.	
ACTION	Please consult the LOGTOOL manual for further information on the RECORD'LENGTH command.	
951	*** QUOTED VALUE LABEL EXPECTED (LTERR 951)	
CAUSE	LOGTOOL is expecting a quoted string which it will use as a value label. LOGTOOL will display a carat indicating where it expected the quoted string to be.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL command.	
952	*** COMMA FOLLOWED QUOTED VALUE LABEL EXPECTED (LTERR 952)	
CAUSE	LOGTOOL was expecting a comma followed by a quoted string which will be used as a value label. LOGTOOL will display a carat indicating where it was expecting the comma to be.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL command.	
953	*** OPENING PARENTHESIS EXPECTED (LTERR 953)	
CAUSE	LOGTOOL was expecting the next token to be an open parenthesis. LOGTOOL will display a carat indicating where it was expecting the open parenthesis to be.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL or VARIANT command.	
954	*** VALUE LABELS MAY BE SPECIFIED FOR INTEGER FIELDS ONLY (LTERR 953)	
CAUSE	VALUE'LABELS may only be used for fields which have been defined as of type INTEGER within the FIELD'FORMAT command.	
ACTION	Please consult the LOGTOOL manual for further information on the VALUE'LABEL command.	

955	*** VALUE LABEL IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 955)
CAUSE	The VALUE LABEL command has already been given or commands which must proceed the VALUE LABEL command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the VALUE LABEL command.
956	*** LABEL ALREADY SPECIFIED FOR THIS INDEX (LTERR 956)
CAUSE	A label has already been given for the field specified by the field index.
ACTION	Please consult the LOGTOOL manual for further information for the VAR LABEL command.
957	*** INDEX FOR THIS LABEL IS OUT OF RANGE (LTERR 957)
CAUSE	The field for the VAR LABEL cannot be located in the FIELD FORMAT command.
ACTION	Please consult the LOGTOOL manual for further information on the VAR LABEL command.
958	*** QUOTED VAR LABEL STRING EXPECTED (LTERR 958)
CAUSE	LOGTOOL was expecting a quoted string, which will be used as a var label. LOGTOOL will display a carat indicating the location where it was expecting the quoted string.
ACTION	Please consult the LOGTOOL manual for further information on the VAR LABEL command.
959	*** VALUE LABEL IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 959)
CAUSE	The VALUE LABEL command has already been given, or commands which must proceed the VALUE LABEL command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the VALUE LABEL command.
963	*** VARIANT IS INVALID HERE OR HAS ALREADY BEEN SPECIFIED (LTERR 963)
CAUSE	The VARIANT command has already been given, or commands which must proceed the VARIANT command were not given.
ACTION	Please consult the LOGTOOL manual for further information on the VARIANT command.
964	*** LABEL ON LINE ! TO LONG - TRUNCATED TO ! CHARACTERS "! " (LTERR 964)
CAUSE	User has specified a record label longer than 20 characters. The label was truncated to 20 characters. The resulting string will be displayed to the user.
ACTION	Please consult the LOGTOOL manual for further information on the RECORD LABEL command.

966	*** INTERNAL ERROR - DUMP VAL_LABELS LABEL COUNTS DO NOT MATCH. EXPECTED = ! : ACTUAL = ! (LTERR 966)
CAUSE	LOGTOOL encountered an error while trying to save the layout file for the :APPEND option.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameters. b) Any other error messages displayed with this error message. c) The version (A.nn.nn) of the LOGTOOL being executed. A copy of the record definition file used in the LAYOUT LOGTOOL command should also be submitted with the service request.
967	*** INTERNAL ERROR - DUMP=ENTRY VAR COUNTS DO NOT MATCH. EXPECTED = ! : ACTUAL = ! (LTERR 967)
CAUSE	LOGTOOL encountered an error while trying to save the layout file for the :APPEND option.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameters. b) Any other error messages displayed with this error message. c) The version (A.nn.nn) of the LOGTOOL being executed. A copy of the record definition file used in the LAYOUT LOGTOOL command should also be submitted with the service request.
968	*** INTERNAL ERROR DELETING FILE ! (LTERR 968)
CAUSE	LOGTOOL encountered an error while trying to delete the file indicated in the message.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
969	*** INTERNAL ERROR CLOSING FILE ! (LTERR 969)
CAUSE	LOGTOOL encountered an error while trying to close the file indicated in the message.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
978	WARNING: TRUNCATING THE FOLLOWING STRING TO ! CHARACTERS (LTERR 978)
CAUSE	LOGTOOL was forced to truncate the indicated string by the number of characters indicated inorder to perform the ;APPEND option of the LAYOUT command.
ACTION	Please consult the LOGTOOL manual for further information on the format of a record definition files

1000		*** LLIOSTAT TYPE REQUIRES 32 BITS (LTERR 1000)
CAUSE	The FIELD'FORMAT command contained an entry with a type of LLIOSTAT and a parameter which was not a value of 32. A Low level I/O status as defined by the MPE/XL operating system requires 32 bits.	
ACTION	Please consult the LOGTOOL manual for correct usage of the FIELD'FORMAT command.	
1002		*** DATE TYPE REQUIRES 16 BITS (LTERR 1002)
CAUSE	The FIELD'FORMAT command contained an entry with a type of DATE and a length which was not a value of 16. A datestamp as defined by the MPE/XL operating system requires 16 bits.	
ACTION	Please consult the LOGTOOL manual for correct usage of the FIELD'FORMAT command.	
1003		*** MPEXLSTAT TYPE REQUIRES 32 BITS (LTERR 1003)
CAUSE	The FIELD'FORMAT command contained an entry with a type of MPEXLSTAT and a length which was not a value of 32. A MPE/XL status as defined by the MPE/XL operating system requires 32 bits.	
ACTION	Please consult the LOGTOOL manual for correct usage of the FIELD'FORMAT command.	
1004		*** TIME TYPE REQUIRES 32 BITS (LTERR 1004)
CAUSE	The FIELD'FORMAT command contained an entry with a type of TIME and a length which was not a value of 32. A timestamp as defined by the MPE/XL operating system requires 32 bits.	
ACTION	Please consult the LOGTOOL manual for correct usage of the FIELD'FORMAT command.	
1005		*** LAST RECORD'ID OR TAG'ID IS NOT COMPLETE (LTERR 1005)
CAUSE	LOGTOOL has encountered a RECORD'ID or TAG'ID command before all required information for the previous record definition has been given. The partial record definition will be ignored.	
ACTION	Please consult the LOGTOOL manual for the correct format of a record definition.	
1006		*** VALUE GIVEN IN N'FIELDS CANNOT BE NEGATIVE (LTERR 1006)
CAUSE	The number of fields for the current record definition number be equal to or greater than zero. This value must match the number of entries given in the FIELD'FORMAT command.	
ACTION	Please consult the LOGTOOL manual for the correct usage of the N'FIELD command.	
1007		*** UNABLE TO COPY FILE ! TO SYSTEM DISK (LTERR 1007)
CAUSE	LOGTOOL was not able to copy the layout files to the main system disk.	
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.	

1008	*** UNABLE TO FULLY QUALIFY LAYOUT FILE NAME (LTERR 1008)
CAUSE	LOGTOOL was not able to fully qualify the file name for the layout files. LOGTOOL cannot create the layout files with fully qualifying the file name.
ACTION	An error message containing further information will be displayed with this message. Please refer to the ACTION clause of that error message for further information.
1009	*** LENGTH OF FIELD CONTAINING HARDWARE STATUS COUNT EXPECTED (LTERR 1009)
CAUSE	LOGTOOL was expecting value which is the length of the field containing the hardware status count. LOGTOOL will display a carat indicating the location where it was expecting the value.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
1010	*** INDEX TO THE FIELD CONTAINING HARDWARE STATUS COUNT EXPECTED (LTERR 1010)
CAUSE	LOGTOOL was expecting value which is the index where the field containing the hardware status count can be found. LOGTOOL will display a carat indicating the location where it was expecting the value.
ACTION	Please consult the LOGTOOL manual for further information on the FIELD'FORMAT command.
30011	WARNING: LOGTOOL DOES NOT USE "ERRCOUNT" PARAMETER. (LTWARN 30011)
CAUSE	The user gave the ERRCOUNT parameter when initiating LOGTOOL through the diagnostic user interface. LOGTOOL does not use this parameter.
ACTION	LOGTOOL will ignore the ERRCOUNT parameter and will continue execution.
30012	WARNING: LOGTOOL DOES NOT USE "LOOPCOUNT" PARAMETER. (LTWARN 30011)
CAUSE	The user gave the LOOPCOUNT parameter when initiating LOGTOOL through the diagnostic user interface. LOGTOOL does not use this parameter.
ACTION	LOGTOOL will ignore the LOOPCOUNT parameter and will continue execution.
30013	WARNING: LOGTOOL DOES NOT USE "ERRONLY" PARAMETER. (LTWARN 30012)
CAUSE	The user gave the ERRONLY parameter when initiating LOGTOOL through the diagnostic user interface. LOGTOOL does not use this parameter.
ACTION	LOGTOOL will ignore the ERRONLY parameter and will continue execution.
30014	WARNING: LOGTOOL DOES NOT USE "ERRPAUSE" PARAMETER. (LTWARN 30013)
CAUSE	The user gave the ERRPAUSE parameter when initiating LOGTOOL through the diagnostic user interface. LOGTOOL does not use this parameter.
ACTION	LOGTOOL will ignore the ERRPAUSE parameter and will continue execution.

Contents

2. HP-UX Logging Facility	
Introduction	2-1
Defects and Enhancements	2-1
Minimum Configuration	2-1
Operating Instructions	2-1
Test Execution	2-2
Test Section Descriptions	2-2
DELOG	2-2
DECODE	2-6
Error Messages	2-10



HP-UX Logging Facility

Introduction

The HP-UX Logging Facility provides the user with a means of obtaining and decoding Diagnostic Event Messages (DEM's). The acquisition of these event messages is handled by the HP-UX DELOG program. The decoding of these messages is accomplished by using the HP-UX DECODE program. The DELOG process runs as a daemon and reads diagnostic event messages from the diagnostic driver "diag0" while DECODE runs as a user application and does not require any special privilege status. Raw DEM's are written into the logfile "/usr/adm/diaglog" on a regular basis by DELOG. DECODE reads a DEM from either "stdin" or a designated file and then decodes it into readable text. (DELOG and DECODE are standard HP-UX operating system commands and do not necessarily need to be run from the Online Diagnostic subsystem DUI.)

Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database using the Standard Defect Traffic System (SDTS) number 92453A.

Minimum Configuration

The minimum configuration required to run this utility consists of an HP Precision Architecture RISC computer system up and running on the HP-UX operating system.

Operating Instructions

To bring up the Online Diagnostic subsystem, enter the following command to the HP-UX prompt:

```
% /usr/diag/bin/sysdiag
```

The system responds with the following prompt indicating that access has been gained to the Online Diagnostic subsystem:

```
DUI (n)>
```


Test Execution

Users may now call **DELOG** to alter the operation of the **DELOG** background log process daemon, or **DECODE** to display a particular Diagnostic Event Message (DEM). The example below shows how to run either of these standard HP-UX commands from the Online Diagnostic subsystem DUI by preceeding each command entry with an exclamation point:

```
DUI (n)> !delog
```

or

```
DUI (n)> !decode
```

Test Section Descriptions

The following section describes the functions and use of the **DELOG** and **DECODE** commands. Each description includes syntax, options, and an example.

DELOG

DELOG requires at least one special device file in **/dev** to run. The name of that file should be "diag0" and the "minor number" must be 0. If any other name is used for the file, **DELOG** cannot use the default name it knows and the user (system administrator) must edit the **/etc/rc** script to cause **DELOG** to use the new name.

A second special device file may be created using the "minor number" other than zero. This will allow **DELOG** to enter "diag0" as a user application but with limited access to the functionality of "diag0".

In both cases, only the root should have access to these files.

The Diagnostic Event Logger (**DELOG**) is written to be run as an HP-UX daemon. It is started up by "rc" at bootup time when the system goes from single-user to multi-user mode. The main purpose of **DELOG** is to read Diagnostic Event Messages (DEM) from the diagnostic driver "diag0". **DELOG** can also write a short synopsis of the DEM to the system console and block the processing of all DEM's at the system level.

The DEM is a generic error message. Whenever a driver has a message (either warning, error, or simply informational) that should be logged, it sends a DEM to the diagnostic driver, "diag0". DEM's are very flexible messages. Each DEM consists of three parts:

A fixed portion	Contains such general information as error class (hardware or software), retry counts, and special flags.
Hardware status	Hardware status and (MSI) share a variable buffer.
Manager Specific Information (MSI)	(MSI) and hardware status share a variable buffer. MSI gives the manager a flexible area in which any additional information may be placed.

DELOG uses standard system calls to read DEM's from the diagnostic drivers and to write the DEM back out to the diagnostic log file. When DELOG reads from the diagnostic driver, the driver puts the logger to sleep waiting for an event to be received. When "diag0"'s interrupt handler processes an event, it wakes up DELOG and completes the pending read request. DELOG then opens the log file and writes the DEM into a log file. The log file is then closed and the whole process repeats when necessary.

Before the DEM is actually written out to the log file, information is retrieved from the I/O tree and tagged on to it. The I/O tree is tied directly to the system configuration and contains such information as:

- Manager name
- Hardware path
- lu
- Major number

Although DELOG uses some of this information, most of the information is saved with the DEM for the DECODE process' detailed report on all logged DEM's. Thus, DECODE does not need to be directly tied to the system and its configuration. The log file may be moved to a remote system and be decoded there even though that system uses an entirely different I/O configuration.

Syntax:

`Delog [<optional parameters>]`

Optional Parameters:

DELOG has several optional parameters. These are broken into two classes. The first class alters the default parameters of DELOG. The second class alters the functionality of either DELOG or "diag0". Each parameter is independent of the other, although changing the "vmunix" file may also require setting the version. When DELOG is run, all necessary parameters are defaulted to an appropriate value.

The following list of parameters may be used to change the default values:

Option	Value Changed
-d	<device file> Sets the device file; the users must enter the full path. The default value is /dev/diag0.
-L	<log file> Sets the diagnostic log file; the user must enter the full path. The default value is "/usr/adm/diaglog".
-U	<vmunix file> Sets the kernel file name; the user must enter the full path. The default value is "/hp-ux".
-V	<version number> Sets the version number of the kernel. The default value is -3; this will change for release.

There are four more parameters that DELOG will accept which can alter its functionality or that of "diag0".

Option	Value Changed
-w	Print out a brief synopsis of every event received.
-e [1/2]	Print out a brief synopsis of only those events that are errors and for which the I/O transaction actually failed.
-b	Send synopsis to both the system console and a user-defined file.
-f	Sets the output file path name. Default is /dev/null.

2-4 HP-UX Logging Facility

Console Messages:

When **DELOG** is run without any parameters, no messages are written to the system console. In most cases, however, a system administrator will want some level of logging to be done at the system console. The **-w** and **-e** options provide this.

The **-w** option prints out warning and error information. Warnings are informational and are not always significant. Errors may be informational or they may be reports of fatal errors.

Therefore, a large amount of data, mostly informational, will go to the system console. The **-e** option causes **DELOG** to print messages that are only I/O subsystem errors. The **-e** and **-w** options are mutually exclusive. However, due to the nature of I/O subsystem messages, not all superfluous informational messages can be suppressed. The format of the output for the **-e** option is the same as that of the **-w** option.

To further reduce the amount of data written to the system console, the **-e** option has a second numeric parameter. The default value of the number is 1 if nothing is specified. When the number is 2, the hardware status is suppressed.

The example message can be redirected to a file using the **-f** option. In combination with the **-b** option the messages can be sent to the system console and the designated file. If the **-b** option is used by itself, the result is the same as with the **-w** option.

Example:

```
Diagnostic class type from XX (at a.b.c) status = x
Hw stat = (hardware status printout as a HEX string)
```

Where:

class	is replaced with either hardware or software.
type	is replaced with either error or warning.
XX	is the name of the driver that reported the event.
a.b.c	is the hardware path derived from the I/O tree.
x	is an ASCII string corresponding to the LLIO status.



When a system call fails, **DELOG** terminates. For example, if the file system is full, logging to the log file will force an error and **DELOG** will halt. Termination of **DELOG** indicates an error.

DECODE

This command allows the user to gain access to the Diagnostic Event Decoder (DECODE). No special requirements are needed for DECODE to work. In fact, the log files from one HP-UX system can be moved to another HP-UX system and be decoded there. This is true even if the two systems are not configured in a similar fashion.

DECODE is a user application program that translates a diagnostic event message (DEM) into readable text. DEM's are read from the log file, which is updated by DELOG. DELOG is the HP-UX daemon logger process that receives DEM's from the Operating System. (For more details, refer to the previous DELOG discussion.)

DECODE reads a DEM from either "stdin" or a designated file. If the input is from a file, DECODE will decode all of the DEM in that file. The actual formatting of the message depends on the options the user specifies. The options are used to selectively look at one or more types of DEM's. The following section defines what the options are and how they work.

Syntax:

Decode [*<optional parameters>*]

Optional Parameters:

The options for Decode are:

<i>-L</i>	<i><error log></i> Sets the error log path/name. (default is stdin)
<i>-d</i>	<i><dev file></i> Sets the path for a special device file.
<i>-e</i>	<i><physical path></i> Sets the hardware path. (uses the 1.2.3 notation)
<i>-m</i>	<i><major number></i> Sets the major number.
<i>-n</i>	<i><driver name></i> Sets the driver name.
<i>-p</i>	<i><port></i> Sets the port number.
<i>-t</i>	<i><tail></i> Sets the tail parameter. (default is last 10 events)
<i>-w</i>	<i><wait></i> Sets a flag to wait after printing each event.

When setting the log file path, the full path must be specified unless the log file is in the current directory.

The options -d, -e, -m, -n, and -p are used to selectively print out information. If the -d option is used then the -m option is ignored. This results because device file information is found by converting the device special file to a "major number" and "lu". This can conflict with the -m option which specifies the "major number".

The -t option acts like the tail command. When "n" is positive it shows the last "n" events in the log file. When "n" is negative the first "n" events in the log file are skipped over.

The -w option provides a way to step through the logfile one event at a time.

Input Example:

DECODE displays events in the following format:

Diagnostic Event number X, Date = DATE & TIME

Port number of originating event = aaaaaaaa

Manager of originating event = bbbbbbbb

Physical path = a.b.c

Diagnostic event status:

error	- cccccccc
proc num	- dddddddd
subsystem	- eeeeeeee

Diagnostic class = ffffffff

Number of retries = gggggggg

Hardware status = (status bytes in HEX formatted
string)

Mgr info = (Manager specific information in HEX
formatted string)

Where:

x	the diagnostic event number relative to the start of the log file.
DATE & TIME	the date and time that "diag0" (DEV) processed the event.
aa	port number of the I/O subsystem entity that reported the event.
bb	manager name for reporting port number derived from the I/O tree.
a.b.c	physical path as derived from the I/O tree and the manager specific information.
cc	is an ASCII string corresponding to the LLIO status.
dd	is an unique number used to identify the I/O subsystem entity.
ee	is an unique number used to identify the I/O subsystem entity.
ff	is replaced with either DIAG_HW_EVENT or DIAG_SW_EVENT.
gg	the number of retries the reporting entity tried before sending the event.

Output Example:

A example of a DEM that DECODE has processed is shown below:

```
decode diagnostic event

Read diagnostic event from /usr/adm/diaglog

Diagnostic Event number 51,   Date = Tue Apr 29 09:55:46 1986

Port number of originating event = 14

Manager of originating event = tape11

Physical path = 8.2.3

Diagnostic event status:
    error          - TIMEOUT
    proc num       - 4
    subsystem      - 1042

Diagnostic class = DIAG_HW_EVENT

Number of retries = 4

Hardware status = 0x0000000038ac

Mgr info = 0x041200000008000818020000000000
```

The above example is of a DEM from the magtape driver. The event was the result of trying to TAR to the magtape drive when it was offline. The “port number”, “manager name”, and “physical path” are derived from the “iotree” by DELOG when the event was first logged. The physical path tracks the hardware path out to the device that sent the message. In this case, 8.2.3 would lead through the following path: the channel adapter with module 8, to slot 2 in the CIO backplane, and finally to the device that was using address 3 in that slot.

DECODE will issue an error message if the user specifies a log file which does not exist. See the error message section of this diagnostic.

Error Messages

The following is a list of error messages which may appear when using the HP-UX Logging facility. Listed below each error message are a probable cause and recommended action statement. In addition, the user will see a usage line printed if an invalid or incomplete parameter is used while running DECODE or DELOG. This usage line is as follows:

For DECODE:

```
[-L logfile] [-d dev file path] [-e physical path]
[-m major number] [-n driver/manager] [-p port]
[-t [events] ] [-w]
```

For DELOG:

```
[ [-e [number] ] [-w] ] [-b] [-d device file] [-f output file]
i [-l logfile] [-u umunix file] [-v version number]
```

The following are error messages that can be seen while running the HP-UX Logging facility:

ERROR:	WRONG FORMAT FOR PHYSICAL PATH
CAUSE	The user has entered physical path information that is not valid.
ACTION	The -e option must have its parameter in the following form: number [.number].number]
ERROR:	NO DEVICE FILE CORRESPONDING TO SPECIFIED PATH.
CAUSE	The user has a special device file that does not exists.
ACTION	
ERROR:	OPEN FAILED, FOR X WHERE X IS THE PATH/NAME OF THE DIAGNOSTIC LOG.
CAUSE	The user has a diagnostic log file that does not exists.
ACTION	
ERROR:	THE -T OPTION IS OUT OF RANGE.
CAUSE	The user has specified a value for the -t option that does not exist in the file. (See manual for -t option).
ACTION	
ERROR:	ERROR IN DETERMINING SIZE OF X WHERE X IS THE PATH/NAME OF THE DIAGNOSTIC LOG.
CAUSE	System call failed.
ACTION	

ERROR:	ERROR IN SELECTING EVENT IN WHERE X IS THE PATH/NAME OF THE DIAGNOSTIC LOG.
CAUSE	System call failed.
ACTION	



Contents

3. System Map	
Introduction	3-1
Defects and Enhancements	3-1
Minimum Configuration	3-1
Operating Instructions	3-1
RUN Command	3-1
Test Execution	3-2
Test Section Descriptions	3-3
IOMAP	3-3
IOMAP Commands	3-4
GENERAL MAP	3-5
CLASS MAP	3-6
STEPPING	3-8
CONFIRM	3-10
SUSPEND	3-12
EXIT	3-12
EXIT ALL	3-12
CPUMAP	3-13
MEMMAP	3-15
MODULEMAP	3-16
Information returned by MODULEMAP	3-18
Error Messages and Warnings	3-19

System Map

Introduction

The System Map utility provides users with a means of obtaining hardware configuration information about an HP Precision Architecture RISC computer system running online within the MPE XL operating system. System Map provides information concerning four areas of the computer system; Input/Output system (IOMAP), Central Processing Unit(s) (CPUMAP), System Memory (MEMMAP), and system modules (MODULEMAP). Maps of these three areas are available only on the host system.

Defects and Enhancements

Submit defect reports and enhancement requests for this diagnostic through the STARS database referencing product number 30600-10021.

Minimum Configuration

The minimum configuration required to run this utility consists of an HP Precision Architecture RISC computer system up and running on the MPE XL operating system.

Operating Instructions

RUN Command

To bring up the Online Diagnostic Subsystem, enter the following command to the MPE XL system prompt:

SYSDIAG

The system responds with the following prompt indicating that access has been gained to the 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.

Note

The device to be tested must be powered up and on line. Device physical locations (pdev) shown in the RUN commands are those of the devices on the "typical A1002A" system configuration described in the chapter on DUI. The pdev value entered must be correct for the system being tested.

To run the diagnostic, enter:

```
DUI >RUN SYSMAP
|
no parameters required to
load test suite
```

Test Execution

The utility responds with the following header and welcome message:

```
*****
****                                     ***
****                      SYSTEM MAP      ***
****                                     ***
****   (C) Copyright Hewlett Packard Co. 1987   ***
****                      All Rights Reserved.  ***
****                      Version A.00.00      ***
****                                     ***
*****
```

Welcome, Today is MON, MAY 22, 1987, 9:00 AM

Once SYSMAP has been invoked, the following prompt is displayed, indicating an input request:

ENTER MAP>

SYSMAP allows the user to abbreviate all input as long as the input is a unique string of characters. For example, typing c, cpu, or cp will invoke CPUMAP.

3-2 System Map

Typing **HELP** causes **SYSMAP** to list the following menu of global **SYSMAP** commands:

IOMAP	MAP OF I/O SYSTEM
CPUMAP	MAP OF CPUs ON SYSTEM
MEMMAP	MAP OF MEMORY ON SYSTEM
MODULEMAP	MAP OF SYSTEM MODULES
CONFIRM ON/OFF	CONFIRM HARDWARE PRESENCE
TIMEOUT	SET TIME IN SECONDS FOR CONFIRM MODE
SUSPEND	SUSPEND THIS PROCESS
EXIT	EXIT PROGRAM

To terminate any current mapping process or to leave the **HELP** facility, enter **EXIT**.

Test Section Descriptions

The test maps that will be described in this section are:

- **IOMAP**
- **CPUMAP**
- **MEMMAP**
- **MODULEMAP**

IOMAP

IOMAP provides the capacity to map the I/O hardware on a particular HP Precision Architecture RISC computer system. It has two modes of operation and provides users with two kinds of maps; the general map and the class map. **IOMAP** displays those devices currently configured in the system tables. With the use of confirm mode, **IOMAP** can provide the user with a physical map of the I/O system.

The following IOMAP display example is typical of all IOMAP screen output. It explains how to interpret the different values and parameters encountered while using the IOMAP utility:

```
physical address element
```

|
|---
| |
| |
Channel Adapter(2) |
Device Adapter(12) --- hpib <----Type of Device Adapter
Dev(1) [ldev 36 xxxx Disc]
			Product class type
			Product number
			logical device number
physical address element

Physical address:	In the above example the physical address of the device is actually 2.12.1. The physical address of the device adapter is 2.12 and the physical address of the channel adapter is 2. Each level's physical address is a combination of the previous level's physical address element and its own physical address element.
Logical Device Number:	Number assigned to module at system configuration.
Product number:	Hewlett-Packard product number.
Product class:	Type of device (i.e., Tape, Disc, Printer).
Device adapter:	Type of device adapter present, HPIB or Alink.

IOMAP Commands

The following pages document the current IOMAP command set. The commands are explained and examples of display output are shown. Some commands control the mode of IOMAP program operation, some select the kind of mapping to be carried out while others enable various options available through IOMAP.

IOMAP will map terminals if they are present in the physical path table.

GENERAL MAP

The general map creates a map of all the intermediate hardware and peripherals on a particular host system. This map supplies the following information for each device:

- Bus converter connections
- Channel adapter connections
- Device adapter connections
- Device address
- Logical device number
- Product number
- Product type

If the general map is selected, IOMAP produces a general host peripheral map, which includes all the devices configured in the system tables. The user is then returned to the mode selection prompt of IOMAP. The general host peripheral map is displayed in the following form after the user has entered the **GENERAL** command:

```
IOMAP>general
```

```

                                IOMAP GENERAL HOST MAP
-----
Channel Adapter(8)
  Device Adapter(0) --- HPIB
    Device(0) [ ldev 1 HP7935 DISC ]
    Device(1) [ ldev 20 HP7935 DISC ]
    Device(2) [ ldev 21 HP7935 DISC ]
    Device(3) [ ldev 22 HP7935 DISC ]
  Terminal Mux(1)
    Device(0) [ ldev 4 HP2624B TERMINAL ]
    Device(1) [ ldev 5 HP2624B TERMINAL ]
    Device(2) [ ldev 6 HP2624B TERMINAL ]
    Device(3) [ ldev 7 HP2624B TERMINAL ]
    Device(4) [ ldev 8 HP2624B TERMINAL ]
    Device(5) [ ldev 9 HP2624B TERMINAL ]
  Device Adapter(2) --- HPIB
    Device(0) [ ldev 16 HP7935 DISC ]
    Device(2) [ ldev 23 HP7978 TAPE ]
    Device(3) [ ldev 14 HP7978 TAPE ]
    Device(4) [ ldev 15 HP7978 TAPE ]
    Device(6) [ ldev 12 HP2688A PAGE PRINTER ]
    Device(7) [ ldev 13 HP2566A LINE PRINTER ]
-----

IOMAP>
```

CLASS MAP

The class map produces a map for a desired device class only. This map supplies the following information on the chosen device class:

- Bus converter connections
- Channel adapter connections
- Device adapter connections
- Device address
- Logical device number
- Product number
- Product type

If the class map is chosen, the user is asked which device class is to be examined. The following prompt is displayed after the user has entered "class" to the IOMAP> prompt:

IOMAP CLASS>

If HELP is entered, the following menu is displayed along with the prompt:

IOMAP CLASS>

commands:

DEVICE CLASS	(CHOOSE FROM LIST OF DEVICE CLASSES BELOW)
CONFIRM (ON/OFF)	(CONFIRM HARDWARE PRESENCE)
SUSPEND	(SUSPEND THIS PROCESS)
EXIT	(EXIT CLASS MODE)
EXIT ALL	(EXIT SYSMAP)

The following device classes are available:

disk	tape
line printer	page printer
hpib adapter	alink adapter
bus converter	terminal
channel adapters	term mux (6 port mux)

```
IOMAP CLASS>disk
```

When a device class has been entered, IOMAP gathers and displays the desired information. The selected device class map is shown in the following format:

```
                                IOMAP CLASS MAP
DISC(S):
-----
Channel Adapter(8)
  Device Adapter(0) --- HPiB
    Device(0) [ ldev 1 HP7935 DISC ]
    Device(1) [ ldev 20 HP7935 DISC ]
    Device(2) [ ldev 21 HP7935 DISC ]
    Device(3) [ ldev 22 HP7935 DISC ]
  Device Adapter(2) --- HPiB
    Device(0) [ ldev 16 HP7935 DISC ]
-----
```

The user may terminate this mode by entering EXIT in response to the CLASS> prompt. At this time the user is returned to the IOMAP> prompt:

```
IOMAP CLASS> exit
IOMAP>
```

STEPPING

In stepping mode, the user can select a portion of the system to be examined. When this mode is used, information concerning the highest current level (first level of hardware encountered in respect to the bus) of I/O hardware is displayed. The user may decide to follow any of the paths indicated by entering the address of any top level device shown. At this point, the next level of hardware below the specified address will be displayed. The user now has several options that are explained in more detail in the stepping mode section of this document.

When step mode is entered, a display of the highest level of hardware on the system is shown followed by the step mode prompt. The user may step down through the system I/O hardware to any specific section of the map. The user may also indicate which path to follow by entering the physical address of the module to be mapped. If a user enters **all** at any level, then all devices below that level will be mapped. If a device address is specified at a given level, the user will receive a map of the level of devices below the specified device. A new prompt containing the physical address information already entered also appears. The "top" option may be selected at any level by the user to return to a map of the highest level of hardware on the system. Entry of the "same" option will display the current level again. Entry of the "previous" option will display the previous level again. To access the step mode, enter the **STEPPING** command to the **IOMAP>** prompt:

```
IOMAP >step
```

IOMAP STEPPING MODE

```
-----  
Channel Adapter(8)  
-----
```

```
CURRENT POSITION---> TOP  
IOMAP STEP MODE > help
```

Commands

```
ALL      (MAP OF ALL THE HARDWARE BELOW A SPECIFIC DEVICE.)  
TOP      (TO GO BACK TO THE HIGHEST LEVEL OF HARDWARE.)  
SAME     (DISPLAY THE CURRENT LEVEL AGAIN.)  
PREVIOUS (DISPLAY THE PREVIOUS LEVEL AGAIN.)  
CONFIRM  (ON/OFF) (CONFIRM HARDWARE PRESENCE.)  
TIMEOUT  (SET TIME IN SECONDS FOR CONFIRM MODE.)  
SUSPEND  (SUSPEND THIS PROCESS.)  
EXIT     (EXIT STEP MODE.)  
EXIT ALL (EXIT SYSMAP.)
```

Stepping directions

In order view a particular section of the map, the user must select from the items displayed, the numerical value of the path (s)he wishes to traverse.

3-8 System Map

```

IOMAP STEPPING MODE> 8 (The user must type and return "8".)

DEVICES UNDER PHYSICAL ADDRESS 8 - CA
-----
Device Adapter(0) --- HPIB
Terminal Mux(1)
Device Adapter(2) --- HPIB
-----

CURRENT POSITION---> 8.
IOMAP STEP MODE > 2 (The user must type the physical element address.)
DEVICES UNDER PHYSICAL ADDRESS 8.2 - CA.DA
-----
Device(0) [ ldev 16 HP7935 DISC ]
Device(2) [ ldev 23 HP7978 TAPE ]
Device(3) [ ldev 14 HP7978 TAPE ]
Device(4) [ ldev 15 HP7978 TAPE ]
Device(6) [ ldev 12 HP2688A PAGE PRINTER ]
Device(7) [ ldev 13 HP2566A LINE PRINTER ]
-----

CURRENT POSITION---> 8.2.
IOMAP STEP MODE> exit

IOMAP>

```

The user may exit this mode at any time by entering **ex**. IOMAP will then return to the map selection prompt.

CONFIRM

The confirm mode of IOMAP is used when confirmation of the system tables and the physical system is necessary. When this mode is active, all IOMAP's maps are confirmed by issuing roll call and identify commands to each level and device respectively.

The CONFIRM command verifies any map displayed. This is accomplished by issuing roll call and identify commands to each level of hardware encountered. The CONFIRM command can be issued from any of the above-mentioned maps or modes. The default mode of this command is "confirm off". Due to the disruptive nature of some identify commands, only users with the capability of running diagnostics at security level 2 will be allowed use of this mode. The "timeout" option allows the user to set the time in seconds for the confirm mode process. The confirm discrepancies information will follow the general host map display as shown below:

```
IOMAP> confirm on

CONFIRM MODE IN PROGRESS. PLEASE STANDBY.

IOMAP> gen

-----
IOMAP GENERAL HOST MAP
-----
Channel Adapter(8)
  Device Adapter(0) --- HP1B
    Device(0) [ ldev 1 HP7935 DISC ]
    Device(1) [ ldev 2 HP7935 DISC ***NOT PRESENT***]
    Device(2) [ ldev 3 HP7935 DISC ***NOT PRESENT***]
    Device(3) [ ldev 4 HP7935 DISC ***NOT PRESENT***]
    Device(4) [ ldev 14 HP7935 DISC ***NOT PRESENT***]
    Device(5) [ ldev 15 HP7935 DISC ***NOT PRESENT***]
    Device(6) [ ldev 16 HP7935 DISC ***NOT PRESENT***]
    Device(7) [ ldev 17 HP7935 DISC ***NOT PRESENT***]
  Terminal Mux(1)
    Device(0) [ ldev 20 HP2624B TERMINAL ]
    Device(1) [ ldev 21 HP2624B TERMINAL ]
    Device(2) [ ldev 22 HP2624B TERMINAL ]
    Device(3) [ ldev 23 HP2624B TERMINAL ]
    Device(4) [ ldev 24 HP2624B TERMINAL ]
    Device(5) [ ldev 25 HP2624B TERMINAL ]
  Device Adapter(2) --- HP1B
    Device(2) [ ldev 8 HP7978 TAPE ***NOT PRESENT***]
    Device(3) [ ldev 7 HP7978 TAPE ]
    Device(4) [ ldev 10 HP7978 TAPE ***NOT PRESENT***]
    Device(6) [ ldev 19 HP2688A PAGE PRINTER ***NOT PRESENT***]
    Device(7) [ ldev 6 HP2566A LINE PRINTER ***NOT PRESENT***]
-----
```

CONFIRM DISCREPANCIES:

Channel Adapter(8)

Device Adapter(2) --- HPiB

Device Unknown(5) ***NOT CONFIGURED***

IOMAP> confirm off



SUSPEND

IOMAP has a **SUSPEND** command that will suspend the program in the Diagnostic User Interface (DUI). This enables the use of various features of the DUI. In particular, the suspend option permits use of the hardcopy command of the DUI.

The **SUSPEND** command permits the use of Diagnostic User Interface (DUI) commands. An example of this is use of the **HARDCOPY** command. To use **HARDCOPY**, enter **SUSPEND**, turn **HARDCOPY** on, enter **RESUME**, and specify a map to be displayed. The user can now get a hard copy of the desired map.

EXIT

If **EXIT** is selected, IOMAP terminates and returns the user to the **ENTER MAP>** prompt.

EXIT ALL

If **EXIT ALL** is selected, SYSMAP terminates and returns the user to the **DUI >** prompt.

CPUMAP

CPUMAP provides the user with a map of the CPU modules that are configured on the computer system. CPUMAP provides the user with information about the hardware model number, hardware revision number, software model number, and software revision number of the CPU. The slot number of the CPU module is also given.

In order to run Cpumap the user must respond to the **Enter Map** prompt in the following way:

```
DUI >run sysmap
*****
*****                                     *****
*****                               SYSTEM MAP                               *****
*****                                     *****
*****                               (C) Hewlett Packard Corporation           *****
*****                               Version A.02.00                         *****
*****                                     *****
*****
```

Welcome, Today is FRI, APR 14, 1989 at 7:47 AM

ENTER MAP >Cpumap

CPUMAP OUTPUT:

SYSTEM CPU MAP

```
-----  
CPU NUMBER 1 - HARDWARE MODEL NUMBER-----> 101  
              HARDWARE REVISION NUMBER----> 0  
              SOFTWARE MODEL NUMBER-----> 0  
              SOFTWARE REVISION NUMBER----> 65  
              SOFTWARE OPTION-----> 0  
              HARDWARE ID-----> 0  
              SOFTWARE ID-----> 0  
              BOOT ID-----> 0  
              HARD PHYSICAL ADDRESS-----> FFF80000  
              SLOT NUMBER-----> 0
```

Co-processors:

FLOATING POINT COPROCESSOR IS INSTALLED.

CACHES:

```
INSTRUCTION CACHE SIZE - 64K BYTES  
DATA CACHE SIZE        - 64K BYTES  
INSTRUCTION TLB SIZE   - 4K BYTES  
DATA TLB SIZE          - 4K BYTES  
-----
```

MEMMAP

MEMMAP provides the user with a map of the memory modules that are configured on an HP Precision Architecture RISC computer system. Specific information about the location (slot number) of the memory controllers and memory array boards is displayed, along with the Hard Physical Address of each memory controller and the memory size of all memory modules. In order to run MEMMAP, the user must respond to the ENTER MAP> prompt in the following way:

```
DUI > run sysmap
```

```
*****
****                                     ***
****          SYSTEM MAP                ***
****                                     ***
****   (C) Copyright Hewlett Packard Co. 1987   ***
****          All Rights Reserved.             ***
****          Version A.00.00                  ***
****                                     ***
*****
```

```
Welcome, Today is MON, MAY 22, 1987, 9:00 AM
```

```
ENTER MAP > memmap
```

Memory Map will produce the following output to display the current memory configuration:

```
          SYSTEM MEMORY MAP
-----
      MODULE TYPE      SLOT NUMBER      HPA      MEMORY SIZE
      =====
MEMORY CONTROLLER #1          9      FFFA4000      3145728 BYTES
      MEMORY ARRAY #1          8          -      5242880 BYTES
MEMORY CONTROLLER #2         11      FFFAC000      3145728 BYTES
      MEMORY ARRAY #2         10          -      5242880 BYTES
MEMORY CONTROLLER #3         13      FFFB4000      3145728 BYTES
      MEMORY ARRAY #3         12          -      5242880 BYTES
-----
MEMORY CONTROLLER #1 - HARDWARE MODEL NUMBER - 8
MEMORY CONTROLLER #1 - REVISION NUMBER - 0
MEMORY CONTROLLER #2 - HARDWARE MODEL NUMBER - 8
MEMORY CONTROLLER #2 - REVISION NUMBER - 0
-----
```

```
ENTER MAP > exit
      SYSMAP Exiting ...
      SYSMAP (PIN 23) has just terminated.
DUI >exit
```

MODULEMAP

MODULEMAP provides the user with a map of the Precision Architecture RISC Modules. These modules are also referred to as Precision I/O Modules. The definition of such modules is as follows: "An entity which appears in the system address space and adheres to the Precision I/O Architecture." Module types fitting this description are modules such as the native processors, memory controllers, bus converters, foreign processors and bus adapters.

In order to run Module map the user must respond to the Enter Map prompt in the following way:

```
DUI >run sysmap
*****
*****                                *****
*****                                SYSTEM MAP                                *****
*****                                *****
*****                                (C) Hewlett Packard Corporation            *****
*****                                Version A.02.00                          *****
*****                                *****
*****
*****
```

Welcome, Today is MON, APR 3, 1989 at 8:47 AM

ENTER MAP >modulemap

MODULE MAP OUTPUT:

SYSTEM MODULE MAP

```
-----  
MODULE TYPE-----> MEMORY CONTROLLER  
HARDWARE MODEL NUMBER-----> 4  
HARDWARE REVISION NUMBER-----> 2  
SOFTWARE MODEL NUMBER-----> 9  
SOFTWARE REVISION NUMBER-----> 0  
SOFTWARE OPTION NUMBER-----> 0  
HARD PHYSICAL ADDRESS-----> FFF8000  
BUS IDENTIFICATION-----> 3FE  
SLOT NUMBER-----> 0
```

```
MODULE TYPE-----> BUS CONVERTER PORT  
HARDWARE MODEL NUMBER-----> 4  
HARDWARE REVISION NUMBER-----> 2  
SOFTWARE MODEL NUMBER-----> C  
SOFTWARE REVISION NUMBER-----> 0  
SOFTWARE OPTION NUMBER-----> 0  
HARD PHYSICAL ADDRESS-----> FFF82000  
BUS IDENTIFICATION-----> 3FE  
SLOT NUMBER-----> 0
```

.
.
.
.

```
MODULE TYPE-----> NATIVE PROCESSOR  
HARDWARE MODEL NUMBER-----> 80  
HARDWARE REVISION NUMBER-----> 0  
SOFTWARE MODEL NUMBER-----> 4  
SOFTWARE REVISION NUMBER-----> 0  
SOFTWARE OPTION NUMBER-----> 1  
HARD PHYSICAL ADDRESS-----> FFF89000  
BUS IDENTIFICATION-----> 3FE  
SLOT NUMBER-----> 1
```

```
-----  
ENTER MAP >exit  
SYSMAP Exiting ...  
SYSMAP (PIN 23) has just terminated.  
DUI >exit
```

Information returned by MODULEMAP

The following is an explanation of the information returned by MODULEMAP. A knowledge of HVERSIONS and SVERSIONS as described in the latest version of the ACD would give the user a better understanding of the model and revision number information returned by MODULEMAP.

MODULE TYPE	- Module type is that information which identifies what type of Precision Architecture RISC Module information is currently being displayed for. An explanation of a precision architecture module is given in 2.1 Product Overview under the MODULEMAP section.
HARDWARE MODEL NUMBER	- Bits 0..11 of the module's hversion. This field specifies the module hardware implementation. It is changed as required for implementations with incompatible diagnostic and/or system dependent software functionality.
HARDWARE REVISION NUMBER	- Bits 12..15 of the module's hversion. This number is changed for a transparent hardware update.
SOFTWARE MODEL NUMBER	- Bits 4..23 of the module's sversion. This number specifies the software interface. It is changed as required for implementations with incompatible generic software capabilities.
SOFTWARE REVISION NUMBER	- Bits 0..3 of the module's sversion. This number is changed when an enhanced architectural feature is added to the module.
SOFTWARE OPTION NUMBER	- Bits 24..31 of the module's sversion. This field is defined differently for each module type.
HARD PHYSICAL ADDRESS	- This is the 4-Kbyte address range in the I/O address space that the module responds to.
BUS IDENTIFICATION NUMBER	- Bits 4..13 of the module's HPA. This is called the flex field and it identifies the bus to which the module is connected.

Error Messages and Warnings

This section gives a complete list of the error messages that may be generated by SYSMAP, listed in numerical order. Other diagnostic messages may appear at any time. Messages that appear without the (SYSMAP #) trailer are generated by the Online Diagnostic Subsystem or the operating system. Listed below each message are a probable cause and a recommended action statement. The actual cause and action may differ from this list, depending upon the particular circumstances of a given situation. The "!" indicates that a parameter of some sort will replace the exclamation point when the message is displayed.

5000	THIS COMMAND IS INVALID. TYPE HELP FOR ASSISTANCE. (SYSMAPERR 5000)
CAUSE	The command entered by the user is not a valid command recognized by SYSMAP.
ACTION	Make sure that the command that is being typed is a valid command. If in doubt, type HELP to confirm the validity of the command.
5001	THIS PDEV IS NOT VALID FOR THE LEVEL BEING PROCESSED. (SYSMAPERR 5001)
CAUSE	Indicates that while in the step mode the user entered an invalid physical address element.
ACTION	Use the SAME command in step mode to determine the valid physical elements available at the current level. Enter one of the physical address elements available to traverse to the next level. The TOP and PREVIOUS commands can also be used at this time.
5002	THE ENTERED PDEV ENTRY IS TOO LONG. (SYSMAPERR 5002)
CAUSE	Indicates that the length of the entered pdev exceeds the valid length of a valid pdev.
ACTION	Make sure that all pdev value does not exceed the maximum length of 30 characters.
5004	INFORMATION ABOUT DEVICES OF THIS CLASS TYPE IS NOT AVAILABLE. (SYSMAPERR 5004)
CAUSE	This means that SYSMAP was asked to give information about a device class type it has no knowledge of.
ACTION	Type HELP while in the class map mode to determine all valid device classes. SYSMAP can only give information about these particular devices.
5005	SYSMAP COULD NOT FIND ANY INFORMATION OF THE SPECIFIED DEVICE CLASS IN THE IO TABLE. (SYSMAPERR 5005)
CAUSE	This means that SYSMAP searched the I/O configuration table and did not find any information about devices of the given class type.
ACTION	Devices of this class are not available on the system and shouldn't be asked for.

5006	A NUMERIC INPUT IS EXPECTED IN RESPONSE TO THIS PROMPT. (SYSMAPERR 5006)
CAUSE	User entered a non-numeric character in response to the prompt in SYSMAP's timeout command.
ACTION	User must enter a numeric input in response to the timeout prompt.
5007	SYSMAP COULD NOT RETRIEVE MESSAGES FROM ITS MESSAGE CATALOG. (SYSMAPERR 5007)
CAUSE	SYSMAP's initialization process was not successful and was not able to read messages from its catalog.
ACTION	Contact your HP support personnel.
5008	THIS MODE OF CONFIRM IS ALREADY INVOKED. (SYSMAPERR 5008)
CAUSE	User attempted to go into or out of a confirm mode already invoked.
ACTION	
5009	NO LEVEL PREVIOUS TO THIS ONE. ALREADY AT THE TOP. (SYSMAPERR 5009)
CAUSE	This happens when a user in SYSMAP's step mode tries to go to a level of mapping previous to the top level.
ACTION	
5010	SYSMAP ENCOUNTERED AN ERROR WHILE INTERPRETING A COMMAND (SYSMAPERR 5010)
CAUSE	This could only happen if a user entered a valid command and SYSMAP encountered an internal error of some kind that caused it not to recognize the command.
ACTION	Contact your HP support personnel.
5011	SYSMAP ENCOUNTERED AN ERROR WHILE TRYING TO CONFIRM THE PRESENCE OF HARDWARE MODULES. (SYSMAPERR 5011)
CAUSE	Indicates that SYSMAP encountered an error other than a timeout, while trying to communicate with a device.
ACTION	Contact your HP support personnel.
5012	SYSMAP ENCOUNTERED AN ERROR WHILE READING AN INTERNAL DATA BUFFER. (SYSMAPERR 5012)
CAUSE	Indicates that an internal error occurred while moving data into or out of a data buffer.
ACTION	Contact your HP support personnel.

5013	SYSMAP ENCOUNTERED A TIME OUT ERROR WHILE TRYING TO PERFORM A ! COMMAND TO PDEV ! (SYSMAPERR 5013)
CAUSE	Indicates that a timeout error occurred while SYSMAP was trying to accomplish an identify or a roll call command to a device.
ACTION	
5014	USER LACKS THE CAPABILITY REQUIRED TO INVOKE THE CONFIRM MODE (SYSMAPERR 5014)
CAUSE	This means that a user with insufficient capabilities tried to invoke the confirm mode. A user with any capability greater than 2 may not invoke the confirm mode.
ACTION	
5015	SYSMAP ENCOUNTERED AN ERROR WHILE PROCESSING A USER INTERRUPT. (SYSMAPERR 5015)
CAUSE	This will happen only if an internal error occurs when the user tries to suspend SYSMAP.
ACTION	Contact your HP support personnel.
5016	INCORRECT USE OF THIS COMMAND. TYPE HELP FOR ASSISTANCE. (SYSMAPERR 5016)
CAUSE	Indicates that the user typed in a valid command out of context (e.g., IOMAP > on).
ACTION	Although "on" is a valid command, it must be used with the CONFIRM command. Check to see that SYSMAP's commands are used correctly.
5017	SYSMAP ENCOUNTERED AN ERROR WHILE TRYING TO DECODE IDENTIFY DATA. (SYSMAPERR 5017)
CAUSE	Indicates an internal error occurred while SYSMAP was trying to decode identify data.
ACTION	Contact your HP support personnel.
5018	SYSMAP ENCOUNTERED AN ERROR WHILE OBTAINING A DATA BUFFER. (SYSMAPERR 5018)
CAUSE	SYSMAP encountered an error trying to create space for a data buffer.
ACTION	Contact your HP support personnel.
5019	SYSMAP ENCOUNTERED AN ERROR WHILE RETURNING A DATA BUFFER. (SYSMAPERR 5019)
CAUSE	SYSMAP encountered an error trying to return space from a data buffer.
ACTION	Contact your HP support personnel.

5020	SYSMAP ENCOUNTERED AN ERROR WHILE TRYING TO GET THE LDEV OF DEVICE WITH THE FOLLOWING PDEV (!). (SYSMAPERR 5020)
CAUSE	Indicates that an error occurred while SYSMAP was attempting to get the ldev of the device with the given pdev.
ACTION	Contact your HP support personnel.
5021	SYSMAP ENCOUNTERED AN ERROR WHILE TRYING TO RETRIEVE ALL OR SOME OF THE INFORMATION REGARDING DEVICE WITH THE FOLLOWING PDEV(!) (SYSMAPERR 5021).
CAUSE	An error occurred while SYSMAP was attempting to get information from the system tables about a device with the given pdev.
ACTION	Contact your HP support personnel.
5022	SYSMAP WAS UNABLE TO OBTAIN A PRODUCT NUMBER FOR DEVICE WITH THE FOLLOWING PDEV(!) (SYSMAPERR 5022).
CAUSE	An error occurred while SYSMAP was attempting to get the product number of the device with the given pdev.
ACTION	Contact your HP support personnel.
5023	WARNING, BE ADVISED THAT CONTROL_Y MAY NOT BE INVOKED AT THIS TIME. (SYSMAPERR 5023).
CAUSE	This only happens when SYSMAP is running while another process is suspended. When this happens control.y is not active.
ACTION	If the user wishes to invoke SYSMAP with control.y active, SYSMAP must be run without any other diagnostic suspended.
5024	ALL COMMAND IS NOT VALID FROM THIS LEVEL. (SYSMAPERR 5024)
CAUSE	2 This happens when the user is in the step mode of IOMAP and is at the lowest level of mapping possible. If at this level the user issues the ALL command, this error message will appear.
ACTION	
5026	SYSMAP RECEIVED MORE INPUT THAN EXPECTED. EXTRA IGNORED (SYSMAPERR 5026)
CAUSE	
ACTION	
5027	LDEV NUMBERS ARE NOT REQUIRED AS RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5028)
CAUSE	User tried to give SYSMAP a logical device number as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.

3-22 System Map

5028	PDEV NUMBERS ARE NOT REQUIRED AS RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5028)
CAUSE	User tried to give SYSMAP a physical device number as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5029	SECTIONS ARE NOT REQUIRED AS RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5029)
CAUSE	User tried to give SYSMAP a section number as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5030	STEP ARE NOT REQUIRED AS RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5030)
CAUSE	User tried to give SYSMAP a step number as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5031	LOOP COUNT IS NOT APPLICABLE AS A RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5031)
CAUSE	User tried to give SYSMAP a loop count as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5032	ERROR ONLY IS NOT APPLICABLE AS A RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5032)
CAUSE	User tried to give SYSMAP the error only command as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5033	ERROR COUNT IS NOT APPLICABLE AS A RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5033)
CAUSE	User tried to give SYSMAP an error count as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5034	ERROR PAUSE IS NOT APPLICABLE AS A RUN TIME PARAMETERS FOR SYSMAP (SYSMAPWARN 5034)
CAUSE	User tried to give SYSMAP an error pause as a run time parameter.
ACTION	SYSMAP does not need any run time parameters.
5035	TIMEOUT VALUE GIVEN NOT VALID. VALUE MUST BE GREATER THAN ZERO (SYSMAPWARN 5035)
CAUSE	User entered a timeout value less than or equal to zero.
ACTION	Enter a timeout value greater than zero.

5036	UNEXPECTED ERROR OCCURRED WHILE READING THE SYSTEM CONFIGURATION (SYSMAPWARN 5035)
CAUSE	Internal error occurred while SYSMAP was reading the configuration table.
ACTION	Contact your HP support personnel.

3-24 System Map



Contents

4. Terminal Diagnostic Support Monitor	
Introduction	4-1
Defects and Enhancements	4-1
Minimum Configuration	4-1
Operating Instructions	4-3
RUN Command	4-3
Test Execution	4-4
Commands	4-5
COMMENT	4-6
DIAG	4-7
Selftest DTC	4-8
Print to Port	4-13
Internal Port Loopback	4-16
External Port Loopback	4-18
Terminal Write/Read	4-20
DTC	4-23
DUMP	4-24
Logical Device Number	4-26
DTC	4-28
Port	4-31
EXIT command	4-33
HELP	4-34
RESET	4-36
Logical Device Number	4-37
Port	4-38
SIC	4-40
DTC	4-42
STATUS	4-44
DTC	4-45
Port	4-48
Error Messages	4-52

Terminal Diagnostic Support Monitor

Introduction

TERMDSM (Terminal Diagnostic/Support Monitor) is a utility providing various diagnostic services for terminal and serial printer connections on systems of the HP 3000 900 Series computer family. It is patterned after the MPE V utility of the same name. Typically, TERMDSM is used when a problem is suspected with one or more asynchronous devices; the problem may be related to hardware or software.

The purpose of TERMDSM is to gather information on ASC software problems and troubleshoot the DTC hardware. TERMDSM can also be used to test the hardware connections between devices and the DTC. The primary differences between MPE V TERMDSM and MPE XL TERMDSM result from the differing hardware between the two families of HP 3000 systems.

Defects and Enhancements

Submit defect reports and enhancement requests for this diagnostic through the STARS database referencing Product Number 30600-10022.

Minimum Configuration

The following hardware is required to run TERMDSM:

- The minimal mainframe configuration of an HP 3000 Series 900.
- The system and each DTC connected as nodes on the LAN.
- At least one pair of serial interface/connector cards installed on each DTC, with at least one port of each connector card connected to an asynchronous device.
- The following loopback connectors, supplied with each DTC:
 - 25-pin connectors for modem cards (30146-60002)
 - 3-pin connectors for RS-232-C direct-connect cards (30148-60002)
 - 5-pin connectors for RS-422 direct-connect cards (30147-60002)

The software required to use TERMDSM is already included with each MPE XL system. In order for TERMDSM to run, it is necessary that the MPE XL operating system is functioning properly.

The functions of TERMDSM commands are divided into three different security levels. Each security level is associated with the user capabilities that are necessary to use commands of that level. These security levels correspond to the Online Diagnostic Subsystem (also known as SYSDIAG, SYStem DIAGnostics) security levels.

When a user issues a command, TERMDSM checks the user's capability list to see if the user is authorized to use commands of that security level.

User Capabilities	Security Level
SM or DI	0
SM or DI or OP	1
SM or DI or OP or AM	2

Note Users without any of the above capabilities will not be permitted to use TERMDSM.



If the capabilities of the user are insufficient, the command is not executed and an error message is displayed on the user's terminal.

```
Your security level (3) is too low to use TERMDSM. Please
log on with level 0, 1, or 2 security. (TDSM 17)
For level 0, you need SM or DI capability.
For Level 1, you need SM or DI or OP capability.
For Level 2, you need SM or DI or OP or AM capability.
```

Caution If more than one user is running TERMDSM, it is not advisable for one user of TERMDSM to reset another user of TERMDSM. (The user's port, ldev, SIC, or DTC should not be reset.) Such resets may cause the system to crash.



Operating Instructions

TERMDSM is incorporated into the Online Diagnostic Subsystem and is a command-based utility which allows you to monitor and troubleshoot the DTS. It is set up so that you can monitor the condition of a single device or the entire DTC. TERMDSM provides several specific functions. Some commands provide only information, such as listing all the DTCs associated with your system, or the internal configuration information of any DTC. If problems are suspected, TERMDSM may be used to test any DTC and its connections to devices.

If a problem is complex enough that the state of the port and/or DTC subsystem needs to be looked at by an HP representative, the subsystem information may be copied to a disc file and sent to HP. If it has been determined that a device's connection is unusable, TERMDSM is capable of aborting the session on the port so that a new session may be started on the device. This is also true if the connections of several devices are unusable, or if all of the connections of the devices of a DTC are unusable.

RUN Command

To bring up the Online Diagnostic subsystem, enter the following command to the MPE XL 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 Section of this manual for details.

To run the diagnostic, enter:

```
DUI>RUN TERMDSM
      |
      |
      | no parameters required to
      | load test suite
```

Test Execution

When TERMDSM is executed, the following welcome message will be displayed:

```
*****
****                                     ***
****                                TERMDSM                                ***
****                                     ***
****      (C) Copyright Hewlett Packard Co. 1987      ***
****                        All Rights Reserved.      ***
****                        Version A.00.00.013        ***
****                                     ***
****                        Catalog AM.027             ***
****                                     ***
*****
```

Welcome, Today is MON, May 22, 1987, 9:00 AM

Your system has these Distributed Terminal Controllers:

DTC Number	Network Address	Name
1	080009001808	MARKETING
2	080009001234	TIMESHAREROOM

Use this number as the "DTC number" in TERMDSM commands.

Comment DIag DTc DUmP Help Reset SStatus EXIt?

The last line is the TERMDSM prompt. Notice that the prompt lists all the possible commands that you may enter. The capital letters listed are the abbreviations for each command.

You may stop the looping of a diagnostic function by entering **(CTRL) Y** (looping is terminated after the current loop is completed). To stop using TERMDSM, enter the EXIT command. The DUI> prompt will be displayed. Enter EXIT again to return to the : prompt of the system Command Interpreter.



Commands

TERMDSM commands are listed below.

Caution



Some TERMDSM commands abort sessions and cause the loss of data connected with those sessions. If used incorrectly, the commands can unnecessarily disrupt or abort user sessions which are not having problems.

COMMENT	is a utility command that allows you to type in informational comments to use as reference later. The security level of this command is 2.
DIAG	is a command which allows several diagnostic functions to be run on a port. It also allows the user to initiate a DTC selftest. The security level of this command is 0 for the DTC selftest and 1 for individual port tests.
DTC	is an informational command that lists each DTC associated with your system along with its descriptive name and the IEEE 802.3 station address. The security level of this command is 2.
DUMP	is a command which allows the data area of a port to be copied to a disc file for troubleshooting use by HP factory personnel. The security level of this function is 1. The data area of an entire DTC may also be copied to a disc file, and the security level of this function is 0.
EXIT	allows the user to exit the TERMDSM program and return to the DUI> prompt of the Online Diagnostic Subsystem. The security level of this command is 2.
HELP	is a command that displays information regarding TERMDSM and its commands. The security level of this command is 2.
RESET	allows a single port, several ports, or a DTC to accept new sessions. The security level of resetting one or several ports is 1, and the security level of resetting an entire DTC is 0.
STATUS	is an informational command which lists the node name of a specific device's DTC, whether or not the port has a session running on it, and whether or not the device is connected to an interface card capable of modem connections. The security level of this command is 2.

COMMENT

This command is used to make informational comments regarding what you are currently doing in TERMDSM. If you are keeping a hardcopy record of everything that you are typing in at your terminal, you will want to type in any information that you feel will be useful to you later.

This command may be entered at any time while you are in TERMDSM.

The easiest way to keep a hardcopy record of everything you are typing at your terminal is to type in DUT's HARDCOPY command *before* you enter TERMDSM. This command then records everything that appears on your screen.

Another way to do this is to connect a printer to the second port of your terminal and record each line that you type by using the LOG BOTTOM function of your terminal. Refer to your terminal reference manual for more information.

Note



Even though Comment does not appear in the prompt of the Diag, Dump, Reset, and Status commands, a comment is acceptable input in these cases as well.

Security level: 2.

Syntax:

```
C[OMMENT] commenttext
```

Parameter:

commenttext: Any string of alphanumeric characters up to 72 characters long.

DIAG

Determines if the DTC hardware or the connections from a DTC to its devices are functioning properly.

Security level: 0 for DTC selftest; 1 for remaining diagnostic functions.

Syntax:

DI[AG]

Output:

After the DIAG command is entered, the following prompt is printed on your terminal screen:

(diagnostics)

```
SElftest #      (DTC Selftest)
PRint  #, #, # (Print to specified port)
Internal #, #, # (Internal loopback on specified port)
EXTernal #, #, # (Loopback through hood on specified port)
Terminal #, #, # (Write and read to terminal on that port)
```

(carriage return to exit) ?

These are the subcommands that may then be entered. The following pages describe each of these options.

If you enter **Return** instead of one of the subcommands, you will be returned to the major prompt and may then enter another TERMDSM command.

CTRLY can be used during execution of any of the above subcommands to stop the diagnostic looping. However, the current loop will always complete before the **CTRL**Y takes effect. In the case of the DTC selftest, this may take several minutes.

Selftest DTC

Determines if the hardware of a particular DTC is functioning properly.

Security level: 0.

Syntax:

```
SE[LFTEST] dtcnum
```

Parameter:

dtcnum The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.

Range: 1 - *n* , where *n* is the total number of DTCs configured on your system.

Caution



This subcommand aborts all sessions on the DTC under test. Data for those sessions may be lost.

After entering the selftest subcommand, the following is printed on your terminal screen:

Output:

```
Warning: This will reset all users on that DTC.  
As many as 48 terminal users could be affected.  
(TDSM 44) Do you want to continue (yes/no)?
```

If you answer no to this question, you will be returned to the diagnostics prompt. If you answer yes, the following will be printed on your terminal screen:

```
How many times do you want to repeat this operation?
```

You should then enter how many times you want to run the DTC selftest. Enter a number between 1 and 65535. The following question is then printed on your terminal screen:

```
Do you want to stop looping if an error is found (yes/no)?
```

If you answer yes, the series of selftests will stop if an error is encountered. For example, you specified you want the selftest to run 10 times and an error is encountered in the third selftest. If you answered yes to this last question, the remaining seven selftests will not be run. If you answered no, all 10 selftests will be executed regardless of the results of each test.

The following is then printed on your terminal screen:

Each DTC self test can take as long as 7 minutes. Enter a control-y to stop the self test loop (the control-y will take effect when the current self test finishes).

Anytime you want to stop the diagnostic, press **CTRL**Y. After the current loop has executed, the diagnostics prompt will appear and you may enter another subcommand.

The DTC Selftest subcommand causes a complete DTC selftest to be run. This selftest includes the selftests for each of the interface cards within the DTC, and a download of the DTC across the LAN. Any jobs or sessions that are running on devices connected to the DTC are aborted.

Note



Each loop of the DTC Selftest subcommand is functionally identical to the RESET DTC subcommand. The inclusion of both commands is for your convenience, and to provide looping capability in the event an intermittent hardware failure is suspected.

As each selftest completes, the results of each SIC selftest will be printed on your terminal screen in the format:

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
n	type	status	pppp pppp

Where:

n Number of the interface card within the DTC.

type The type of card in the slot. These are the possible values and their meanings:

RS-232 Direct	RS-232-C direct-connect card
RS-232 Modem	modem card
RS-422 Direct	RS-422 direct-connect card
(not installed)	no card in the slot

status The results of the card selftest. These are the possible values and their meanings:

OK	selftest passed
MAIN CIRCUITRY BAD	card hardware failure
ROM CRC BAD	ROM CRC hardware error
ALL PORTS BAD	failure on all ports
PORT 0 BAD	failure on port zero (other ports may also have failed)
SOME PORTS BAD	failure on more than one port but not all (port zero okay)

pppp pppp The number of ports on the card is shown by the number of characters in this string.
Six characters represents a modem card. Eight characters represents a direct connect
card. A G means the port is good; an x means that the port is bad.

The results of the selftest are printed with SIC #5 at the top, followed by #'s 4,3, ... 0. This corresponds with the physical arrangement of the boards in the DTC.

When all selftests have completed, the following will be displayed:

```
The DTC self test passed nnnnnnn times and failed
mmmmmmmm times.
```

Where nnnnnnn and mmmmmmm are the total number of times the selftest passed or failed respectively.

After all selftests complete, the DTC software is downloaded to the DTC so that users may again use devices connected to the DTC. You will receive messages to inform you of each download if you are running TERMDSM from the console.

You will then be returned to the diagnostics prompt.

Example:

```
(diagnostics)
```

```
SElftest #      (DTC Selftest)
PRint  #,#,# (Print to specified port)
Internal #,#,# (Internal loopback on specified port)
EXternal #,#,# (Loopback through hood on specified port)
Terminal #,#,# (Write and read to terminal on that port)
```

```
(carriage return to exit) ?se 2
```

```
Warning: This will reset all users on that DTC. As many as 48 terminal users
could be affected. (TDSM 44) Do you want to continue (yes/no)? y
```

```
How many times do you want to repeat this operation? n
```

```
Do you want to stop looping if an error is found (yes/no)? n
```

```
Each DTC self test can take as long as 7 minutes. Enter a
control-y to stop the self test loop (the control-y will
take effect when the current self test finishes).
```

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
5	(not installed)		
4	(not installed)		
3	(not installed)		
2	RS-232 Direct	OK	GGGG GGGG
1	RS-232 Direct	OK	GGGG GGGG
0	RS-232 Modem	OK	GGGG GG

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
5	(not installed)		
4	(not installed)		
3	(not installed)		
2	RS-232 Direct	OK	GGGG GGGG
1	RS-232 Direct	OK	GGGG GGGG
0	RS-232 Modem	OK	GGGG GG

The test passed 2 times and failed 0 times.

(diagnostics)

SElftest # (DTC Selftest)
 PPrint #,#,# (Print to specified port)
 Internal #,#,# (Internal loopback on specified port)
 EXternal #,#,# (Loopback through hood on specified port)
 Terminal #,#,# (Write and read to terminal on that port)

(carriage return to exit) ?

In the above example, DTC number two is selftested twice. The selftest passes both times.

Print to Port

Determines if the connection between a DTC and one of its devices is functioning properly by printing to the device.

Security level: 1.

Syntax:

`PR[INT] dtcnum,cardnum,portnum`

Parameters:

<i>dtcnum</i>	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
<i>cardnum</i>	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
<i>portnum</i>	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections 0 - 5 for modem connections

Caution



This subcommand aborts the session at the terminal under test. Data for that session is lost.

This diagnostic is supplied primarily for use with devices that do not support loopback tests, such as serial printers, or unsupported terminals.

Output:

When the subcommand is entered you will be asked:

Do you want to stop looping if an error is found (yes/no)?

If you answer yes, the series of print operations will stop if an error is encountered. For example, you specified you want the test to run 10 times and an error is encountered in the third printing. If you answered yes to this last question, the remaining seven tests will not be run. If you answered no, all 10 print operations will be executed regardless of the results of each test. The following is then printed on your terminal screen:

Please connect and make ready (power on, remote mode, etc.)
a supported printer (or terminal), before answering the
"How many times?" message. (TDSM 115)

How many times do you want to repeat this operation?

You should then enter how many times you want to run the print operation. Enter a number between 1 and 65535. Then the following is displayed:

Type CONTROL_Y to stop the looping.

Any time that you want to stop the diagnostic, press **CTRL**Y. The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

The Print to Port subcommand sends the following string of characters to the device:

ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_abcdefghijklmnopqrstuvwxyz

If TERMDASM knows that this test has failed, an error message will be printed listing the reason for the failure and how many milliseconds the test ran before failing.

Note

If the test passes according to TERMDASM, then it is your responsibility to inspect the output of the test to see if it matches the string of characters listed above. TERMDASM cannot know the final outcome of the test; therefore, the only way to be sure that the test was successful is through visual inspection.

The diagnostics prompt will be printed on your terminal screen in order to allow you to enter another subcommand.

Example:

```
(diagnostics)

SElftest #      (DTC Selftest)
PRint  #, #, # (Print to specified port)
Internal #, #, # (Internal loopback on specified port)
EXternal #, #, # (Loopback through hood on specified port)
Terminal #, #, # (Write and read to terminal on that port)

(carriage return to exit) ?PR 2,3,1
Do you want to stop looping if an error is found (yes/no)? Y
Please connect and make ready (power on, remote mode, etc.)
a supported printer (or terminal), before answering the
"How many times?" message.      (TDSM 115)
How many times do you want to repeat this operation? 10

The test passed 10 times and failed 0 times.

(diagnostics)

SElftest #      (DTC Selftest)
PRint  #, #, # (Print to specified port)
Internal #, #, # (Internal loopback on specified port)
EXternal #, #, # (Loopback through hood on specified port)
Terminal #, #, # (Write and read to terminal on that port)

(carriage return to exit) ?
```

The Print to Port subcommand above prints the character string to the second port of the fourth card in DTC number two. (Remember that ports and cards are numbered starting with zero.) The test was successful all ten times that it ran *according to TERMDSM*. You must visually check the output of all ten tests to make sure that the prints were successful.

Internal Port Loopback

Determines if a specific port is functioning properly without disconnecting the device from the port. Does a loopback within the SIC circuitry devoted to that port.

Security level: 1.

Syntax:

```
I[TERNAL] dtcnum,cardnum,portnum''
```

Parameters:

<i>dtcnum</i>	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
<i>cardnum</i>	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
<i>portnum</i>	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections 0 - 5 for modem connections

This diagnostic cannot test all the functions that are possible with a hood (loopback connector), since the DTC connection itself cannot be tested without a hood. However, this diagnostic has the advantage that no cables need to be disconnected from or additional equipment connected to the port.

Output:

When the subcommand is entered you will be asked:

Do you want to stop looping if an error is found (yes/no)?

If you answer yes, the series of loopback tests will stop if an error is encountered. For example, you specified you want the test to run 10 times and an error is encountered in the third test. If you answered yes to the above question, the remaining seven loopback tests will not be run. If you answered no, all 10 loopback tests will be executed regardless of the results of each test.

The following question is then printed on your terminal screen:

How many times so you want to repeat this operation?

You should then enter how many times you want to run this diagnostic function. Enter a number between 1 and 65535. The following is then displayed:

Type CONTROL_Y to stop the looping.

Any time that you want to stop the diagnostic, press **CTRL**Y. The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

4-16 Terminal Diagnostic Support Monitor

If TERMDASM knows that this test has failed, an error message will be printed listing the reason for the failure and how many milliseconds the test ran before failing.

The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

Example:

```
(diagnostics)

SElftest #      (DTC Selftest)
PRint      #,#,# (Print to specified port)
Internal #,#,# (Internal loopback on specified port)
EXternal #,#,# (Loopback through hood on specified port)
Terminal #,#,# (Write and read to terminal on that port)

(carriage return to exit) ?I 3,2,5
Do you want to stop looping if an error is found (yes/no)? y
How many times do you want to repeat this operation? 4
Type CONTROL_Y to stop the looping.

The test passed 10 times and failed 0 times.

(diagnostics)

SElftest #      (DTC Selftest)
PRint      #,#,# (Print to specified port)
Internal #,#,# (Internal loopback on specified port)
EXternal #,#,# (Loopback through hood on specified port)
Terminal #,#,# (Write and read to terminal on that port)

(carriage return to exit) ?
```

The example Internal Port subcommand above executes the internal loopback test four times on the sixth port of the third card of DTC number three. (Remember that ports and cards are numbered starting with zero.) There will be no output to the terminal under test.

External Port Loopback

Determines if a specific port is functioning properly, including the connector. Sends data out and reads it immediately, verifying that the data received is exactly what was sent.

Security level: 1.

Syntax:

EXT[ERNAL] dtcnum,cardnum,portnum

Parameters:

dtnum	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
cardnum	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
portnum	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections Range: 0 - 5 for modem connections

Output:

When this subcommand is entered, the following question is displayed on your terminal screen:

Do you want to stop looping if an error is found (yes/no)?

If you answer yes, the series of loopback tests will stop if an error is encountered. For example, you specified you want the loopback test to run 10 times and an error is encountered in the third test. If you answered yes to the above question, the remaining seven tests will not be run. If you answered no, all 10 loopback tests will be executed regardless of the results of each test.

The following is then printed on your terminal screen:

**Please put a loopback hood on the Port, before answering
the "How many times?" question. (TDSM 113)**

How many times do you want to repeat this operation?

At this time, go to the DTC and connect a loopback hood to the port. Then return and enter how many times you want to run this diagnostic function. Enter a number between 1 and 65535. The following is then displayed:

Type CONTROL_Y to stop the looping.

Any time that you want to stop the diagnostic, press **CTRL Y**. The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

4-18 Terminal Diagnostic Support Monitor

If TERMDSM knows that this test has failed, an error message will be printed listing the reason for the failure and how many milliseconds the test ran before failing.

The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

Example:

```
(diagnostics)

SElftest #      (DTC Selftest)
PRint  ### (Print to specified port)
Internal ### (Internal loopback on specified port)
EXternal ### (Loopback through hood on specified port)
Terminal ### (Write and read to terminal on that port)

(carriage return to exit) ?  ext 3,2,5

Do you want to stop looping if an error is found (yes/no)? y
Please put a loopback hood on the Port, before answering
the "How many times?" question.      (TDSM 113)

How many times do you want to repeat this operation?  5
Type CONTROL_Y to stop the looping.
Read timeout (no data came back).  Time = 16950 msec.  If External loopback,
check loopback hood.  If Terminal loopback, check connections, terminal
configuration, etc.      (TDSM 229)

The test passed 0 times and failed 1 time.

(diagnostics)

SElftest #      (DTC Selftest)
PRint  ### (Print to specified port)
Internal ### (Internal loopback on specified port)
EXternal ### (Loopback through hood on specified port)
Terminal ### (Write and read to terminal on that port)

(carriage return to exit) ?
```

The example External Port subcommand should execute four internal loopback tests on the sixth port of the third card of DTC number three. (Remember that ports and cards are numbered starting with zero.) If the first loopback fails, the other four are not executed because "Stop on Error" was probably requested. The error listed here means that the loopback connector was not attached to the port, or the port is malfunctioning. The user should verify the proper attachment of the loopback hood and run the test again. If the loopback connector is attached correctly to the correct port, the error indicates that the port is defective.

Terminal Write/Read

Determines if the connection between a DTC port and a terminal is functioning properly. TERMDSM writes a line of characters to the terminal and orders it to send that line back. TERMDSM then compares the characters received with those sent.

Security level: 1.

Syntax:

T[ERMINAL] dtcnum,cardnum,portnum

Parameters:

<i>dtcnum</i>	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
<i>cardnum</i>	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
<i>portnum</i>	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections 0 - 5 for modem connections

Output:

When the subcommand is entered the following question is asked:

Do you want to stop looping if an error is found (yes/no)?

If you answer yes, the series of diagnostics will stop if an error is encountered. For example, you specified you want the write/read operation to run 10 times and an error is encountered in the third test. If you answered yes to the above question, the remaining seven write/read operations will not be run. If you answered no, all 10 write/read operations will be executed regardless of the results of each test.

The following is then printed on your terminal screen:

Please connect and make ready (power on, remote mode, etc.)
a supported terminal, before answering the "How many times?"
question. (TDSM 114)

How many times do you want to repeat this operation?

You should prepare the terminal and then enter how many times you want to run this diagnostic function. Enter a number between 1 and 65535. The following is then displayed:

Type CONTROL_Y to stop the looping.

Any time that you want to stop the diagnostic, press **CTRL**Y. The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

4-20 Terminal Diagnostic Support Monitor



This diagnostic function tests the connection between the DTC port up to and including some terminal functions. (It is not a thorough terminal test.) It does this by writing a string of characters to the terminal and ordering the terminal to send those characters back for comparison. The string of characters sent by TERMDSM is preceded by:

`(ESC)H (ESC)J`

which homes the cursor on the screen and clears the screen. Then the character string:

`ABCDEFGHIJKLMNOPQRSTUVWXYZ[\ ]^_ abcdefghijklmnopqrstuvwxyz`

is written to the terminal, followed by:

`(ESC)H (ESC)d`

which homes the cursor and causes the terminal to return the character string. (Sending `(ESC)d` to the terminal is the same as pressing `(Enter)`.)

If TERMDSM knows that this test has failed, an error message will be printed listing the reason for the failure and how many milliseconds the test ran before failing.

The diagnostics prompt will be printed and you will be allowed to enter another subcommand.

Example:

(diagnostics)

```
Selftest #      (DTC Selftest)
Print   #,#,# (Print to specified port)
Internal #,#,# (Internal loopback on specified port)
EXternal #,#,# (Loopback through hood on specified port)
Terminal #,#,# (Write and read to terminal on that port)
```

(carriage return to exit) ? t 3,2,5

Do you want to stop looping if an error is found (yes/no)? y

Please connect and make ready (power on, remote mode, etc.)

a supported terminal, before answering the "How many times?" question. (TDSM 114)

How many times do you want to repeat this operation? 100

Type CONTROL_Y to stop the looping.

The test passed 100 times and failed 0 times.

(diagnostics)

```
SElftest #      (DTC Selftest)
PRint  #, #, # (Print to specified port)
Internal #, #, # (Internal loopback on specified port)
EXTErnal #, #, # (Loopback through hood on specified port)
Terminal #, #, # (Write and read to terminal on that port)
```

(carriage return to exit) ?

The example above executes 100 terminal write/read tests on the sixth port of the third card of DTC number three. (Remember that ports and cards are numbered starting with zero.)

DTC

Lists each DTC assigned to your system and displays the number that TERMDSM has assigned to each DTC.

Security level: 2.

Syntax:

DT[C]

Output:

When this command is entered, information similar to the following is displayed on your terminal:

Your system has these Distributed Terminal Controllers:

DTC Number	Network Address	Name
-----	-----	-----
1	08000900180A	BIOLAB
2	080009000808	MARKETING
3	080009001234	TIMESHAREROOM

Use this number as the "DTC number" in TERMDSM commands.

The DTC Node Name that was associated with each DTC at configuration time is displayed along with the IEEE 802.3 station address of each DTC. The number in the first column is assigned by TERMDSM and these numbers must be used in other TERMDSM commands to reference particular DTCs.

After the above display is printed, the major prompt will be printed and you may enter another TERMDSM command.

If, during a TERMDSM command, you make an error in specifying a DTC number, the above display is automatically printed to your terminal for your convenience.

Note



The listing of DTCs with an IEEE 802.3 station address and name is not a guarantee that each DTC exists, is installed, is powered on, or is otherwise functional. This information is obtained from the NMCONFIG.PUB.SYS file and is therefore only a reflection of what has been configured (correctly or otherwise).

DUMP

Captures data pertaining to a particular port, DTC, or logical device number and writes it to a disc file.

Security level: 0 for DTC dump; 1 for remaining dump functions.

Syntax:

DU[MP]

Output:

After the Dump command is entered, the following prompt is printed on your terminal screen:

(dump)

Ldev #
P0rt #,#,#
DTc #

(carriage return to exit) ?

These are the subcommands that can then be entered. The following pages describe each of these options.

If you enter **Return** instead of one of the subcommands, you will be returned to the major prompt and may then enter another TERMDSM command.

There are two different types of dump files:

- Information regarding a particular DTC. This information is written to a disc file in binary form. If you need the information in the dump analyzed, send it to your HP representative on magnetic tape. You cannot print this file.
- Information regarding a particular Logical Device Number or physical port. If you need the information in this type of dump analyzed, send it to your HP representative on magnetic tape, or in printed form.

Information written to a disc file for this type
of dump will appear in two forms:

- A hexadecimal listing of all data.
 - The ASCII representation of each byte of that data.
- The file will also contain:
- The day and time of the dump.
 - The type of dump (logical device number or port).
 - Headings before each of the major areas of dump information.

If a dump file for an ldev or port is to be printed, a printer file equation with the CCTL option should be used. In the following example, TRM123AB is the name of the dump file:

```
:FILE LIST;DEV=LP;CCTL  
:PRINT TRM123AB.PUB.SYS,*LIST
```

Note

Complete information describing the situation that necessitated the use of Dump is invaluable in assisting the HP representative's analysis of TERMDSM dumps. You may use the Comment command to help document the situation.

Logical Device Number

Captures data pertaining to a specific logical device number and writes it to a disc file.

Security level: 1.

Syntax:

L[DEV] ldevnum

Parameter:

<i>ldevnum</i>	The logical device number associated with the device whose data you want to dump.
----------------	---

Information associated with the logical device number is dumped to a disc file. Since the ldev is also associated with a port, information pertaining to the port is also dumped. Therefore, if you have identified a terminal problem and dump information using the ldev subcommand, complete information about the terminal connection is dumped; there is no need to also use the port subcommand.

Note



Dumping information pertaining to a logical device number does not reset (or otherwise affect) the associated device. If you need to reset the device, use the Reset command.

Output:

When the dump is complete, the following information will be displayed:

The dump file is TRMdddll.PUB.SYS

where *ddd* is the Julian day of the year and *ll* is a two-letter string in the range:

AA, AB, AC, ... , BA, BB, BC, ... , ZZ

This numbering system allows you to create many dump files in one day and to later know on what day each dump was created by simply looking at the file name.

The dump prompt is then repeated so that you may enter another dump subcommand.

Example:

(dump)

Ldev #
POrt #,#,#
DTc #

(carriage return to exit) ?l 41

The dump file is TRM112AG.PUB.SYS

(dump)

Ldev #
POrt #,#,#
DTc #

(carriage return to exit) ?

Information for ldev 41 is placed in the file TRM112AG.PUB.SYS.

DTC

Captures data pertaining to a particular DTC and writes it to a disc file.

Security level: 0.

Syntax:

DT[C] dtcnum

Parameter:

dtcnum The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.

Range: 1 - *n* where *n* is the total number of DTCs configured on your system.

Caution

This subcommand resets all sessions on the DTC under test.



Output:

When this subcommand is entered, the following message will be printed on your terminal screen:

```
Warning: This will reset all users on that DTC.
As many as 48 terminal users could be affected.      (TDSM 44)
Do you want to continue (yes or no)?
```

If you respond no, the Dump prompt will be repeated for you to enter another Dump subcommand. If you respond yes, the dump continues after the following message:

The DTC dump can take as long as 7 minutes.

The DTC dump information is placed in two files. The first file contains information about the part of the DTS software that resides on the host (the HP 3000). When the dump of this information is complete, the following information will be displayed:

The following two files make up the DTC dump. Please deliver both to
your HP representative: (TDSM 64)

```
Shhhhhh.c.PUB.SYS
TRMdddll.PUB.SYS
```

Where *hhhhh* is the last six hexadecimal digits of the DTC's LAN address and *c* is a character in the range from A to Z. The *ddd* is the Julian day of the year and *ll* is a two-character string in the range:

AA, AB, AC, ... , BA, BB, BC, ... , ZZ

This numbering system allows you to create many dump files in one day and to later know on what day each dump was created by simply looking at the file name.

If you need to send dump information to HP to have it analyzed, make sure that you send both dump files.

Note

If all 26 (suffix A through Z) DTC dump files already exist, one will be overwritten by the current DTC dump. If you are doing a lot of DTC dumps, you may wish to store off or rename some dump files to preserve them.

Example:

(dump)

Ldev #
P0rt #, #, #
DTc #

(carriage return to exit) ?DT 3

Warning: This will reset all users on that DTC.
As many as 48 terminal users could be affected. (TDSM 44)
Do you want to continue (yes/no)?y
The DTC dump can take as long as 7 minutes.

(dump)

Ldev #
P0rt #, #, #
DTc #

(carriage return to exit) ?DT 3

S900180C.PUB.SYS
TRM120AC.PUB.SYS

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
---	-----	-----	-----
5	(not installed)		
4	(not installed)		
3	(not installed)		
2	RS-232 Direct	OK	GGGG GGGG
1	RS-422 Direct	OK	GGGG GGGG
0	RS-232 Modem	OK	GGGG GG

(dump)

Ldev #
P0rt #, #, #
DTc #

(carriage return to exit) ?

In the above example, information regarding DTC number three is dumped to the two files listed. The status of each SIC is listed; for more information on this status, refer to the Reset DTC subcommand.

Port

Captures data pertaining to a particular port and writes it to a disc file.

Security level: 1.

Syntax:

```
PO[RT] dtcnum,cardnum,portnum
```

Parameters:

<i>dtcnum</i>	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
<i>cardnum</i>	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
<i>portnum</i>	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections 0 - 5 for modem connections

Information associated with the port is dumped to a disc file. If the port is also associated with an ldev, information pertaining to the ldev is also dumped. Therefore, if you have identified a terminal problem and dump information using the port subcommand, complete information about the terminal connection is dumped; there is no need to also use the ldev subcommand.

Note



Dumping information pertaining to a port does not reset (or otherwise affect) the associated device. If you need to reset the device, use the Reset command.

Output:

When the dump is complete, the following information will be displayed:

The dump file is TRMdddll.PUB.SYS

Where *ddd* is the Julian day of the year and *ll* is a two-character string in the range:

AA, AB, AC, ... , BA, BB, BC, ... , ZZ

This numbering system allows you to create many dump files in one day and to later know on what day each dump was created by simply looking at the file name.

Example:

```
(dump)

Ldev #
P0rt #,#,#
DTc #

(carriage return to exit) ?po 1,0,2
The dump file is TRM112AG.PUB.SYS
(dump)

Ldev #
P0rt #,#,#
DTc #

(carriage return to exit) ?
```

The dump information for the port whose address is DTC 1, card 0, port 2 is placed in the file TRM112AG.PUB.SYS.

EXIT command

Exits the TERMDSM program and returns you to the DUI prompt.

Security level: 2.

Syntax:

```
EXIT[ ]
```


HELP

Lists information regarding TERMDISM and its commands.

Security level: 2.

Syntax:

H[ELP]

TERMDISM is a utility providing various terminal oriented functions to Customers and HP Service representatives. It is provided to help investigate and recover from problems associated with the HP DTS, Distributed Terminal Subsystem. The DTS is made up of one or more DTCs (Distributed Terminal Controllers), which are units providing terminal connections on one end, and are connected on the other to an HP mainframe via an IEEE 802.3 Local Area Network (LAN) connection. In addition to the DTCs, the subsystem includes the software from the mainframe downloaded to the DTC, and software in the mainframe itself.

One of the functions of TERMDISM is "Dump". This function allows the user to dump the mainframe and DTC software associated with one terminal ("port") on the DTC, or the entire DTC. The dump is written to disc.

TERMDISM may also be used to reset one port, a set of ports on the DTC (one SIC), or the entire DTC, and in each case the associated mainframe software.

TERMDISM may be used to obtain status information from the DTC, either for a particular port, or the entire DTC.

TERMDISM may be used to run hardware diagnostics on a particular port, or the entire DTC. These diagnostics also exercise the LAN between the DTC and mainframe, but are not a LAN diagnostic. Please use the LAN diagnostic for this purpose.

Finally, TERMDISM will provide the user with a list of the DTCs configured on that mainframe, their LAN address, and the user-specified name.

Some commands need you to type in the ldev, port, SIC (Serial Interface Card- a set of ports), or DTC number.

Ldevs are one number ("200" for example).

DTCs are also one number ("2"). For a list of all DTCs on this system, enter DTC in response to the main prompt.

SICs are identified by two numbers, the DTC number first, then the number of the SIC within that DTC. For example, "2,3" means DTC number 2, and SIC number 3 in that DTC.

SICs are numbered from 0, and SIC 0 is in the bottom of the DTC.

Ports are identified with three numbers: the DTC, the SIC within the DTC that has that port, and the port number on that SIC. For example, "4,3,7" means DTC number 4, SIC number three and port 7 on that SIC.

Ports are numbered from 0, and as you face the BACK, port 0 is leftmost. Each prompt asks you for the particular entity (ldev, DTC, SIC, or port), and reminds you of how many parameters are necessary.

For example, the Reset prompt is:

(reset)

Ldev #
Port #,#,#
Sic #,# (multiplexer card in DTC)
Dtc #

To specify ldev 200, enter "L 200".

To specify port 4 of SIC 3 of DTC 1, say "PD 1,3,4"; DTC, then SIC, and finally port.

To specify SIC 2 on DTC 1, say "SI 1,2".

To specify DTC 2, say "DT 2".

Not all commands have all choices. For example, the Status command only has port or DTC. As in all cases, the prompt reminds you which choices are available and how many parameters are necessary for that choice.

Reset and dump commands for a particular port or ldev act upon the ldev and its associated port, regardless of whether the "port" or "ldev" was specified. It is not necessary to specify both an ldev and its port; one or the other is sufficient.

Likewise, reset SIC resets all ldevs associated with that SIC, and reset/dump of a DTC resets all ldevs on that DTC.

After the above information is displayed, the major prompt will be printed and you may enter another TERMDSM command.

RESET

To clear hardware and software associated with a logical device number, physical port, serial interface card, or DTC so that a new session or job may be started on each port.

Security level: 0 for DTC; 1 for remaining functions.

Syntax:

R[ESET]

Output:

After the Reset command is entered, the following prompt is printed on your terminal screen:

```
(reset)

Ldev #
P0rt #,#,#
Sic #,* (multiplexer card in DTC)
Dtc #

(carriage return to exit) ?
```

These are the subcommands that may then be entered. The following pages describe each of these options.

If you enter Return instead of one of the subcommands, you will be returned to the major prompt and may then enter another TERMDSM command.

Some port/ldev problems may not be fixed with ldev or port resets. If a port is still unusable after a reset, consider using reset SIC. Note that this will affect all users (up to 8) on that card. Finally, if the SIC reset is also unsuccessful, then the entire DTC should be reset. Note that this will affect all the users on the DTC (up to 48).

Logical Device Number

Clears the software associated with a particular logical device number (and its associated port) so that the associated device may be used again.

Security level: 1.

Syntax:

L[DEV] ldevnum

Parameter:

ldevnum The logical device number associated with the connection you want to reset.

This subcommand also resets the port associated with this ldev. Also, any session or job running on the device is aborted.

When it is complete, the reset prompt will be displayed and you may then enter another reset subcommand.

Example:

```
(reset)

Ldev #
P0rt #,#,#
S1c #,# (multiplexer card in DTC)
DTc #

(carriage return to exit) ?l 41

(reset)

Ldev #
P0rt #,#,#
S1c #,# (multiplexer card in DTC)
DTc #

(carriage return to exit) ?
```



In the above example, any information associated with the ldev 41 is reset. This includes the physical port (resetting the physical port is not required). Any associated session is aborted.

Port

Clears the software associated with a particular port and its associated ldev so that the associated device may be used again.

Security level: 1.

Syntax:

```
PO[RT] dtcnum,cardnum,portnum
```

Parameters:

<i>dtcnum</i>	The number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This list is also printed when you make a mistake while typing in a DTC number.
<i>cardnum</i>	Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC. Range: 0 - 5
<i>portnum</i>	Number of the port on the interface card. As you face the back of the DTC, the ports are numbered from zero beginning at the left of the board. Range: 0 - 7 for direct connections 0 - 5 for modem connections

This subcommand resets the ldev that is currently associated with the port. It also aborts any session or job that is running on the port.

When it is complete, the reset prompt will be displayed and you may then enter another reset subcommand.

Example:

```
(reset)

Ldev #
Port #,#,#
SIc #,# (multiplexer card in DTC)
DTc #

(carriage return to exit) ?PO 2,3,1
```

(reset)

Ldev #
P0rt #,#,#
SIc #,# (multiplexer card in DTC)
DTc #

(carriage return to exit) ?

The Port subcommand above resets the port with the physical address DTC 2, card 3, port 1. (Remember that ports and cards are numbered starting with zero.) The associated ldev is also reset (explicitly resetting the Ldev is not required). Any associated job or session is aborted.

SIC

Resets all ports of a specific Serial Interface Card in a specific DTC and the associated ldevs.
Security level: 1.

Syntax:

SI[C] dtcnum,cardnum

Parameters:

dtcnum Number of the DTC to which the device is connected. This is the number TERMDSM associates with this DTC.

cardnum Number of the interface card within the DTC to which the device is connected. The cards are numbered from zero beginning at the bottom of the DTC.

Range: 0 - 5.

When the subcommand is entered, all ports of the card are reset. When the reset completes, the following will be displayed:

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
---	-----	-----	-----
n	type	status	pppp pppp

Where:

n Number of the interface card within the DTC.

type= The type of card in the slot. These are the possible values and their meanings:

RS-232 Direct	RS-232-C direct-connect card
RS-232 Modem	modem card
RS-422 Direct	RS-422 direct-connect card
(not installed)	no card in the slot

status The results of the card selftest. These are the possible values and their meanings:

OK	selftest passed
MAIN CIRCUITRY BAD	card hardware failure
ROM CRC BAD	ROM CRC hardware error
ALL PORTS BAD	failure on all ports
PORT 0 BAD	failure on port zero

SOME PORTS BAD (other ports may also have failed)
failure on more than one port but not all (port zero okay)

pppp pppp= The number of ports on the card is shown by the number of characters in this string. Six characters represents a modem card. Eight characters represents a direct connect card. A G means the port is good; an x means that the port is bad.

The reset prompt will then be displayed and you may enter another reset subcommand. This subcommand should also be used when resets of a particular port on an SIC have been unsuccessful. If the reset of the card is also unsuccessful, consider using the DTC subcommand to reset the entire DTC.

Example:

```
(reset)

Ldev #
P0rt #,#,#
Sic #,# (multiplexer card in DTC)
Dtc #

(carriage return to exit) ?si 3,2
```

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
2	RS-232 Modem	OK	GGGG GG

```
(reset)

Ldev #
P0rt #,#,#
Sic #,# (multiplexer card in DTC)
Dtc #

(carriage return to exit) ?
```

The SIC subcommand above resets all ports on the third card of DTC number three. (Remember that cards are numbered starting with zero.) The selftest of the card (a modem card) passed.

DTC

Resets the hardware of a particular DTC.

Security level: 0.

Syntax:

DT[C] dtcnum

Parameter:

<i>dtcnum</i>	The DTC number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. Range: 1 - <i>n</i> , where <i>n</i> is the total number of DTCs configured on your system.
---------------	--

Output:

After entering the DTC subcommand, the following is printed on your terminal screen:

```
Warning: This will reset all users on that DTC. As many as
48 terminal users could be affected. (TDSM 44)
Do you want to continue (yes/no)?
```

If you answer no to this question, you will be returned to the reset prompt. If you answer yes, the reset continues and the following messages are displayed:

```
The reset operation can take as long as 7 minutes.
```

```
The reset operation causes a DTC selftest to occur.
```

The DTC subcommand causes a complete DTC selftest to be run. This selftest includes the selftests for each of the interface cards within the DTC. All jobs or sessions that are running on devices connected to the DTC are aborted.

Note



The Reset DTC subcommand is functionally identical to one Selftest DTC subcommand of the TERMDSM DIAG command. The inclusion of both commands is for your convenience.

If the selftest completes successfully, the results of each of the card selftests will be displayed. Refer to the SIC subcommand for the explanation of these results.

The DTC software is then downloaded to the DTC so that users may again use devices connected to the DTC. If you are running TERMDSM at the system console, you will receive informational messages relating to this download.

You will then be returned to the reset prompt and may enter another reset subcommand.

Example:

(reset)

Ldev #
P0rt #,#,#
Sic #,# (multiplexer card in DTC)
Dtc #

(carriage return to exit) ?DT 2

Warning: This will reset all users on that DTC. As many as 48 terminal users could be affected. (TDSM 44) Do you want to continue (yes/no)? Y
The reset operation can take as long as 7 minutes.
The reset operation causes a DTC selftest to occur.

SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
5	(not installed)		
4	(not installed)		
3	(not installed)		
2	RS-232 Direct	OK	GGGG GGGG
1	RS-232 Direct	OK	GGGG GGGG
0	RS-232 Modem	OK	GGGG GG

(reset)

Ldev #
P0rt #,#,#
Sic #,# (multiplexer card in DTC)
Dtc #

(carriage return to exit) ?

In the above example, DTC number two is reset. The results of the card selftests show that there are three cards in the DTC (in the first three slots) and each of the three passes its selftest. The first and second cards are RS-232-C direct-connect cards and card number 2 is a modem card. Refer to the SIC subcommand for information on how to interpret the results of the card selftest.

STATUS

Obtains information about DTC or port configuration and the state of the DTC or port.

Security level: 2.

Syntax:

ST[ATUS]

Output:

After the Status command is entered, the following prompt is printed on your terminal screen:

(status)

DTC #
Port #, #, #

(carriage return to exit) ?

These are the subcommands that can then be entered. The following pages describe these two options. Neither of the Status subcommands affect the operation of the system, DTCs, or ports.

If you enter **Return** instead of one of the subcommands, you will be returned to the major prompt and then may enter another TERMDSM command.

DTC

Obtains configuration and status information about a particular DTC.

Security level: 2.

Syntax:

DT[C] dtcnum

Parameter:

dtcnum This is the number TERMDSM associates with a particular DTC. The DTC command lists each DTC with its assigned number. This is also printed when you make a mistake while typing in a DTC number.

When this subcommand is entered, a large amount of information related to the DTC is printed on your terminal screen.

Example:

(status)

DTC #

Port #, #, #

(carriage return to exit) ?DT 1

Machine Type = HP2345

DTC Node Name = DTC888888888888

NOVRAM Value :

= 00.00

Nodal address = 88-88-88-88-88-88

Boot Multicast address = 09-00-09-00-00-04
00-46

Host Node Address = 08-00-09-00-03-01

DTC Lan Multicast Addresses :

Probe Primary = 00-00-00-00-00-00

Probe Secondary = 00-00-00-00-00-00

LAN Analyst = 00-00-00-00-00-00



SIC	SIC type	SIC selftest status	Ports (Good or x:bad)
5	(not installed)		
4	(not installed)		
3	(not installed)		
2	RS-232 Direct	OK	GGGG GGGG
1	RS-422 Direct	OK	GGGG GGGG
0	RS-232 Modem	OK	GGGG GG

Version Numbers :

Managment Version = X.00.01.004
 Lan Version = X.00.02.004
 AFCP Version = X.00.03.004
 ADCP Version = X.00.04.004
 CPU ROM Version = A.00.01.16
 SIC 1 ROM Version = A.00.01.16
 SIC 2 ROM Version = A.00.01.16
 SIC 3 ROM Version = A.00.01.16
 SIC 4 ROM Version = A.00.01.16
 SIC 5 ROM Version = A.00.01.16
 SIC 6 ROM Version = A.00.01.16

Enabled Severities = \$07F3

Severity : 2 Logged = 1 Lost =

Events Not Acked = 0
 Event Total Retries = 0
 MGR Pending MGR Requests = 1
 MGR Pending Port Dumps = 0
 MGR Cant Get Message = 0
 MGR Unknown MP request = 0
 MGR MP error Responses = 0
 MGR Port Mux Busy = 0

RMP Total Retries = 0

AFCP counters :

Active Connections = 4

Receive Packets :

Data Packets = 115778
 Acknowledgments = 256
 Negative Acknowledgments = 0
 Duplicate = 0
 Out of Order = 0
 Status Requests = 559

m 160

Transmit Packets :

Data Packets = 113589
 Acknowledgments = 2760

```

Negative Acknowledgments =      0
      Retransmit =      0
      Window Probe =      0

LAN counters :
  Receive :
    All Frames =      0
    To My Addr Frames =      0
    CRC Errors =      0
    No Resources =      0
    Missed =      0

  Transmit :
    Total Sent =    116616
    One Collision =      36
    More Collisions =      23
    Too Many Collisions =      0
    Lost Carrier =      0
    No Heart Beat =      0
    No Ring Space =      0
    Deferred =      1199
    Late Collision =      0

Event Index 2
  Reporting Task =    AFCP
  Severity =      2
  Procedure Number =      0
  Report Time =      450
  Connection =      1

Event =      1536

```

Where:

Machine Type	The HP product number of the DTC.
DTC Node Name	The user-assigned node name of the DTC.
NOVRAM Value	A code that indicates location of Nodal address and Boot Multicast address within DTC RAM.
Nodal address	802.3 station address of the DTC.
Boot Multicast address	802.3 multicast address which DTC uses when requesting its code file.
Host Node Address	802.3 address of MPE/XL system that downloaded the DTC.
DTC Lan Multicast Addresses	These fields will be used by DTC diagnostics in future releases.
Selftest status readout	Provides results of most recent DTC selftest. The first two columns list and identify SIC cards. The next two columns show status of the SICs. For details on the DTC selftest readout, refer to the description of the DIAG Selftest DTC subcommand.
Version Numbers	The version numbers of hardware and software modules on the DTC.

The remaining fields provide a detailed record of DTC events. They tell what events are being logged (by MPE XL) and list event frequencies. This information can be used by HP support representatives for detailed diagnosis of DTC problems.

Port

Obtains status information about a particular port.

Security level: 2.

Syntax:

PO[RT] dtcnum,cardnum,portnum

Parameters:

dtcnum The DTC number TERMDASM associates with a particular DTC. The DTC command lists each DTC with its assigned number.

Range: 1 - *n*, where *n* is the total number of DTCs configured on your system.

4-48 Terminal Diagnostic Support Monitor

When this subcommand is entered, a large amount of information related to the port is printed on your terminal screen.

Example:

(status)

Port #, #, #

(carriage return to exit) ?PO 1,1,7

Revision Number :

Hardware = \$0000 Software = \$5025
Current Host Name = default Current Host Name = default
Port Type = RS-422 Direct
Connection State = Undefined

DIODAM State :

Idle

Driver State :

Normal

Maximum Read Length = 255 Current Write Length = 0
Number of Bytes Transferred From the SIC to the CPU = 0

SIC State :

Read Pending

Number of User Characters Received by SIC = 0
Last Special Character Received by SIC = \$00


```

Modem State :
    Clear to Send = ON          Request to Send = ON
    Data Set Ready = ON        Data Terminal Ready = ON
    Ring Indicator = OFF       Data Rate Select = ON
    Data Carrier Detect = ON

    Disconnect Timer = 50      Open Timer = 1200
    DCD Timer = 300           DCD Counter = 50

    Backspace = $08           Line Delete = $18
    EOR Character = $0D        Subsystem Break Character = $19
    Alternate EOR Character = $0A Read Trigger Character = $11
    Block Mode Signal Car. = $12

    Baud Rate = 9600bps
    Parity = NONE ignore parity error
    Flow Control Timer = 0      Read Limit Timeout = 200

Read Option:
    Send CRLF = ON            3000Edit Mode = ON
    Echo Mode = ON           VPlus Block Mode = ON
    Block Mode = ON
    Read Trigger Option = ON

Device Handshake Option :
    Binary Mode = OFF         Echo !!!CR LF = ON
    Host XON/XOFF = ON        Device XON/XOFF = ON
    Enable Flow Control = ON

Asynchronous Event:
    Flow Control Timeout = OFF Subsystem Break = OFF
    Break = OFF

```

Where:

Revision Number	Provides version number of SIC hardware and software.
Current Host Name	If a connection is active, and it was initiated by the terminal, then this is the name of the host to which the port is connected. Otherwise, this field is blank.
Port Type	The type of DTC connector card attached to the port. Possible values are: RS-232-C Direct RS-422 Direct Modem
Connection State	The status of the connection between the port and the MPE XL system. Possible values are: Disconnected —Port is not in use; no MPE XL session is active on this port. Connect from AFCP Pending —MPE XL has requested and is awaiting a connection to this port. Connect from DIODAM Pending —Device has requested and is awaiting a connection to the MPE XL. Connected —The port is connected to an MPE XL system. Disconnect from AFCP Pending —MPE XL has requested and is awaiting disconnection from the port. Disconnect from DIODAM Pending —Device has requested and is awaiting disconnection from MPE XL. Bad —The port has failed its selftest, or this is a non-existent port on the SIC.

Error Messages

The following is a list of error messages which may appear when using TERMDSM. Other error messages may occur which do not have the TDSM# trailer; these messages are generated by the Online Diagnostic Subsystem or the operating system. Consult the DUI section of this manual and the operating system manuals for these errors.

15	You have entered an unknown or incorrect command. Please try again (TDSM 15)	
CAUSE	You have entered something that is not a TERMDSM command or a command that is not valid in this context.	
ACTION	Commands that are valid at this time are listed in the prompt. Please try again. Type "HELP" or refer to chapter 4 for more information.	
16	TERMDSM has compiled without \$partial eval\$ in effect. It cannot continue. This problem must be fixed at the factory. (TDSM 16)	
CAUSE	Internal error.	
ACTION	Contact your HP support representative.	
17	Your security level (3) is too low to use TERMDSM. Please log on with level 0, 1, or 2 security. (TDSM 17) For level 0, you need SM or DI capability. For Level 1, you need SM or DI or OP capability. For Level 2, you need SM or DI or OP or AM capability.	
CAUSE	You do not have the correct user capabilities to use TERMDSM.	
ACTION	Logon as a user with the correct capabilities for the TERMDSM functions you wish to use. Refer to chapter 4 for the security levels of each command and subcommand.	
18	You have entered a bad port number. Port numbers range from 0 to 7. (TDSM 18)	
CAUSE	The value you entered is not a number between 0 and 7.	
ACTION	Try again. Check the port number on the back of the DTC if necessary. (Note that modem ports range from 0 to 5 instead of 0 to 7.)	
19	The port number is missing or invalid. Port numbers range from 0 to 7. (TDSM 19)	
CAUSE	The port number was not entered or is unrecognizable.	
ACTION	Try again. Refer to the prompt or chapter 4 for syntax.	
20	You have entered a bad ldev number. Ldev numbers are positive integers. (TDSM 20)	
CAUSE	The value that you entered as a logical device number is not a number greater than zero.	
ACTION	Try again. Double check the ldev if necessary. Note that you may also use the port subcommand.	

21	The ldev number is missing or invalid. Ldev numbers are positive integers. (TDSM 21)
CAUSE	The logical device number was not entered or is unrecognizable.
ACTION	Try again. Refer to the prompt or chapter 4 for syntax.
22	You have entered an invalid SIC card number. SIC card numbers range from 0 to 5. (TDSM 22)
CAUSE	The card number that you typed in is not the range 0 to 5.
ACTION	If you made an error in typing, try again. Otherwise, doublecheck the card numbers by looking at the back of the DTC and try again.
23	The SIC card number is missing or invalid. SIC card numbers range from 0 to 5. (TDSM 23)
CAUSE	The card number was not entered or is unrecognizable.
ACTION	Try again. Refer to the prompt or chapter 4 for syntax.
24	You have entered an incorrect DTC number. Please choose a "DTC number" from this list. (TDSM 24)
CAUSE	The value entered does not correspond with any DTC on your system.
ACTION	The DTCs on your system will be listed out with their numbers. You will again be allowed to enter one of the numbers.
25	The DTC number is missing or invalid. Please choose a "DTC number" from this list. (TDSM 25)
CAUSE	The DTC number was not entered or is unrecognizable.
ACTION	The DTCs on your system will be listed out with their numbers. Try again. Refer to the prompt for syntax.
27	TERMDSM cannot do the dump because there are already too many dump files. Please purge some. Dump files are of the form TRMnnnaa.PUB.SYS. where nnn is a 3 digit number and aa is 2 alphabetic characters. (TDSM 27)
CAUSE	The dump files TRMdddAA through TRMdddZZ already exist. (The sequence ddd is the current Julian date.)
ACTION	Delete some files beginning with the file TRMdddAA since this is probably the oldest file. Note that if all files for this Julian date already exist on your system, there are over 600 dump files for today alone. You may wish to delete the large majority of these files.

32	Your security level (2) is too low to take dumps. Please log on with Level 1 security to dump ports, or level 0 security to dump ports and DTCs. (TDSM 32) For level 0, you need SM or DI capability. For Level 1, you need SM or DI or OP capability.
CAUSE	You do not have the correct user capabilities to use this function of TERMDSM. Your security level (2) is too low to use TERMDSM. Please log on with Online Diagnostic System level 0 or 1 security.
ACTION	You do not have the correct user capabilities to use this function of TERMDSM. Your security level (2) is too low to use TERMDSM. Please log on with Online Diagnostic System level 0, 1, or 2 security. Refer to chapter 4 for information on security requirements.
34	That ldev does not exist. Please check the ldev number, or dump the associated port instead. (TDSM 34)
CAUSE	Although you have entered a number in a range that could make it an ldev, that number is not configured on your system. Only DTC terminals may be referenced. TERMDSM cannot act upon the console or virtual terminals.
ACTION	Doublecheck the number by using the "SHOWJOB" command and try again. You can also dump information associated with the device by using the port subcommand (which uses the physical address instead of the ldev).
35	That DTC is owned by another person running TERMDSM. You may override that user and continue. (WARNING This may have unpredictable results. It's safer to find the other user and exit that TERMDSM. (TDSM 35). Do you want to override the other user (yes or no)?
CAUSE	More than one person is using TERMDSM, and users are directing operations to the same DTC at the same time.
ACTION	It is permissible for two or more users to run TERMDSM simultaneously as long as the users do not reference the same DTC. In this case, message 35 will not appear. If one TERMDSM user chooses to override another by continuing, it is likely that the overridden TERMDSM will produce error messages. This facility is provided primarily for the case where a terminal running TERMDSM is hung and therefore cannot exit TERMDSM. Note that if one TERMDSM user resets the port of another TERMDSM user, this may cause the system to go down.

36	No ldevs seem to be configured on your system. (TDSM 36)
CAUSE	Necessary configuration information does not exist on your system.
ACTION	Make sure the file NMCONFIG.PUB.SYS exists. Bring the system down and back up again, making sure that the proper SYSGEN files are used. (The SYSGEN files used are determined by the boot path. The file NMCONFIG.PUB.SYS is used automatically.) If the problem persists, contact your HP support representative.
37	The dump process can't find out the length of the data area. (TDSM 37)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
38	The data area is longer than TERMDSM's dump buffer. (TDSM 38)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
39	TERMDSM never received a reply to its dump data area request. (TDSM 39)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
42	You must have high (0) security to perform this function. (TDSM 42)
	For level 0, you need SM or DI capability.
CAUSE	You do not have the correct user capabilities to use TERMDSM.
ACTION	Logon as a user with the correct capabilities for the TERMDSM functions you wish to use. Refer to chapter 4 for the security levels of each command and subcommand.
43	You can't do this from your own DTC. Please run TERMDSM from another DTC or from the console. (TDSM 43)
CAUSE	If you were permitted to selftest, dump or reset the DTC to which you are logged on, your session would be aborted and you would be unable to get the test results.
ACTION	Move to a terminal on another DTC or the system console to run TERMDSM. If you intended to enter a different number, try again.

44	Warning this will reset all users on that DTC. As many as 48 terminal users could be affected. (TDSM 44)
CAUSE	If you continue, all user sessions on this DTC will be aborted.
ACTION	If you want to continue, you may wish to alert all users of this DTC that their sessions will be aborted and possibly data lost.
56	Your answer must be at least 1 and no more than 65535. (TDSM 56)
CAUSE	The value you entered is not in the range 1 through 65535.
ACTION	Enter a value in the selected range. Diagnostic loop counts are limited to this range.
62	Use the CCTL option (:FILE LIST;DEV=LP;CCTL) when printing this file. (TDSM 62)
CAUSE	Informational message.
ACTION	When printing the dump file, use the CCTL option in your print file equation. This will format the printed output in a more organized way. Printed output of dumpfiles will still be acceptable if not printed with the CCTL option.
64	The following two files make up the DTC dump. Please deliver both to your HP representative. (TDSM 64)
CAUSE	Informational message.
ACTION	Document the problem (the reason you took a dump), and deliver it to your HP support representative along with the dump files, preferably on magnetic tape.
70	There are no DTCs configured on your system. (TDSM 70)
CAUSE	Necessary configuration information does not exist on your system.
ACTION	Check the NMCONFIG.PUB.SYS file for proper DTC configuration (using :NMMGR). Bring the system down and back up again. If the problem persists, contact your HP support representative.
71	There are more DTCs on your system than TERMDSM can reference. Here are the ones TERMDSM can reference. (TDSM 71)
CAUSE	You are using more than 32 DTCs on your system. You will not be able to perform any TERMDSM functions on any DTCs that are not listed after this message.
ACTION	Contact your HP support representative.

80	You can't reset your own port or ldev. Please run TERMDSM from a different port or ldev to do this. (TDSM 80)
CAUSE	Doublecheck the numbers that you entered. If you want to reset a port you cannot be logged on to that port; the operation will abort your session.
ACTION	If you made an error on typing, try again. If you want to reset the port you are using, move to another port or the system console to run TERMDSM.
81	That ldev is unknown (not configured). Please check the ldev number, or use reset port. (TDSM 81)
CAUSE	Although you have entered a number in a range that could make it an ldev, that number is not configured on your system. Only DTC terminals may be referenced. TERMDSM cannot act upon the console or virtual terminals.
ACTION	Doublecheck the number by using the "SHOWJOB command and try again. You can also reset the port by using the port subcommand (which uses the physical address instead of the ldev).
82	You must have high (0) or medium (1) security to do resets, and high (0) security to reset DTCs. (TDSM 82) For level 0, you need SM or DI capability. For Level 1, you need SM or DI or OP capability.
CAUSE	You do not have the correct user capabilities to do this function.
ACTION	Logon as a user with the correct capabilities for the TERMDSM functions you wish to use. Refer to chapter 4 for the security level of each command and subcommand.
86	You must have high (0) or medium (1) security to test ports, and high (0) security to selftest DTCs. (TDSM 86) For level 0, you need SM or DI capability. For Level 1, you need SM or DI or OP capability.
CAUSE	You do not have the correct user capabilities to do this function.
ACTION	Logon as a user with the correct capabilities for the TERMDSM functions you wish to use. Refer to chapter 4 for the security level of each command and subcommand.
93	TERMDSM could not open the DTC manager, so the DTC dump could not be obtained. (TDSM 93)
CAUSE	Internal error, unless another TERMDSM was being run against that same DTC and you chose not to override the other user. There is no error in this case.
ACTION	Contact your HP support representative (if it is an internal error).

110	You can't test your own port. Please run TERMDISM from another port. (TDSM 110)	
CAUSE	Doublecheck the numbers that you entered. If you wish to test a port, you cannot be logged on to that port; the operation will abort your session and you will be unable to obtain the results of the test.	
ACTION	If you made an error in typing, try again. If you want to test the port to which you are logged on, move to another port or the system console to run TERMDISM.	
113	Please put a loopback hood on the port, before answering the "How many times?" question. (TDSM 113)	
CAUSE	Instruction to set up the right hardware as part of the diagnostic.	
ACTION	Refer to chapter 4 for complete information on port loopback tests.	
114	Please connect and make ready (power on, remote mode, etc.) a supported terminal, before answering the "How many times?" question. (TDSM 114)	
CAUSE	Instructions to set up the right hardware as part of the diagnostic.	
ACTION	Refer to chapter 4 for complete information on terminal loopback tests.	
115	Please connect and make ready (power on, remote mode, etc.) a supported printer (or terminal), before answering the "How many times?" message. (TDSM 115)	
CAUSE	Instructions to set up the right hardware as part of the diagnostic.	
ACTION	Refer to chapter 4 for complete information on the port test.	
121	Warning: the LDM dump is suspect. Consecutive dumps did not match. (TDSM 121)	
CAUSE	This message is written to dump files and never appears as an error message to a user terminal. There was activity during the dump process, so part of the dump relating to the Logical Device Manager may be inconsistent with the rest of the dump.	
ACTION	Submit the dump to your HP support representative anyway.	
122	You can't reset your own SIC. Please run TERMDISM from another SIC or DTC, or from the console. (TDSM 122)	
CAUSE	Your own session is on this SIC (serial interface card). If you wish to reset a SIC, you cannot be logged onto that card; the operation will abort your session and you will be unable to obtain the results.	
ACTION	If you made an error in typing, try again. Otherwise, move to a port on a different DTC or SIC, or to the system console, to run TERMDISM.	

170	The Distributed Terminal Subsystem software does not exist. Please verify that the NMCONFIG.PUB.SYS file is correct and validated, and the system has booted up without errors. This situation prevents TERMDSM from continuing. (TDSM 170)
CAUSE	The NMCONFIG.PUB.SYS file does not exist, or has no DTCs configured.
ACTION	Check the NMCONFIG.PUB.SYS file for proper DTC and terminal configurations (using NMMGR), then shutdown and restart the system.
200	DTC and mainframe software versions are incompatible. (TDSM 200)
CAUSE	Incompatibilities were found between subsystem software on the HP3000 and subsystem software on the DTC.
ACTION	Contact your HP support representative to check the DTCM and download file versions with NMMAINT. If this is not the problem, the cause is an internal error.
201	Bad message length. (TDSM 201)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
203	Invalid operation. (TDSM 203)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
204	Request not supported. (TDSM 204)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
205	Sub request not supported. (TDSM 205)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

206	Entity not supported. (TDSM 206)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
207	No such entity. (TDSM 207)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
208	No parm for entity. (TDSM 208)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
213	The DTC reported an error or failure. (TDSM 213)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
218	End of Record received. Time = xx msec. (TDSM 218)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.
219	Too many characters came back. Time = xx msec. (TDSM 219)
CAUSE	More characters were returned than were sent. Possibly someone pressed a key on the terminal under test, if this was a terminal loopback. If so, try again. Otherwise, probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.

220	Parity error. Time = xx msec. (TDSM 220)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.
221	Hardware receive overrun. Time = xx msec. (TDSM 221)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.
222	Framing error. Time = xx msec. (TDSM 222)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.
223	SIC Buffer overflow. Time = xx msec. (TDSM 223)
CAUSE	Internal error.
ACTION	Contact you HP support representative.
226	No record separator in Vplus. Time = xx msec. (TDSM 226)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.

229	Read timeout (no data came back). Time = xx msec. If external loopback, check loopback hood. If terminal loopback, check connections, terminal configuration, etc. (TDSM 229)
CAUSE	No or too few characters came back from the loopback.
ACTION	If this is an internal loopback, this indicates hardware problem. Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDASM RESET command), the DTC selftest, and the DTC offline diagnostics. If this is an external loopback, check the loopback hood to be sure it is securely connected and on the correct port. If this does not help, follow the steps in the "internal" case above. If this is a terminal loopback, check that a supported, powered on terminal is securely connected to the proper port. Check baud rate, parity, etc. Note that the terminal configuration must match the configuration file, not the state the terminal was in when it was last used. If this does not help, run internal, then external loopback to discriminate between a DTC problem and a terminal/cabling problem.
230	Alternate end-of-record received. Time = xx msec. (TDSM 230)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDASM RESET command), the DTC selftest, and the DTC offline diagnostics.
231	Subsystem break. Time = xx msec. (TDSM 231)
CAUSE	Probable hardware failure.
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDASM RESET command), the DTC selftest, and the DTC offline diagnostics.
232	No modem was connected during the operation. Time = xx msec. (TDSM 232)
CAUSE	During the test, the modem on that port disconnected.
ACTION	Reconnect the modem and try again. If the failure continues, or there was no modem in use on that port, there is a probable hardware failure. Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDASM RESET command), the DTC selftest, and the DTC offline diagnostics.

234	Console attention. Time = xx msec. (TDSM 234)
CAUSE	Probable hardware failure.
ACTION	Contact your HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.
235	Operation aborted. (TDSM 235)
CAUSE	The operation was aborted by the DTC. Possibly, the download file is incorrect.
ACTION	Check the version and integrity of the download file. Contact your HP support representative. You may obtain additional information by running the DTC selftest DTC selftest or the DTC offline diagnostic.
236	End of dump file encountered. (TDSM 236)
CAUSE	The end of the DTC upload (dump) file has been encountered when there is still more information to write to it.
ACTION	Check to see if there is sufficient disc space (approx. 2200 sectors are required). Otherwise, contact your HP support representative.
238	Security invalid. (TDSM 238)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
239	Bad packet. (TDSM 239)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
240	During a DTC reset/dump request from the mainframe. (TDSM 240)
CAUSE	Informational message which may be printed after an error message. It describes which phase of an operation the error occurred.
ACTION	Refer to the cause and action descriptions for the actual error message preceding message 240.

241	During the DTC download operation. (TDSM 241)
CAUSE	Informational message which may be printed after an error message. It describes which phase of an operation the error occurred.
ACTION	Refer to the cause and action descriptions for the actual error message preceding message 241.
242	During a DTC upload (dump) operation. (TDSM 242)
CAUSE	Informational message which may be printed after an error message. It describes which phase of an operation the error occurred.
ACTION	Refer to the cause and action descriptions for the actual error message preceding message 242.
243	Bad parameter from TERMDISM. (TDSM 243)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
244	Timeout waiting for the DTC to respond. The DTC may be powered off, malfunctioning, or misconfigured. (TDSM 244)
CAUSE	Your TERMDISM command resulted in an instruction being sent to a DTC, and the DTC did not respond in any way.
ACTION	Note that the DTC list printed by TERMDISM, when you first run TERMDISM, or response to the "DTC" command, is only a list of DTCs configured, not necessarily a list of installed, powered on, and working DTCs. If the DTC in question was working, and no changes were made to the configuration file, ensure that the DTC is powered on. If so, this error indicates a probable hardware error. Contact your HP support representative. If this is a new installation or DTC, which has never been run before, the above comments may still apply. However, if the configuration is incorrect, a functional DTC may still not respond. Check the configuration file, power to the DTC, connections, etc. If problems persist, contact your HP support representative. This error may also be the result of a LAN problem. If other DTC(s) are working on the same LAN, the problem is probably related to the DTC or its configuration. If there is only one DTC, or other DTCs on the same LAN are unresponsive, then the LAN hardware or software may be at fault. The LAN diagnostic LANDAD can be run to investigate LAN problems.

245	Write to IEEE 802.3 DAM failed. (TDSM 245)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
247	Bad Network Management packet received from DTC. (TDSM 247)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
248	The DTC requested to be downloaded during the operation. (TDSM 248)
CAUSE	The DTC was trying to power up at the same time you were trying to access it through TERMDSM (or the DTC detected an error).
ACTION	Wait for the download, or upload/download, to complete. (As shown by the console messages.) See then whether the DTC is now operational. If not, try your TERMDSM operation again. If this still does not resolve the problem, contact your HP support representative.
249	The DTC requested that it be dumped during the operation. (TDSM 249)
CAUSE	The DTC detected an internal error at the same time you were trying to access it through TERMDSM.
ACTION	Wait for the download, or upload/download, to complete. (As shown by the console messages.) Then see whether the DTC is now operational. If not, try your TERMDSM operation again. If this still does not resolve the problem, contact your HP support representative.
250	DTC and mainframe software versions are incompatible. (TDSM 250)
CAUSE	Incompatibilities were found between subsystem software on the HP3000 and subsystem software on the DTC.
ACTION	Contact your HP support representative to check the DTCM and download file versions with NMMAINT. If this is not the problem, the cause is an internal error.
251	Bad RMP packet received from DTC. (TDSM 251)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

252	Could not open the download or dump file. (TDSM 252)
CAUSE	Files pertaining to this operation cannot be accessed.
ACTION	If this is a dump, check for sufficient disc space to create the dump file. Approximately 2200 sectors of disc space are required. If enough disc space seems available (or if you were not trying to perform a dump operation) contact your HP support representative.
253	Unable to get a system timer. (TDSM 253)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
254	Software bug encountered. (TDSM 254)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
255	Dump overflow DTC sent more data than expected. (TDSM 255)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
256	MPE/XL create_object failed. (TDSM 256)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
257	Unknown internal error. (TDSM 257)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

258	Application request timeout occurred . (TDSM 258)
CAUSE	Internal error.
ACTION	This means two users of TERMDSM were directing TERMDSM commands to the same DTC at the same time, and the other TERMDSM user answered "yes" to the override question (see TERMDSM message 35). Other error messages are likely to be printed also, because your operations were interrupted. These error messages should be ignored. There is nothing wrong with multiple users of TERMDSM running at the same time, as long as they do not try to direct commands to the same DTC at the same time. The message and interlock system of message 35 was designed to prevent this conflict. Contact your HP support representative.
259	DTC not opened by this TERMDSM. Possibly your TERMDSM was overridden by another TERMDSM. (TDSM 259)
CAUSE	Another user of TERMDSM has explicitly overridden your use of TERMDSM.
ACTION	Locate and coordinate users of TERMDSM. This means two users of TERMDSM were directing TERMDSM commands to the same DTC at the same time, and the other TERMDSM user answered "yes" to the override question (see TERMDSM message 35). Other error messages are likely to be printed also, because your operations were interrupted. These error messages should be ignored. There is nothing wrong with multiple users of TERMDSM running at the same time, as long as they do not try to direct commands to the same DTC at the same time. The message and interlock system of message 35 was designed to prevent this conflict.
260	Bad NMP error code received. (TDSM 260)
CAUSE	Unknown error reported by the Distributed Terminal Subsystem to TERMDSM.
ACTION	Contact your HP support representative.
261	Bad RMP error code received. (TDSM 261)
CAUSE	Unknown error reported by the Distributed Terminal Subsystem to TERMDSM.
ACTION	Contact your HP support representative.

262	The DTC subsystem has reported that an error was detected. Here is the specific message. (TDSM 262)	
CAUSE	This is the standard message that precedes specific error messages.	
ACTION	Follow the instructions for the error message that follows this message.	
263	An error of unknown type was reported by the DTC subsystem. (TDSM 263)	
CAUSE	An error unknown to TERMDISM was reported by the Distributed Terminal Subsystem.	
ACTION	Contact your HP support representative.	
264	The DTC reported an unknown status as a result of the loopback. Time taken=xx. (TDSM 264)	
CAUSE	An error unknown to TERMDISM was reported by the Distributed Terminal Subsystem.	
ACTION	Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.	
270	The port is malfunctioning, or not present on that SIC. Modem SICs have ports 0 through 5, and all other SICs have ports 0 through 7. (TDSM 270)	
CAUSE	The DTC has reported that the port in question is bad or does not exist.	
ACTION	Note the range of ports in the error message. If you are specifying a port that does not exist, check your information to determine the correct port. For reset and dump, you can specify an Ldev number instead, of the port to do the operation. If the port does exist, a hardware failure has probably occurred. Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.	
271	That SIC is malfunctioning, or not installed. TERMDISM can not always tell the difference between "not installed" and a serious malfunction. (TDSM 271)	
CAUSE	Possible hardware problem or no card is installed in that slot.	
ACTION	Doublecheck that the SIC number you specified is correct. If it is, it is likely that the SIC is defective. Contact you HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDISM RESET command), the DTC selftest, and the DTC offline diagnostics.	

272	The operation was aborted because it was reset. (TDSM 272)
CAUSE	Another TERMDSM user reset the operation while in progress.
ACTION	See message 246 for a discussion of multiple TERMDSM operations directed against the same DTC. Try your operation again.
273	The modem on that port disconnected. (TDSM 273)
CAUSE	The modem on that port disconnected during the operation.
ACTION	Reconnect the modem and try again. If there was in fact no modem, then this is indicative of an internal problem. Contact your HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.
274	The DTC reported that an unexpected event occurred. (TDSM 274)
CAUSE	A user on that terminal may have typed control-y or break (or there is an internal error).
ACTION	Try the operation again without touching keys on the terminal under test. If no change, an internal error or hardware failure is indicated. Contact your HP support representative. You may obtain additional information by running the card selftest (through the SIC subcommand of the TERMDSM RESET command), the DTC selftest, and the DTC offline diagnostics.
275	The loopback data is too long. (TDSM 275)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
280	Dump failed: Open of DTCM failed, or DTC owned and not overridden (TDSM 280)
CAUSE	This message is written to dump files, and never appears as an error message to a user terminal. If the user was informed the DTC was in use by another TERMDSM (see message 35), and the second user chose "NOT to override, then this message is normal and is not cause for alarm. Otherwise, it is indicative of an internal error.
ACTION	If the problem persists, take another dump. If you are sure another TERMDSM was not involved, contact your HP support representative.

281	Dump failed: Send or Receive of dump request failed. (TDSM 281)
CAUSE	Internal error. This message is written to dump files and never appears as an error message to a user terminal.
ACTION	Contact your HP support representative.
282	Dump failed: Bad function in reply to dump request. (TDSM 282)
CAUSE	Internal error. This message is written to dump files and never appears as an error message to a user terminal.
ACTION	Contact your HP support representative.
283	Dump failed: Bad reply (llio) status from DTCM. (TDSM 283)
CAUSE	Internal error. This message is written to dump files and never appears as an error message to a user terminal.
ACTION	Contact your HP support representative.
290	Sorry, TERMDSM does not provide services for the console. (TDSM 290)
CAUSE	The user entered the ldev number of the physical console for a reset or dump. TERMDSM cannot act upon the physical console port because it is not on the DTC. Note that the discussion here is for the physical console, not a terminal which has taken control of the logical console via the "CONSOLE command. TERMDSM can only operate on ldevs and ports that are on the DTC.
ACTION	Check the ldev number again. TERMDSM cannot be of help if there is a problem with the physical console. If the physical console has a problem, move the console to another port with the CONSOLE command, or reboot the system. If the problem persists, contact your HP support representative.
300	IPC message received in wrong subqueue. (TDSM 300)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

301	DTCM port data area not initialized. (TDSM 301)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
302	Bad machine type found in RMP packet. (TDSM 302)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
303	Non supported download file type found in RMP packet. (TDSM 303)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
304	Bad source IEEE 802.3 address found in RMP packet. (TDSM 304)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
305	Bad session ID found in RMP packet. (TDSM 305)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
306	Bad path_id received from WDM. (TDSM 306)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

307	Boot (download) complete message received before expected. (TDSM 307)
CAUSE	The DTC has reported that it has finished the download of its operating firmware before the DTCM expects it.
ACTION	Often, this situation will recover by itself. Look on the console for a "boot complete" message. Otherwise, try the operation again. If the problem persists, contact your HP support representative.
308	IO-get-frame failed. (TDSM 308)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
309	Unimplemented action called. (TDSM 309)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
310	Bad download file type. (TDSM 310)
CAUSE	Possible corrupt download file.
ACTION	Contact your HP support representative to check the integrity of the download file.
311	Bad reply from memory manager for Aux data area. (TDSM 311)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
312	DCC sent bad reply in response to config. data request. (TDSM 312)
CAUSE	Internal error.
ACTION	Contact your HP support representative.

313	DTC did not request the configuration to be download. (TDSM 313)
CAUSE	After a download, the DTC did not request its configuration.
ACTION	Sometimes this situation is self-correcting. Look at the console for a message saying "configuration complete". Otherwise, try the operation again. If the problem persists, contact your HP support representative.
314	DCC did not reply to configuration data request. (TDSM 314)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
315	Unexpected RMP packet received. (TDSM 315)
CAUSE	Internal error.
ACTION	Contact your HP support representative.
316	DTC unexpectedly requested config. information to be downloaded. (TDSM 316)
CAUSE	The DTC requested its configuration information at a time the DTCM did not expect it.
ACTION	Sometimes this situation is self-correcting. Wait and see if the DTC becomes operational; monitor the console for messages saying the configuration is complete. Otherwise, try the operation again. If the problem persists, contact your HP support representative.



Contents

5. I/O Test Tool	
Introduction	5-1
Defects and Enhancements	5-1
Minimum Configuration	5-2
Operating Instructions	5-2
RUN Command	5-2
Test Execution	5-3
Program Mode	5-4
Command Summary	5-4
Control Commands	5-5
ABORT	5-6
EXIT	5-7
RESUME	5-8
RUN	5-9
SUSPEND	5-10
User Program File Commands	5-11
LOAD	5-12
PURGE	5-13
SAVE	5-14
SHOWFILE	5-15
Program Editing Commands	5-16
DELETE	5-17
LIST	5-18
MODIFY	5-19
MOVE	5-20
RENUMBER	5-21
Miscellaneous Commands	5-22
HELP	5-23
REDO	5-24
Instruction Summary	5-25
Test Environment Instructions	5-25
ERRPAUSE	5-26
RELDEVICE	5-27
SETDEVICE	5-28
SETTIMER	5-29
SHOWDEV	5-30
Buffer Manipulation Instructions	5-31
ADJBUFF	5-32
ALTBUFF	5-33
COMPBUFF	5-34
COPYBUFF	5-35
DBUFF	5-36

DEFBUFF	5-37
FILLBUFF	5-38
FINCBUFF	5-39
RELBUFF	5-40
SHOWBUFF	5-41
Predefined I/O Request Instructions	5-42
ABORTIO	5-43
EINCADDR	5-44
EXECUTE	5-45
DSTATUS	5-46
INCADDR	5-47
PROTECT	5-49
SETADDR	5-50
SETDATA	5-51
SETOPTION	5-52
SHOWPARM	5-53
HP-IB Device Adapter Program Instructions	5-54
CASEJUMP	5-55
CLEAR	5-57
CRCCOMP	5-58
CRCINIT	5-59
CRCWRITE	5-60
DSJ	5-61
ENDHPIB	5-63
HALT	5-64
IDENTIFY	5-65
JUMP	5-66
ONTIMEOUT	5-67
PINDEX	5-68
RBURST	5-69
RDATA	5-70
RDMA	5-71
SETHPIB	5-72
SHOWHPIB	5-73
TIMEOUTOFF	5-74
TIMESTAMP	5-75
UNLOCK	5-76
WAITPOLL	5-77
WBURST	5-78
WDATA	5-79
WDMA	5-80
WINTERF	5-81
HP-CIO DMA Chain Instructions	5-82
ADDQUAD	5-83
ENDHPCIO	5-85
SETHPCIO	5-86
SHOWHPCIO	5-87
Program Statement Summary	5-88
Program Descriptions	5-89
Program Statements	5-89
COMMENT	5-90

DO-LOOP TO	5-91
GOTO	5-92
IF-THEN or IFN-THEN	5-93
PAUSE	5-95
PRINT	5-96
STOP	5-97
Predefined I/O Requests	5-98
Examples	5-103
Example 1: Read and Write to a CS80 Disk.	5-103
Example 2: Limited Disk Pattern Search	5-104
Example 3: Forcing a Protocol Error on a CS80 Disk	5-105
Example 4: Locate and Read Full Sector for a CS80 Disk	5-106
Example 5: Unblocked Write to a Streaming Tape	5-107
Example 6: Write Lines of Text to EPOC	5-108
Example 7: Ripple Print of Test to a CIPER Printer	5-109
Example 8: Initiating Selftest on a CIPER Printer	5-111
Error and Warning Messages	5-113

Tables

5-1. Tape Drive Device	5-99
5-2. Disk Drive	5-100
5-3. Page Printer Devices	5-101
5-4. CIPER Devices	5-101
5-5. HP-IB Device Adapter	5-101
5-6. CIO Channel Adapter	5-102
5-7. TMUX Device Adapters	5-102

I/O Test Tool

Introduction

The I/O Test Tool (IOTT) is a utility intended for diagnosis of I/O-related problems online from any system terminal. No restriction on terminals exists. This utility runs only with the MPE XL operating system on any HP Precision Architecture RISC computer system. The IOTT provides the user with a means to directly access the I/O system. Numerous commands, instructions, and program statements are available as inputs through I/O Test Tool. The user can also display data that was used for the I/O request. The HELP facility provides users with information on any of the I/O Test Tool inputs and input syntax. Part of this HELP facility can also be accessed through the Diagnostic User Interface (DUI). To access some devices, the user must have sole access to the system; this mode of operation is called Single User Mode (SUM). To initiate SUM, the user must return to the DUI. The IOTT does not have the capability of altering the operating mode of the diagnostic system.

Defects and Enhancements

Submit defect reports and enhancement requests concerning this diagnostic through the STARS database referencing product number 30600-10020.

Minimum Configuration

The minimum configuration required to run this utility consists of an HP Precision Architecture RISC computer system up and running on the MPE XL operating system.

Operating Instructions

Before attempting to run the utility, ensure that the user has diagnostic level 0 security as described in the Online Diagnostics Overview under Security.

RUN Command

To bring up the Online Diagnostic subsystem, enter the following command to the MPE XL system prompt:

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 chapter of this manual for details.

Note



The device to be tested must be powered up and on line. Device physical locations (pdev) shown in the **RUN** commands are those of the devices on the "typical A1002A" system configuration described in the chapter on DUI. The pdev value entered must be correct for the system being tested.

To run the diagnostic, enter:

```
DUI >RUN IOTT
      |
      |  no parameters required to
      |  load test suite
```

Test Execution

If the LDEV or PDEV parameter is given, I/O Test Tool will perform a SETDEVICE instruction using the information from the LDEV and PDEV parameter, before prompting for input. The following message will be displayed to indicate that action was taken:

```
Performing SETDEVICE on LDEV = ! , please wait ...
Performing SETDEVICE on PDEV = ! , please wait ...
```

If the user entered the ERRPAUSE parameter, IOTT will perform an ERRPAUSE ON instruction, before prompting for input. The following message will be displayed to indicate that action was taken:

```
Performing ERRPAUSE ON instruction, please wait ...
```

If any of the DUI RUN command parameters are not used, warning messages will be displayed to indicate that these parameters will be ignored.

The diagnostic displays the following header and welcome message:

```
*****
****                                     ***
****                               I/O TEST TOOL                               ***
****                                     ***
****      (C) Copyright Hewlett Packard Co. 1987                             ***
****                               All Rights Reserved.                       ***
****                               Version A.01.00                             ***
****                                     ***
**** Warning: Use of this utility by persons other                          ***
**** than Hewlett-Packard Support Personnel could                          ***
**** result in software support contract                                    ***
**** invalidation and loss of data.                                         ***
****                                     ***
*****
```

```
Welcome, Today is MON, August 10, 1987 at 12:30 PM
```

```
I/O Test Tool initialization in progress, please wait ...
```

Once the IOTT is invoked, the following prompt will be displayed to the user:

```
IOTT >
```

IOTT is now waiting for an input from the user. The three types of inputs available in I/O Test Tool are: commands, instructions, and program statements. The available inputs are listed in this section under command, instruction, program statement summary and descriptions, and predefined I/O requests. In addition selected script examples are listed at the end of this section ahead of the error message listing. To exit IOTT, the user simply types EXIT. Upon termination of the IOTT, the following message will be displayed and control will return to the diagnostic system:

```
IOTT Exiting ...
IOTT (PIN #) has just terminated.
DUI >
```

Program Mode

To increase control over the flow of instruction execution, inputs to I/O Test Tool can be presented in program form. Statements (such as GOTO) added to other I/O Test Tool instructions allow repeated execution of a particular sequence. The program can be saved into a disk file for later use.

Although program mode is available, I/O Test Tool is not intended to be used as a language for writing formal diagnostic programs. Rather, it is intended to increase user control capability by providing program statements that direct control to different parts of the program.

A program in I/O Test Tool is a sequence of instructions and program statements, each preceded by an integer line number. The program may be entered by typing each line on the terminal or reading each line from a disk file using the LOAD command. The integer line number, preceding each input, will indicate to the I/O Test Tool that the instruction is part of a program and is not to be executed immediately. In program mode, instructions and program statements will not be executed until the RUN command is issued. For this reason, each input line is stored in the Program Storage Area. Individual input lines can be overwritten by retyping the line. Program editing commands are also available to aid in any modifications that need to be performed on the program stored in the Program Storage Area.

The syntax of each line will be checked as it is entered. If any errors are detected, the line will not be stored in the Program Storage Area. References to line numbers in statements such as GOTO will be checked directly after RUN is entered. If any errors are detected, execution of the program will not begin. If any errors are detected during execution of the stored program, the line number of the statement will be displayed to the user, and the program will continue.

If the user wishes to perform some task after an error is detected during program execution, the PAUSE mode may be entered. PAUSE mode will allow execution of any instruction or command which will not alter the contents of the Program Storage Area. PAUSE mode can be entered in two ways. A PAUSE instruction is available, which will force I/O Test Tool into PAUSE mode when encountered. The second method involves the ERRPAUSE instruction. When ERRPAUSE is enabled, PAUSE mode will be entered whenever an error is encountered during the execution of a program.

Command Summary

I/O Test Tool commands are inputs which are executed immediately after they are received and cannot, except for the LOAD command as described above, be used as part of a program. Commands which alter the contents of the program storage area will not be executed in PAUSE mode. There are four categories of input commands available to the user:

- Control Commands (CC),
- User Program File Commands (UPFC),
- Program Editing Commands (PEC),
- Miscellaneous Commands (MC).

The following is a list and brief description of commands available in IOTT.

Command Name	Command Category	Description
ABORT	(CC)	Terminates program currently in PAUSE mode.
DELETE	(PEC)	Deletes the indicated line(s) from the Program Storage Area.
EXIT	(CC)	Terminates I/O Test Tool and returns control to the Diagnostic User Interface (DUI).
HELP	(MC)	Accesses HELP facility for information about I/O Test Tool and its commands.
LIST	(PEC)	Lists the contents of the Program Storage Area.
LOAD	(UPFC)	Reads program statements from the file indicated and stores those statements into the Program Storage Area.
MODIFY	(PEC)	Allows user to edit one line stored in the Program Storage Area.
MOVE	(PEC)	Allows the user to move line(s) in the Program Storage Area to another part of the program.
PURGE	(UPFC)	Removes a load file created by an I/O Test Tool.
REDO	(MC)	Displays (for command editing) and replays previous command.
RENUMBER	(PEC)	Renumbers the statements in the Program Storage Area.
RESUME	(CC)	Resumes processing of a PAUSED program stored in the Program Storage Area.
RUN	(CC)	Executes the program stored in Program Storage Area.
SAVE	(UPFC)	Saves the program stored in the Program Storage Area into a disk file.
SHOWFILE	(UPFC)	Lists all of the files in the diagnostic group and account.
SUSPEND	(CC)	Temporarily suspends I/O Test Tool and returns control to the Diagnostic User Interface (DUI).

Control Commands

The following commands are used to control the current execution mode of I/O Test Tool. These include commands for initiating execution of a program stored in the Program Storage Area, as well as resuming or terminating the program after it enters PAUSE mode. There is also a command to temporarily return control to the Diagnostic User Interface, and a command to exit I/O Test Tool.

ABORT

The **ABORT** command is used to abort the program stored in the Program Storage Area after the **PAUSE** instruction was encountered. This command is only accessed in **PAUSE** mode. The user may verify that **PAUSE** mode has been entered by the **IOTTPAUSE>** prompt. If the user entered **PAUSE** mode using **ERRPAUSE**, this command terminates **PAUSE** mode.

Syntax:

ABORT

Limitations:

The **ABORT** command will only be accessed in **PAUSE** mode.

Example:

```
IOTT>10 print "line 10"
IOTT>20 pause
IOTT>30 print "line 30"
IOTT>run
line 10
PAUSE Statement Encountered on Line Number 20
IOTTPAUSE>abort
```

EXIT

The EXIT command causes I/O Test Tool to terminate, and returns control to the Diagnostic User Interface. All I/O devices not released by the user are returned to the system. All outstanding unblocked I/O requests are aborted. If any problems are encountered while performing the above tasks, appropriate error messages are displayed to the user before I/O Test Tool terminates. The EXIT command is not accepted in PAUSE mode, leaving the user to enter the ABORT or RESUME command to leave PAUSE mode before entering the EXIT command.

Syntax:

EXIT

Limitations:

The EXIT command is not accepted in PAUSE mode.

Example:

```
IOTT>exit
IOTT Exiting...
IOTT (PIN 100) has just terminated.
DUI >
```



RESUME

The **RESUME** command resumes execution of the program stored in the Program Storage Area after the **PAUSE** program statement was encountered. This command is only accessed in **PAUSE** mode. The user is able to verify that **PAUSE** mode has been entered by the **IOTTPAUSE>** prompt. If the user entered **PAUSE** mode through **ERRPAUSE**, this command will terminate **PAUSE** mode.

Syntax:

RESUME

Limitations:

The **RESUME** command is only accessed in **PAUSE** mode.

Example:

```
IOTT>10 print "line 10"
IOTT>20 pause
IOTT>30 print "line 30"
IOTT>run
line 10
PAUSE Statement Encountered on Line Number 20
IOTTPAUSE>resume
IOTT>line 30
```

RUN

The **RUN** command begins execution of the program stored in the Program Storage Area. The user may optionally specify the number of times the program is to be executed. If no value is given, the program executes once.

After the **RUN** command is issued, the following checks are performed before the program is executed:

- Verify that all lines referenced in the program are present in the Program Storage Area.
- Verify that the DO-LOOP program constructs are nested correctly.
- Verify, for an HP-IB DAP program, that all lines referenced are HP-IB instructions.
- If any DSJ or CASEJUMP instructions are used within the HP-IB DAP program, verify that lines given are forward reference.

If any of the errors described are detected, the user is notified of the error, as well as the line number of the statement, and the program is not executed.

During execution of the program, the user may abort the program by entering a user interrupt ((**CTRL**) /Y). When an error is detected during the execution of the program, the line number of the statement which produced the error is displayed to the user.

Syntax:

RUN [*count*]

Optional Parameters:

count Integer value greater than zero. Used to specify number of times operation is to be performed.

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>run
line 10
line 20
line 30
IOTT>run 2
line 10
line 20
line 30
line 10
line 20
line 30
IOTT>
```

SUSPEND

The **SUSPEND** command allows the user to temporarily suspend I/O Test Tool and return control to the Diagnostic User Interface. At this point, the user may execute any Diagnostic User Interface command. I/O Test Tool can be resumed by entering the Diagnostic User Interface **RESUME** command.

Syntax:

SUSPEND

Example:

```
IOTT>suspend
DUI>resume
IOTT>
```

User Program File Commands

The following commands are available to utilize user program files. These commands give the user the flexibility of saving programs into a disk file for later use.

LOAD

The **LOAD** command reads program statements from the file indicated and stores those statements into the Program Storage Area. The Program Storage Area must be cleared before a program can be loaded from a file (use **DELETE ALL**). This prevents the user from accidentally overwriting a program that was to be saved. The syntax of each line is checked as each line is read. If any errors are detected, the user is notified of the error and that line will not be entered into the Program Storage Area. References to line numbers in statements, such as **GOTO**, are checked directly after the **RUN** command is entered. All files given are assumed to reside in the diagnostic group and account. This procedure assumes that the file was created by an I/O Test Tool session. Using the **LOAD** command with any files created differently may result in an error message. The user is also cautioned to inspect the Program Storage Area (use **LIST** command) if a user interrupt (**CTRL**) /Y termination was invoked.

Syntax:

LOAD {file name}

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Limitations:

File must be in diagnostic group and account.

Example:

```
IOTT>load example
IOTT>list all
10 print "line 10"
20 print "line 20"
30 print "line 30"
IOTT>
```

PURGE

The PURGE command allows the user to purge a file created by an I/O Test Tool process. If the user tries to purge other files, an error message is displayed and the file remains untouched.

Syntax:

PURGE {file name}

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Limitations:

File must be in diagnostic group and account.

Example:

```
IOTT>showfile example
EXAMPLE
IOTT>purge example
IOTT>showfile example
IOTT>
```

SAVE

The **SAVE** command allows the user to save the program stored in the Program Storage Area into a disk file. The parameter given is the file name which is created in the Diagnostic group and account. The file name is limited to 8 alphanumeric characters, and the first character must be an alpha. If the file already exists, the user is asked if the file is to be overwritten. A **YES** answer overwrites the file. The user can only overwrite files which were created by an I/O Test Tool process. If an attempt is made to overwrite any other file, an error message is displayed to the user, and the file is left untouched. Once the program has been saved, the number of lines written to the file is displayed.

Syntax:

SAVE {file name}

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Limitations:

The file will be created and saved in the diagnostic group and account.

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>save example
Purge Old File (YES/NO)? yes
3 Lines Were Saved In File example
IOTT>
```

SHOWFILE

The **SHOWFILE** command displays all the files within the diagnostic group and account. The parameter for this command is a file specifier string, which can be a maximum of 8 characters. A wild card character of “@” can be used also. If a file specifier is not given by the user, all the files within the diagnostic group and account are displayed. This is the equivalent of a **SHOWFILE *** command.

Syntax:

SHOWFILE [file specifier string]

Limitations:

File must be in diagnostic group and account.

Example:

```
IOTT>showfile ex@  
EXAMPLE  
IOTT>
```

Program Editing Commands

The following commands can be used to manipulate the contents of the Program Storage Area. Entering a program from the terminal requires that all program statements be preceded with an integer line number. This indicates to I/O Test Tool that the input is part of a program. Individual program lines can be overwritten by re-entering the line.

DELETE

The **DELETE** command deletes the indicated line(s) from the Program Storage Area. The parameter given indicates the line number of the statement to be deleted, or a range of statements to be deleted. For the range of statements to be deleted, the starting line given must be present in the Program Storage Area. All lines are displayed to the user as they are deleted. **DELETE ALL** clears the Program Storage Area, and in this case, the lines deleted are not be displayed.

Syntax:

```
DELETE {line number}
DELETE {line number}/{line number}
DELETE ALL
```

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>40 print "line 40"
IOTT>delete 20/30
20 print "line 20"
30 print "line 30"
IOTT>list all
10 print "line 10"
40 print "line 40"
IOTT>
```



LIST

The **LIST** command displays the contents of the Program Storage Area. The user specifies statements to be displayed by giving a line number or a range of line numbers. The user may also use the reserved word **ALL** to display the entire Program Storage Area and terminate the listing of the Program Storage Area by entering a user interrupt (**CTRL**) /Y.

Syntax:

```
LIST {line number}
LIST {line number}/{line number}
LIST ALL
```

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>list 10
10 print "line 10"
IOTT>list all
10 print "line 10"
20 print "line 20"
30 print "line 30"
IOTT>
```

MODIFY

The **MODIFY** command allows the user to modify one line stored within the Program Storage Area. This command deletes the indicated statement from the Program Storage Area, and permits the user to edit the statement before reinserting it into the Program Storage Area. This line editing process is similar in format to the **REDO** command. Please refer to the **REDO** command for editing characters used to modify the text. If the syntax of the resulting statement is not correct, an error message is displayed and the line will not be reinserted into the Program Storage Area.

Syntax:

MODIFY {line number}

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 print "line to"
IOTT>modify 10
> 10 print "line to"
>
> 10 print "line 10"
>
IOTT>10 print "line 10"
IOTT>list all
10 print "line 10"
IOTT>
```


MOVE

The **MOVE** command allows the user to move line(s) in the Program Storage Area to another location of the program. The first parameter is the existing line number, or a range of line numbers for the statements to be moved. The parameter after the reserved word "TO" is the new line number, indicating where the statements are to be moved. For the range of line numbers, the starting line number must exist. If the new line number already exists, the move will not be performed. If there is not enough room at the destination, an error message is displayed and the move is not performed. In this case, the user must renumber the program and then try the move again.

Syntax:

```
MOVE {line number}/{line number} TO {line number}
MOVE {line number} TO {line number}
```

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>40 print "line 40"
IOTT>50 print "line 50"
IOTT>move 30/40 to 15
IOTT>list all
10 print "line 10"
15 print "line 30"
16 print "line 40"
20 print "line 20"
50 print "line 50"
IOTT>
```

RENUMBER

The **RENUMBER** command allows the user to renumber the statements within the Program Storage Area. The optional parameter indicates the value by which the renumbering is performed. This value must be greater than zero. All references to line numbers are adjusted (i.e., GOTO 30). The maximum line number is 9999. If no value is given, the program stored in the Program Storage Area is renumbered by increments of 10.

Syntax:

RENUMBER [value]

Optional Parameters:

value An integer value used to specify a data value. This value can be preceded by a base identifier (\$=HEX, %=OCT, *=DEC).

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 print "line 30"
IOTT>40 print "line 40"
IOTT>renumber 1000
IOTT>list all
1000 print "line 10"
2000 print "line 20"
3000 print "line 30"
4000 print "line 40"
IOTT>
```

Miscellaneous Commands

The following commands are provided for the user to make I/O Test Tool easier to use.

HELP

The **HELP** command displays information from the I/O Test Tool help facility. By entering **HELP** alone with no parameters, the user accesses the general help facility. This is the same help facility which can be displayed at the Diagnostic User Interface level, but within I/O Test Tool, an additional message appears to request if the user wishes information on any of the following:

- Summary of I/O Test Tool Commands.
- Brief list of I/O Test Tool Commands.
- Summary of predefined identifiers for tape devices.
- Summary of predefined identifiers for CS80 devices.
- Summary of predefined identifiers for Epoc devices.
- Summary of predefined identifiers for CIPER devices.
- Summary of predefined identifiers for CIO Channel Adapters.
- Summary of predefined identifiers for HP-IB Device Adapters.
- Summary of predefined identifiers for Six-Channel Terminal Multiplexers.

After the requested information is displayed, I/O Test Tool will display the IOTT> prompt for further input.

The user may also request information on a specific I/O Test Tool command, instruction, or program statement. The first parameter for the **HELP** command is the input identifier. At this point, information concerning that input is displayed to the user, as well as the syntax for that input. If the option **SYNTAX** is given, only the input syntax information for that input will be displayed.

Syntax:

HELP [command, instruction, or statement name] [: **SYNTAX**]

Example:

```
IOTT>help help : syntax
```

Input Format:

```
IOTT>HELP [ Command, instruction, or statement name ] [ : SYNTAX ]
```

```
IOTT>
```

REDO

The REDO command allows the user to redo the last input given. When the REDO command is given, the last input is displayed to the user. The user may edit this input using any of the following editing characters:

- R Text following this character overlays existing text starting at the location of the R. This character is also the default, if the first character is not one listed here, the replace function is assumed.
- I Text following this character is inserted at the location just before the location of the I.
- D Each D causes the corresponding character to be deleted.

Entering a carriage return alone indicates to the REDO command that the input is ready to be processed. If REDO is given as the first command, the text used is a blank line.

Syntax:

REDO

Example:

```
IOTT>d aall
-

*** UNRECOGNIZED COMMAND ( IOTTERR 6005 )
IOTT>redo
>d aall
>iete
>delete aall
>      rl
>delete alll
>      d
>
IOTT>delete all
*** PROGRAM STORAGE AREA IS EMPTY ( IOTTERR 3000 )
IOTT>
```

Instruction Summary

Instructions are inputs which can be used as single inputs, or as part of a program. As a single input, the instruction is executed immediately, similar to a command. As part of a program, the instruction executes when the program is initiated. All instructions can be given as program statements by prefixing each with an integer line number.

There are five categories of input Buffer Manipulation Instructions available to the user:

- Test Environment Instructions (TEI),
- Buffer Manipulation Instructions (BMI),
- Predefined I/O Request Instructions (PIORI),
- HP-IB Device Adapter Program Instructions (HPIBPI),
- HP-CIO DMA Chain Instructions (HPCIOI).

The following is a list and brief description of instructions available in IOTT.

Test Environment Instructions

The following instructions are used to set the environment in which to use I/O Test Tool. Tasks which must be performed before an I/O request can be issued include selection of an I/O device to be tested, and a timeout value to be used against each I/O request. There is also an instruction to enter PAUSE mode whenever an error is detected during execution of I/O Test Tool instruction.

ERRPAUSE

The **ERRPAUSE** instruction causes I/O Test Tool to enter **PAUSE** mode after an error is detected during execution of any instruction. The only parameters for this instruction are the reserved words **ON** or **OFF**. Initially the error pause is disabled. The **RESUME** command can be used to continue execution. If the user is executing a program, the **RESUME** command causes the program to continue.

Syntax:

```
ERRPAUSE ON
ERRPAUSE OFF
```

Example:

```
IOTT>errpause on
IOTT>defbuff buff,3000,word
IOTT>defbuff buff2,4000,word
IOTT>altbuff buff1(0),0,0,0
IOTT>altbuff buff2(0),1,2,3
IOTT>compbuff buff1(0),buff2(0),1
BUFF1(0) = 0
BUFF2(0) = 1
IOTT>resume
IOTT>
```

RELDEVICE

The RELDEVICE instruction releases control of the device to the system. If the device to be released is the current device, all pending unblocked I/O requests are aborted. In this case, the user must perform another SETDEVICE instruction before requesting an I/O to be performed.

Syntax:

```
RELDEVICE LDEV={ldev number}
RELDEVICE PDEV={CA#} [.DA# [.Device#]]
```

Example:

```
IOTT>setdevice ldev=9
IOTT>setdevice ldev=3
IOTT>showdev
LDEV      PDEV      PRODUCT
3         1.6.2     HP7978
9         1.2.3     HP7933

Current device
LDEV      PDEV      PRODUCT
3         1.6.2     HP7978
IOTT>reldevice ldev=9
IOTT>showdev

LDEV      PDEV      PRODUCT
3         1.6.2     HP7978

Current device
LDEV      PDEV      PRODUCT
3         1.6.2     HP7978
IOTT>
```


SETDEVICE

The SETDEVICE instruction allows the user to specify the device to be tested. The parameter given is used to identify the device. This parameter can be given as a logical device number (LDEV), which is specified by the operating system. Logical device numbers are available for peripheral devices only. Intermediate hardware is not allocated a logical device number. Therefore, to access an intermediate hardware device, the user must give a physical path identifier (PDEV). The physical path identifier describes the path to the intermediate device by specifying as appropriate, the channel adapter number, device adapter number, and device number. The user only needs to give enough information to locate the device.

When this instruction is executed, I/O Test Tool requests exclusive access to the device. This means that the current session of I/O Test Tool is the only process accessing that device. If the device is currently being tested by another diagnostic, an error message is displayed and the instruction will not be executed. There are situations where exclusive access to the device cannot be obtained, for example, devices on the cold load path. For these devices, exclusive access will not be granted. Therefore, only non-data destructive I/O requests can be performed. If any of the above situations occur, a warning message is shown to the user.

Syntax:

```
SETDEVICE LDEV={ldev number}  
SETDEVICE PDEV={CA#}[.DA#][.Device#]
```

Limitations:

Exclusive access to cold load path devices and system disks cannot be obtained.

Example:

```
IOTT>setdevice pdev=1.2.3  
IOTT>setdevice ldev=3  
IOTT>showdev  
LDEV          PDEV          PRODUCT  
3             1.6.2         hp79780  
9             1.2.3         hp7933  
  
Current device  
LDEV          PDEV          PRODUCT  
3             1.6.2         hp7978  
IOTT>
```

SETTIMER

The SETTIMER instruction allows the user to specify the maximum length of time in seconds that I/O Test Tool waits for an I/O request to complete. If an I/O completion message has not been received from the I/O system within the specified length, I/O Test Tool aborts the request and displays an error message. The initial timeout value is 60 seconds. The instruction SHOWPARM displays the current value of the timeout.

Syntax:

SETTIMER {value}

Optional Parameters:

value An integer value is used to specify a data value. This value can be preceded by a base, identifier (\$=HEX, %=OCT, #=DEC).

Example:

```
IOTT>settimer 30
IOTT>showparm
Buffer Name      : No Buffer has been specified
Auxiliary Buffer : No Buffer has been specified
Option1         : No Option has been specified
Current Address  : Reserved area will be accessed
Status Buffer    : No Status Buffer has been specified
Timeout         : 30 seconds
Einc value      : 0
IOTT>
```

SHOWDEV

The **SHOWDEV** instruction displays all devices which have been allocated to I/O Test Tool. The current device being accessed is also indicated. The data displayed are the Logical device number (LDEV), if applicable, and Physical path number (PDEV) along with the product number of the device. A “*” mark will appear in front of the LDEV number, if exclusive access was not granted for that device.

Syntax:

SHOWDEV

Example:

```
IOTT>setdevice ldev=9
IOTT>setdevice ldev=3
IOTT>showdev
LDEV      PDEV      PRODUCT
3         1.6.2      hp7978
9         1.2.3      hp7933

Current device
LDEV      PDEV      PRODUCT
3         1.6.2      hp7978
IOTT>
```

Buffer Manipulation Instructions

There are two types of I/O buffers available to the user: integer buffer and byte buffers. Integer buffers contain 32-bit entities, which can span from -2147483648 to 2147483647. Byte buffers contain 8-bit entities which can span 0 to 255. The user declares the I/O buffers and the length by the instruction `DEFBUFF`. Each buffer must be defined before it can be referenced by other instructions. The buffer name has a maximum of eight alphanumeric characters and must start with an alpha character. The I/O buffers can be deallocated using the `RELBUFF` instruction.

Buffer function instructions give the user the ability to fill, modify, and display data which was used for the I/O request. For functions which involve two buffers, both buffers must be of the same type.



ADJBUFF

The ADJBUFF instruction allows the user to adjust values within the specified I/O buffer. The first parameter of this instruction is the buffer name and index when the buffer adjustment is to begin. The second parameter will be the value by which the contents of each element within the buffer is to be adjusted. This value is added to the contents of the buffer, and the result replaces the original value. Positive and negative values are allowed. The third parameter specifies the number of buffer elements to be altered. If this parameter is not given, only one buffer element will be altered.

Syntax:

ADJBUFF {buffer name}({index}},{value},{count]

Optional Parameters:

count An integer value greater than zero. Used to specify number of times operation is to be performed.

Example:

```
IOTT>debuff buff,300 word
IOTT>altbuff (0),1,2,3,4,5
IOTT>dbuff buff 0/4
BUFF(0):      1      2      3      4      5
IOTT>adjbuff buff(0),2,3
IOTT>dbuff buff 0/4
BUFF(0):      3      4      5      4      5
IOTT>adjbuff buff(2),-1
IOTT>dbuff buff 0/4
BUFF(0):      3      4      4      4      5
IOTT>
```

ALTBUFF

The **ALTBUFF** instruction allows the user to modify individual elements in an I/O buffer. The first parameter for this instruction is the name of the buffer to be accessed and the index where the alteration is to start. The values following are the new values to be written into the I/O buffer. These values must be within the span of the buffer type. These values can be given as literal values, or expressions containing literal values. The alteration of buffer elements occurs sequentially for all values given. For a BYTE buffer, the list of values can be given in the form of ASCII text. At least one value must be given. All values are range checked before the buffer is altered. If any errors are detected, the buffer remains untouched.

Syntax:

```
ALTBUFF {buffer name}({index}},{value},[value]
ALTBUFF {buffer name}({index}),"ascii text"
```

Optional Parameters:

value An integer value is used to specify a data value. This value can be preceded by a base, identifier (\$=HEX, %=OCT, #=DEC).

Example:

```
IOTT>defbuff buff,300,word
IOTT>altbuff buff(0),1,2,3,4,5
IOTT>dbuff buff 0/4
BUFF(0)            1            2            3            4            5
IOTT>altbuff buff(0),20,40,60,80
IOTT>dbuff buff 0/4
BUFF(0)            20          40          60          80          5
IOTT>
```

COMPBUFF

The **COMPBUFF** instruction allows the user to compare the contents of two buffers. The first two parameters specify the buffers involved in the comparison and the indices within each buffer where the comparison is to begin. Both buffers must be of the same type. The next parameter specifies the length of the comparison. This value is in units of the buffer type. The optional count specifies the maximum number of mis-comparisons to be shown. If the optional value is not given, all mis-comparisons found are shown. An optional value of zero disables the displaying of information. The next optional parameter can be the reserved word **DIFF** or **SIMILAR**. If the **DIFF** reserved word is given, the buffers are compared for differences. Any values which do not match are displayed. If the **SIMILAR** reserved word is given, the buffers are compared for similarities. Any values which match are displayed. If no option is given, the **DIFF** option is assumed. This instruction also accepts a display mode to specify the base of which the values are to be displayed. The input format of **COMPBUFF** requires that each command statement consist of only one line.

Syntax:

```
COMPBUFF {buffer}<{index}>,{buffer}<{index}>,{length},{count}

[,diff  ]:[display mode]
[ similar]
```

Optional Parameters:

count An integer value greater than zero. Used to specify number of times operation is to be performed.

display mode A mode in which values are to be displayed: HEX, OCT or DEC.

Example:

```
IOTT>defbuff buff1,256,word
IOTT>defbuff buff2,200,word
IOTT>altbuff buff1(0),0,0,0,0,0
IOTT>altbuff buff2(0),1,2,3,4,5
IOTT>compbuff buff1(0),buff2(0),5,2    {Show first 2 mismatch}
buff1(0) = 0
buff2(0) = 1
buff1(1) = 0
buff2(1) = 2
IOTT>altbuff buff2(0),0,0,0,0,0
IOTT>compbuff buff1(0),buff2(0),5       {Identical buffers}
IOTT>
```

COPYBUFF

The COPYBUFF instruction allows the user to copy the contents of one buffer to another buffer. The contents of the first buffer, starting at the index given, is copied into the second buffer starting at the specified index. The LENGTH parameter specifies the number of elements to be copied. This parameter is in buffer element units. If the optional count is given, the copying process will repeat starting from the end of the last copy, for the count given. Both buffers must be of the same type. If the count parameter is not given, the copy operation will be performed once. The input format of COPYBUFF requires that each complete command statement consist of only one line.

Syntax:

COPYBUFF {buffer}({index}},{buffer}({index}},{length},{count]

Optional Parameters:

count An integer value greater than zero. Used to specify number of times operation is to be performed.

Example:

```
IOTT>defbuff buff,256,word
IOTT>defbuff another,200,word
IOTT>altbuff buff(0),1,2,3,4,5
IOTT>altbuff another(0),0,0,0,0,0
IOTT>copybuff buff(0),another(0),3,1 {Copy pattern once}
IOTT>dbuff another 0/4
ANOTHER(0):     1     2     3     0     0
IOTT>
```


DBUFF

The DBUFF instruction allows the user to display the contents of the specified I/O buffer. The first parameter of this instruction is the name of the buffer to be displayed. The following optional parameters are an index number or a range of index numbers. This specifies what part of the buffer is to be displayed. If this parameter is not given, the entire buffer is displayed. The user may also optionally specify the base in which the data values are to be displayed. The reserved words HEX, DEC, and OCT should be used when specifying a base. If a base is not specified, a byte buffer is displayed in ASCII, and an integer buffer is displayed as decimal values. The user may enter a user interrupt (CTRL) /Y to terminate this instruction.

Syntax:

```
DBUFF {buffer name} [:display mode]
DBUFF {buffer name} [index] [:display mode]
DBUFF {buffer name} [index/index] [:display mode]
```

Optional Parameters:

display mode A mode in which values are to be displayed: HEX, OCT or DEC.

index A number or range of numbers that specifies what part of the buffer is to be displayed.

Example:

```
IOTT>defbuff bbuff,30,byte
IOTT>defbuff wbuff,30,word
IOTT>altbuff wbuff(0),0,1,2,3,4,5,0
IOTT>dbuff wbuff 0/6
WBUFF(0):  0  1  2  3  4
WBUFF(5):  5  0
IOTT>altbuff bbuff(0),0,1,2,3,4,5,0
IOTT>dbuff dbuff 0/6: hex
BBUFF(0)  $0  $1  $2  $3  $4
BBUFF(5)   $5  $0
IOTT>
```

DEFBUFF

The DEFBUFF instruction allows the user to define an I/O buffer. The first parameter is the buffer name, which must be a maximum of eight alphanumeric characters, starting with an alpha character. The second parameter is the length of the buffer. For a byte buffer, the length is specified in the number of bytes (8-bit). For an integer buffer, the length is specified in the number of words (32-bit elements). The third parameter specifies the type of buffer to be allocated by giving the reserved word BYTE or WORD. The I/O buffer itself is not initialized to any specific value. If the user redefines the buffer, there is no guarantee that the old data will remain.

The user can use the STATUS option to specify that this particular buffer is to be used to store the last hardware status returned from the I/O request. Only one buffer can have this option at any one time. The status buffer is altered only if the I/O request fails. If the size of the buffer is smaller than the minimum length of a hardware status, the length will be modified to the minimum length.

Syntax:

```
DEFBUFF {buffer name},{length},{BYTE} [:STATUS]
DEFBUFF {buffer name},{length},{WORD} [:STATUS]
```

Example:

```
IOTT>defbuff buff,4000,byte
IOTT>defbuff status,20,byte;status
IOTT>defbuff name,1000,word
IOTT>showbuff
BUFFER NAME      TYPE      LENGTH
  BUFF          BYTE      4000
  STATUS        BYTE       32
  NAME          WORD      1000
IOTT>
```

FILLBUFF

The **FILLBUFF** instruction allows the user to fill an I/O buffer with a specific value. The first parameter specifies the buffer name and index where the filling is to start. The **VALUE** parameter is the value to be inserted into the buffer. This value must be within the span of the buffer type. If an option **COUNT** parameter is given, the filling occurs for the indicated count. If the count is not given, the entire buffer is filled.

Syntax:

FILLBUFF {buffer}({index}),(value),[count]

Optional Parameters:

count An integer value greater than zero. Used to specify number of times operation is to be performed.

Example:

```
IOTT>defbuff buff,256,word
IOTT>altbuff buff(0),256,256,256,256,256
IOTT>fillbuff buff(0),1,3
IOTT>dbuff buff 0/4
BUFF(0):   1     1     1     256     256
IOTT>
```

FINCBUFF

The **FINCBUFF** instruction incrementally fills the specified buffer. The first parameter is the name of the buffer and the index where the filling is to begin. The **START** parameter specifies the starting value, and the **END** parameter specifies the ending value for the buffer filling. The starting value increments in values of the **INC** parameter. If the **INC** parameter is not given, an increment value of 1 is used. The resulting pattern repeats until the entire buffer is filled. For byte buffers, starting and ending values can be an ASCII character enclosed in quotes (" ").

Syntax:

```
FINCBUFF {buffer name}({index}},{start},{end}[,inc]
```

Example:

```
IOTT>defbuff buff,40,word
IOTT>fincbuff buff(0),1,9,3
IOTT>dbuff buff 0/10
BUFF( 0):    1    3    6    9    1
BUFF( 5):    3    6    9    1    3
BUFF(10):    9
IOTT>
```

RELBUFF

The RELBUFF instruction will release and purge the buffer specified. The only parameter for this instruction is the buffer name.

Syntax:

RELBUFF {buffer name}

Example:

```
IOTT>defbuff buff,20,byte
IOTT>defbuff status,30,word : status
IOTT>showbuff
BUFFER NAME      TYPE      LENGTH
  BUFF           BYTE         20
  STATUS         WORD         30
IOTT>relbuff buff
IOTT>showbuff
BUFFER NAME      TYPE      LENGTH
  STATUS         WORD         30
IOTT>
```

SHOWBUFF

The SHOWBUFF instruction lists the name, length, and type of all buffers declared. The length of the buffer will be in the units of the buffer element.

Syntax:

SHOWBUFF

Example:

```
IOTT>defbuff buff,4000,word
IOTT>defbuff status,6000,byte
IOTT>showbuff
BUFFER NAME      TYPE      LENGTH
  BUFF           WORD       4000
  STATUS         BYTE       6000
IOTT>
```

Predefined I/O Request Instructions

The following instructions will allow the user to specify all information necessary for predefined I/O request, as well as an instruction for issuing the request itself. Before issuing the I/O request, the user can use any of the following instructions to give all information needed for the request.

SETADDR	Indicates the location by address on the device where the I/O is to occur.
SETDATA	Indicates the I/O buffer and length to be used for the I/O request.
SETOPTION	Indicates any additional options needed for the I/O request.
EINCADDR	Indicates that the address given by SETADDR is to be incremented after each I/O request.

Refer to predefined I/O Requests in this section for further information required for each type of predefined I/O requests. The information setting functions are separated into many instructions, so that the user will not need to retype each parameter for each request.

After all the information has been given, the I/O request is initiated with the EXECUTE instruction. If the I/O request fails, the hardware status is stored in the user defined status buffer. The instruction DSTATUS is available to decode the hardware status into text.

ABORTIO

The **ABORTIO** instruction aborts all pending unblocked I/O requests for the current device. This instruction aborts those requests issued by I/O Test Tool.

Syntax:

ABORTIO

Example:

```
IOTT>setdevice ldev=7
IOTT>defbuff status,32,byte:status
IOTT>defbuff buffer,256,byte
IOTT>setdata buffer,256
IOTT>execute read,21:unblock
IOTT>execute ioreult
I/O HAS NOT YET COMPLETED (DSSERR 510)
IOTT>abortio
```



EINCADDR

The EINCADDR instruction increments the address given by SETADDR each time the EXECUTE instruction issues an I/O request. The value by which the address increments can be positive or negative. The incremental function is always performed, even if the SETADDR value is not needed for the I/O requested. This instruction can be disabled by giving an increment value of zero. If an INCADDR instruction has not been performed previously, a DESCRIBE I/O request is issued to obtain the maximum address for that device. The DESCRIBE is performed after the EXECUTE instruction has completed. An error message is displayed if the INCADDR instruction is requested with the SETADDR instruction indicating the reserved area on the disk.

Syntax:

EINCADDR {value}

Example:

```
IOTT>setdevice ldev=3 {CS80 disk}
IOTT>defbuff buff,256,byte
IOTT>defbuff status,20,byte:status
IOTT>setdata buff,256
IOTT>setaddr 2000
IOTT>eincaddr 1
IOTT>execute read      {Read sector 2000}
IOTT>showparm
Buffer Name           : BUFF
Transfer Length       : 256
Auxiliary Buffer       : No Buffer has been specified
Option1               : No Option has been given
Current Address       : 2001
Status Buffer          : STATUS
Timeout               : 60 seconds
Einc Value            : 1
Error Pause           : OFF
IOTT>execute read      {Read the next sector}
IOTT>showparm
Buffer Name           : BUFF
Transfer length       : 256
Auxiliary Buffer       : No Buffer has been specified
Option1               : No Option has been given
Current Address       : 2002
Status Buffer          : STATUS
Timeout               : 60 seconds
Einc Value            : 1
Error Pause           : OFF
IOTT>
```

EXECUTE

The **EXECUTE** instruction initiates a predefined I/O request. The first parameter can be a reserved word to indicate the type of I/O to be performed. If any necessary information is not given, an appropriate error message is displayed, and the I/O request is not issued. An optional count parameter can be given to indicate the number of times the I/O request is to be issued. If this count parameter is not given, the I/O request is issued once. All I/O requests issued are performed in **BLOCK** mode, indicating that I/O Test Tool waits until the I/O request completes before processing the next input. The user may issue unblocked I/O requests before processing the next input. The **UNBLOCK** option is not available for all devices.

If an I/O error is encountered, the **EXECUTE** instruction terminates immediately. A description of the error is displayed to the user, and the hardware status returned can be found in the user-defined status buffer. If the status buffer defined by the user is not large enough to hold the hardware status, the status buffer is redefined to the correct length. If appropriate, the I/O transfer count and the number of hardware status bytes obtained are displayed.

Syntax:

```
EXECUTE {function}[,count][: UNBLOCK]
```

Optional Parameters:

count An integer value greater than zero used to specify the number of times an operation is to be performed.

Example:

```
IOTT>setdevice ldev=3     {CS80 disk}
IOTT>defbuff status,20,byte:status
IOTT>execute reqstat     {Request status}
IOTT>
```

DSTATUS

The **DSTATUS** instruction allows the user to display the hardware status stored within the user defined status buffer in text. I/O Test Tool assumes that the decoding format to be used is that of the current device selected. The display of the status information is identical to that produced by the device diagnostic program. The diagnostic program will be performing that actual translation on behalf of I/O Test Tool. If the diagnostic program is not present on the system, an error message is displayed to the user, and this instruction will terminate.

Syntax:

DSTATUS

Example:

```
IOTT>setdevice ldev=3      {CS80 disk}
IOTT>defbuff status,20,byte:status
IOTT>fillbuff status(0),0
IOTT>execute reqstat      {Request status}
IOTT>dbuff status 0/9 : hex

STATUS( 0):$F $FF $0 $0 $0 $0 $0 $0 $0 $0 $0 $0 $0 $0
STATUS(15):$0 $0 $0 $0 $0 $0 $1
IOTT>dstatus

      Status =

      Unit = 15   Volume = 0

      No units with status pending

      New target address is :
      Blocked address = 1
IOTT>
```

INCADDR

The INCADDR instruction increments the address given by SETADDR by the value specified. This value can be a positive or negative value. The first time INCADDR is entered, I/O Test Tool issues a DESCRIBE I/O request to the device to obtain the maximum address for the device. If the incremental shift exceeds the maximum address, an error message is displayed to the user, and the original value remains. An error message is displayed if the INCADDR instruction is requested with the SETADDR instruction indicating the reserved area on the disk.

Syntax:

INCADDR {value}

Example:

```
IOTT>setdevice ldev=3 {CS80 disk}
IOTT>setaddr 0          {set logical address 0}
IOTT>incaddr 20         {set logical address 0+20}
IOTT>showparm
Buffer Name      : No Buffer has been specified
Auxiliary Buffer : No Buffer has been specified
Option1         : No Option has been specified
Current Address  : 20
Status Buffer     : No Status Buffer has been specified
Timeout         : 60
Einc value       : 0
Error Pause     : OFF
IOTT>
```

RESETIO

The **RESETIO** instruction resets the I/O software and hardware to power-on state for the current device. Any pending I/O request is aborted. If exclusive access to the current device was not obtained, this instruction will fail.

Syntax:

RESETIO

Example:

```
IOTT>setdevice ldev=7 {A tape device}
IOTT>debuff status,32,byte:status
IOTT>debuff buffer,256,byte
IOTT>setdata buffer,256
IOTT>execute read,21:unblock
IOTT>execute ioreult
I/O HAS NOT YET COMPLETED (DSSERR 510)
IOTT>resetio
```

PROTECT

The PROTECT instruction will allow the user to enable or disable the protection system within I/O Test Tool, which requires the user to have exclusive access to the current device before issuing a data destructive request. Initially, I/O Test Tool will have the protection system enabled (ON). The PROTECT OFF instruction will have no effect on the I/O requests using the RDI, WDI, RFSI, LOCVERI, HPIB, and HPCIO predefined identifiers. To use these particular I/O requests successfully, the user must have exclusive access to the device.

Note



The user is advised to take the proper precautions before issuing any data destructive I/O request, and to perform the PROTECT ON instruction at the earliest opportunity.

Input

```
IOTT> PROTECT ON  
IOTT> PROTECT OFF
```

Example

```
IOTT> EXECUTE WRITE  
EXCLUSIVE ACCESS TO THE DEVICE REQUESTED (IOTTERR 5578)  
IOTT> PROTECT OFF  
IOTT> EXECUTE WRITE  
IOTT>
```

SETADDR

The **SETADDR** instruction allows the user to specify the location on the device where the function is to be performed. This instruction is only be used by devices which understand the concept of an address. The optional parameter can be either a literal value or an expression of literal values which are evaluated to the 32-bit logical address. This address can also be given in cylinder-head-sector format (i.e. for a disk, the address is a sector number). If no parameter is given, the reserved area on the disk is used. This instruction will not perform any bounds checking of the address given.

Syntax:

```
SETADDR CLY={cylinder};HEAD={head};SECT={sect}  
SETADDR [value]
```

Optional Parameters:

value An integer value is used to specify a data value. This value can be preceded by a base, identifier (\$=HEX, %=OCT, #=DEC).

Example:

```
IOTT>setaddr cyl=0;head=0;sect=0  
IOTT>showparms  
Buffer Name      : No data buffer has been specified  
Auxiliary Buffer: No buffer has been specified  
Option1         : No option has been specified  
Current Address : cyl = 0; head = 0; sect = 0  
Status Buffer    : No Status Buffer has been defined  
Timeout        : 60 seconds  
Einc value      : 0  
Error Pause     : OFF  
IOTT>
```

SETDATA

The SETDATA instruction allows the user to specify the I/O buffer and transfer length that is to be used for the I/O request. The first parameter is the name of the buffer involved in the I/O. The transfer always starts at the first element, and continues for the length specified. The length is assumed to be given in buffer units. If a word buffer is given, the length is assumed to be a word count. An auxiliary buffer can be given for I/O requests which require two I/O buffers.

Syntax:

SETDATA {buffer},{length}[,{aux buffer, length}]

Example:

```
IOTT>debuff buff,3000,word
IOTT>setdata buff,300      {use only 300 words}
IOTT>showparms
Buffer Name      : BUFF
Transfer Length  : 300
Auxiliary Buffer  : No Buffer has been specified
Option1         : No options has been given
Current Address  : Reserved area will be accessed
Status Buffer     : No Status Buffer has been specified
Timeout         : 60 seconds
Einc value      : 0
Error Pause     : OFF
IOTT>
```


SETOPTION

The SETOPTION instruction allows the user to specify the options necessary to execute some of the predefined I/O requests. The user can specify up to two options separated by a comma. The user can specify an option as a literal value, or as a reserved word option. Please refer to predefined I/O Requests in this section for a list of acceptable reserved words for various predefined I/O requests.

Syntax:

SETOPTION {option}[,option]

Example:

```
IOTT>setdevice ldev=3      {CS80 disk}
IOTT>defbuff status,30,byte:status
IOTT>setoption 0
IOTT>showparm
Buffer Name      : No data buffer has been specified
Auxiliary Buffer  : No Buffer has been specified
Option 1         : 0
Option 2         : No option has been specified
Current Address   : Reserved area will be accessed
Status buffer     : STATUS
Timeout          : 60 seconds
Einc value        : 0
Error Pause       : OFF
IOTT>execute selftest      {Initiate selftest zero}
IOTT>
```

SHOWPARAM

The SHOWPARAM instruction displays all the information which are presently set to be used for the next predefined I/O request. The instructions which indicated the displayed information are SETDATA, SETADDR, SETOPTION, SETTIMER, and EINCADDR. The name of the I/O buffer which is used to store the I/O hardware status information is also given.

Syntax:

SHOWPARAM

Example:

```
IOTT>defbuff buff,3000,word
IOTT>defbuff status,20,byte:status
IOTT>setdata buff,300
IOTT>setaddr 0
IOTT>setoption read
IOTT>showparam
Buffer Name      : BUFF
Transfer Length  : 300
Auxiliary Buffer  : No buffer has been specified
Option 1         : READ
Option 2         : No Option has been specified
Current Address  : 0
Status Buffer     : STATUS
Timeout         : 60 seconds
Einc value       : 0
Error Pause      : OFF
IOTT>
```

HP-IB Device Adapter Program Instructions

I/O Test Tool provides instructions for the user to create unique HP-IB device adapter programs (DAP). This will give the user more control over the protocol between the HP-IB Device Adapter and a peripheral device. The format used to create the HP-IB Device Adapter program is:

SETHPIB:	Indicates the start of HP-IB program
HP-IB:	Device Adapter Program Instructions
ENDHPIB:	Indicates the end of the HP-IB program Declaration.

Each HP-IB DAP program must start with a **SETHPIB** instruction, and end with an **ENDHPIB** instruction. Any instruction can be placed between these instructions, except another **SETHPIB** instruction. It is recommended that the DAP instructions be entered in program mode. Instructions which require relative offsets, such as **DSJ** and **JUMP**, must be given in program mode, since I/O Test Tool will be calculating these values. The instruction **SHOWHPIB** can be used to monitor the progression of the DAP build.

The user can execute the HP-IB program by using an **EXECUTE** instruction with an "HP-IB" function identifier. Values set by **SETADDR** and **SETDATA** will not be used during the execution of the HP-IB program. The device specified by the **SETDEVICE** instruction must point to a peripheral. After the I/O request has completed, a software **RESET** function will be issued to the I/O software associated with the peripheral device selected. This will permit the I/O software to re-download its DAP program onto the HP-IB device adapter.

The status values returned from the I/O request will be stored in the user defined status buffer. The first byte of this buffer is reserved for the **HSTAT** value. The remainder of the status information is user defined. The user will not be able to redefine the first byte of the status buffer.

For further information on HP-IB Device Adapter Programs, refer to the HP 27110B Software External Reference Manual. Since unique HP-IB Device Adapter Programs have the potential of forcing the device into an abnormal state, such program usage will be available **ONLY IN SINGLE USER MODE**.

CASEJUMP

The CASEJUMP instruction will append the case_jump DAP command to the DAP currently being built. The first value parameter given will be appended to the "PB" buffer at the next available space. This will be the value which will be used by the DAP to evaluate the case_jump. The following parameters specify the destinations for the jump, which will be performed depending on the value given. A value of zero will cause a jump to the first line number parameter given. A value of one will cause a jump to the second line number parameter. The user must specify at least one jump location. The line number parameters used to specify the jump location must point to an HP-IB instruction. The actual relative offset values will be filled in by the ENDHPIB instruction. Until then zeros will be placed into the DAP as place holders. The case_jump causes a transfer of control to a place in the DAP indicated by a value in the parameter block.

Syntax:

```
{line number} CASEJUMP {value},{line number}[ ,line number]
```

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 sethpiB
IOTT>20 casejump 2,30,40,50
IOTT>30 halt 3,3
IOTT>40 halt 3,4
IOTT>50 halt 3,5
IOTT>run
IOTT>showhpiB
```

Device Adapter Program Build in Progress

Device Adapter Program. Current length = 15 bytes, maximum = 512 bytes

```
DAP( 0):  10  1  3  0  0  0  64  3  3  64  3  4
DAP(12):  64  3  5
```

Parameter Block. Current length = 1 bytes, maximum = 62 bytes.

```
PB(0):      2
```

Status Buffer. Required length = 3 bytes, maximum = 62 bytes.

```
WARNING: Status Buffer has not been defined.
IOTT>endhpib
IOTT>showhpib

Device Adapter Program has been Built.

Device Adapter Program. Current length = 15 bytes, maximum = 512 bytes

DAP( 0):  10  1  3  0  3  6 64  3  3 64  3  4
DAP(12): 64  3  5

Parameter Block. Current length = 1 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.
IOTT>
```

CLEAR

The **CLEAR** instruction will append the clear DAP command to the HP-IB DAP currently being built. The **VALUE** parameter must be between the values of 0 to 255.

Clear causes the AMIGO clear command and the HP-IB selected device clear command to be sent to the device. This causes the device to reset itself per value given.

Syntax:

CLEAR {value}

Example:

```
IOTT>sethpb  
IOTT>clear 0  
IOTT>  
showhpb : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes

DAP(0): \$3C \$0

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.
IOTT>



CRCCOMP

The CRCCOMP instruction will append the crc.compare command to the DAP currently being built. The LINE NUMBER parameter indicates the jump destination to be used if CRC does not match. This parameter must point to an HP-IB instruction. The relative offset needed by this instruction will be filled in by the ENDPBIB instruction. Until then, zeros will be placed into the DAP as place holders. The crc is read by the device adapter from the device and the device adapter and compared. If they match, the transaction continues. Otherwise, transaction will jump to relative offset given.

Syntax:

```
{line number} CRCCOMP {line number}
```

Example:

```
IOTT>10sethpbib  
IOTT>20 crccomp 30  
IOTT>30 halt 3,3  
IOTT>40 endhpbib  
IOTT>run  
IOTT>showhpbib : hex
```

Device Adapter Program has been Built.

Device Adapter Program. Current length = 5 bytes, maximum = 512 bytes

DAP(0): \$36 \$0 \$40 \$3 \$3

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 3 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.

IOTT>

CRCINIT

The **CRCINIT** instruction will append the `Crc_initiate` command to the DAP currently being built. `Crc_initiate` causes the CRC in both the device adapter's and device's ABI's to be reset. No error check is done.

Syntax:

CRCINIT

Example:

```
IOTT>sethpiib  
IOTT>crcinit  
IOTT>showhpiib
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 1 bytes, maximum = 512 bytes

DAP(0): 52

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.
IOTT>

CRCWRITE

The **CRCWRITE** instruction will append the **Crc.write** command to the DAP currently being built. **Crc.write** command causes the device adapter's and device's CRC bytes to be sent to the device.

Syntax:

CRCWRITE

Example:

```
IOTT>sethpid  
IOTT>crcwrite  
IOTT>showhpid: hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 1 bytes, maximum = 512 bytes.

DAP(0): \$38

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

```
IOTT>
```

DSJ

The DSJ instruction will append the DSJ command to the DAP currently being built. Instruction will allow conditional jump to another part of the program depending on the value of the one byte of status returned from the device. The SINDEX parameter specifies the index within the user defined status buffer where the one byte of status is to be stored. The following parameters specify the destinations for the jump, which will be performed depending on the value returned. A status value of zero will cause a jump to the first line number parameter given. A status value of one will cause a jump to the second line number parameter. The user must specify at least one jump location. The line number parameters used to specify the jump location must point to a HP-IB program instruction. The actual relative offset values will be filled in by the ENDPBIB instruction. Until then zeros will be placed into the DAP as place holders. The DSJ command reads one byte of status from the device and puts it into the status buffer. A case jump is performed using the value of status. The input of this command must be entered as one line.

Syntax:

```
{line number} DSJ {sindex},{line number},[line number],[line number]
```

Optional Parameters:

line number An integer value greater than zero but less than 9999. Used to specify program input and line number of program.

Example:

```
IOTT>10 sethpb  
IOTT>20 dsj 1,30,40,50  
IOTT>30 halt 2,1  
IOTT>50 halt 2,3  
IOTT>run  
IOTT>showhpb : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 15 bytes, maximum = 512 bytes

```
DAP( 0):  $3E  $1  $3  $0  $0  $0  $40  $2  $1  $40  $2  $2  
DAP(12):  $40  $2  $3
```

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 2 bytes, maximum = 62 bytes.

```
WARNING: Status Buffer has not been defined.
IOTT>endhpib
IOTT>showhpib : hex

Device Adapter Program has been Built.

Device Adapter Program.  Current length = 15 bytes, maximum = 512 bytes

DAP( 0): $3E  $1  $3  $6  $40  $2  $1  $40  $2  $1  $40  $2  $2
DAP(12): $40  $2  $3

Parameter Block.  Current length = 0 bytes, maximum = 62 bytes.

Status Buffer.  Required length = 2 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.
IOTT>
```

ENDHPIB

The **ENDHPIB** instruction indicates that the building of a HP-IB Device Adapter Program (DAP) has completed. If the DAP contains any instructions which utilize relative or absolute offsets, such as DSJ or JUMP, this instruction will calculate these values and fill in the reserved areas within the DAP. If any problems are encountered, appropriate error message will be displayed and this instruction will terminate.

Syntax:

ENDHPIB

Example:

```
IOTT>defbuff status,62,byte:status
IOTT>fillbuff status(0),0
IOTT>sethplib
IOTT>clear 0
IOTT>halt 3,1
IOTT>endhplib
IOTT>showhplib : hex
```

Device Adapter Program has been Built.

Device Adapter Program. Current length = 5 bytes, maximum = 512 bytes

DAP(0): \$3C \$0 \$40 \$3 \$1

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 3 bytes, maximum = 62 bytes.

```
STATUS(0) $0   $0   $0
IOTT>
```

HALT

The **HALT** instruction will append the **HALT** command to the DAP currently being built. The **STATUS LENGTH** parameter will specify the number of bytes to be returned when the transaction terminates. These bytes will be stored in the user defined status buffer. The **HSTAT** parameter is the halt status value required by the DAP instruction. Both of these parameters must be within the range of 0 to 255.

HALT causes the transaction to be terminated without error. There may be more than one **HALT** within a program, but there should be at least one. The **HSTAT** value specified will be stored within the status buffer after the device adapter program completion, to indicate which **HALT** statement was executed.

Syntax:

HALT {status length},{hstat}

Example:

```
IOTT>defbuff status,62,byte:status
IOTT>fillbuff status(0),0
IOTT>sethpb
IOTT>halt 2,1
IOTT>endhpb
IOTT>showhpb : hex
```

Device Adapter Program has been Built.

Device Adapter Program. Current length = 3 bytes, maximum = 512 bytes

DAP(0): \$40 \$3E \$1

Parameter Block. Current length = 0 bytes, maximum = 62 bytes

Status Buffer. Required length = 2 bytes, maximum = 62 bytes

```
STATUS ( 0):    $0    $0
IOTT>
```

IDENTIFY

The IDENTIFY instruction will append the IDENTIFY command to the DAP currently being built. The SINDEXT parameter indicates the index within the user defined status buffer where the returned data is to be stored. The identify DAP command will return two (2) bytes of data; this causes the identify bytes to be read from the device and send to the status buffer.

Syntax:

IDENTIFY {sindex}

Example:

```
IOTT>sethpib
IOTT>identify 1
IOTT>showhpib
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes

DAP(0): 58 1

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 3 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.

IOTT>

JUMP

The **JUMP** instruction will append the jump.a command to the DAP currently being built. The **LINE NUMBER** parameter will specify the destination of the jump. The absolute address needed for this command within the device adapter program will be calculated by the **ENDHPIB** instruction. Until the **ENDHPIB** instruction is given, zeros will be inserted as place holders. The line number must reference an HP-IB device adapter program instruction, if any other statement is referenced an error message will be displayed and the instruction will not be appended to the current DAP.

Syntax:

```
{line number} JUMP {line number}
```

Example:

```
IOTT>10 sethpb  
IOTT>20 jump 50  
IOTT>30 halt 2,0  
IOTT>40 halt 2,1  
IOTT>50 jump 30  
IOTT>run  
IOTT>defbuff status,2,byte:status  
IOTT>showhpb : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 12 bytes, maximum = 512 byte

```
DAP( 0):  $2  $0  $0  $40  $2  $0  $40  $2  $1  $2  $0  $
```

Parameter Block. Current length = 0 bytes, maximum = 512 bytes.

Status Buffer. Required length = 2 bytes, maximum = 62 bytes.

```
STATUS(0): $0  $0
```

```
IOTT>endhpb  
IOTT>showhpb : hex
```

Device Adapter Program has been Built.

Device Adapter Program. Current length = 12 bytes, maximum = 512 byte

```
DAP( 0):  $2  $0  $9  $40  $2  $0  $40  $2  $1  $2  $0  $
```

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 2 bytes, maximum = 62 bytes.

```
STATUS(0): $0  $0  
IOTT>
```

ONTIMEOUT

The ONTIMEOUT instruction will append the on.timeout command to the DAP currently being built. The TIMEOUT parameter will be loaded into the device's timer. This is a 16-bit number in 10's of milliseconds. The SINDEK will specify the index into the user defined status buffer where the 2-byte PC value is to be stored if a timeout occurred. The LINE NUMBER parameter indicates which HP-IB instruction to jump to when the timeout occurs. On.timeout causes the device timer to be loaded with the time value and the address to be set up as a place to transfer when a device takes an excessive amount of time transferring data or responding to a parallel poll.

Input:

```
IOTT> {line number} ONTIMEOUT {timeout},{sindex},{line number}
```

Example:

```
IOTT>10 sethpiB
IOTT>20 ontimeout 60,1,30
IOTT>30 halt 3,2
IOTT>40 endhpiB
IOTT>run
IOTT>showhpiB : hex
```

Device Adapter Program has been Built.

Device Adapter Program. Current length = 8 bytes, maximum = 512 bytes.

DAP(0): \$44 \$0 \$3C \$1 \$0 \$40 \$3 \$2

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 3 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.

PINDEX

The PINDEX instruction will set the parameter block index to point to the value specified. The value given will be used as the index of the next available space in the "PB" buffer. This instruction is provided for those users who require control over the PINDEX entries required for some DAP instructions. If this instruction is used, the user is responsible for verifying that the contents of the "PB" buffer is correct. Use of this instruction may cause the values in the "PB" buffer to be overwritten.

Syntax:

PINDEX {value}

Example:

```
IOTT>debuff buff1,10,byte
IOTT>fillbuff buff1 (0),2
IOTT>debuff buff2,10,byte
IOTT>fillbuff buff2(0),4
IOTT>sethpb
IOTT>winterf buff1,5
IOTT>pindex3
IOTT>winterf buff2,6
IOTT>showhpb
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 6 bytes, maximum = 512 bytes.

DAP(0): 40, 5 1 40 6 4

Parameter Block. Current length = 9 bytes, maximum = 62 bytes.

PB(0): 2 2 2 4 4 4 4 4 4

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

RBURST

The RBURST instruction will append the read_burst command to the DAP currently being built. The SECONDARY parameter is device dependent and will be used as the secondary command on the HP-IB. The BUFFER NAME parameter specifies the I/O buffer to which the data is to be stored. The length of the transfer is given by the LENGTH parameter, which is assumed to be given in buffer units. The #BURST parameter indicates the number of burst involved in the transfer. The BURSTLEN parameter indicates the length of each burst in bytes (8-bits). Read_burst causes a sequence of data buffers to be sent from the device to the user with a break between each buffer.

Syntax:

RBURST {secondary},{buffer name},{length},{#burst},{burstlen}

Example:

```
IOTT>debuff buff,30,byte
IOTT>sethpb
IOTT>rburstm 3,buff,30,2,15
IOTT>showhpb
```



Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 4 bytes, maximum = 512 bytes.

DAP(0): 32 3 2 15

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

DMA Buffer Name = BUFF DMA Transfer Length = 30 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

RDATA

The RDATA instruction will append the read command to the current DAP being built. The SECONDARY parameter is device dependent and will be used as the secondary command on the HP-IB. The SINDEK specifies where in the user defined status buffer the data transfer is to begin for the length, in bytes (8-bits), given by the LENGTH parameter. Read causes the device to be addressed and a buffer of data to be sent from the device to the status buffer.

Syntax:

RDATA {secondary},{sindex},{length}

Example:

```
IOTT>sethpib
IOTT>rdata $5,1,5
IOTT>showhpib : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 4 bytes, maximum = 512 bytes.

DAP(0): \$22 \$5 \$5 \$1

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 7 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.

RDMA

The RDMA instruction will append the read.dma command to the DAP currently being built. The SECONDARY parameter is device dependent and will be used as the secondary command on the HP-IB. The data sent to the device will be obtained from the buffer specified by the BUFFER NAME parameter, for the length specified. The LENGTH parameter is assumed to be given in the buffer defined units. Read.dma causes the device to be addressed and a buffer of data to be obtained from the device. The data will be terminated by an EOI.

Syntax:

RDMA {secondary},{buffer name},{length}

Example:

```
IOTT>debuff buff,4000,byte
IOTT>sethpib
IOTT>rdma 15,buff,4000
IOTT>showhpib : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes.

DAP(0): \$16 \$F

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

DMA Buffer Name = BUFF DMA Transfer Length = 4000 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

SETHPIB

The SETHPIB instruction will indicate to I/O Test Tool that a building of HP-IB device adapter program (DAP) is to begin. The DAP which may have been built previously will be destroyed. Also, if a DAP was previously being built, a warning message will be displayed to indicate that the pending HP-IB DAP will be destroyed.

Among the tasks performed by this instruction is the allocation of two I/O buffers. One buffer, which has the buffer name "DAP", will hold the device adapter program while it is being built. The second buffer, which has the buffer name "PB", will hold the parameter block while the DAP is being built. These I/O buffers will be allocated as BYTE buffers of the maximum length acceptable by the HP-IB device adapter. These two buffers will be redefined if the user has created these buffers prior to executing these instructions. Since "DAP" and "PB" are I/O buffers, the ALTBUFF instruction may be used to alter any part of these buffers. This will allow the user to access HP-IB DAP commands which may not be available through I/O Test Tool.

Syntax:

SETHPIB

Example:

```
IOTT>sethpib
IOTT>showbuff
BUFFER NAME      TYPE      LENGTH
PB               BYTE      62
DAP              BYTE      512
IOTT>showhpib
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 0 bytes, maximum = 512 bytes

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

SHOWHPIB

The SHOWHPIB instruction will display the contents of the HP-IB device adapter program built through the HP-IB DAP instructions. The information displayed is the I/O buffer "DAP" which contains the device adapter program, the I/O Buffer "PB" which contains the parameter block for the I/O request, and the user-defined status buffer. The length of each buffer is also displayed, as well as the I/O buffer involved in the DMA transfer. Only the portions of the buffers with valid data will be displayed. The user may request the values be displayed in HEX, OCT, or DEC by specifying the appropriate option. If a display option is not given, decimal will be used.

Syntax:

SHOWHPIB [:display mode]

Optional Parameters:

display mode A mode in which values are to be displayed in HEX, OCT, or DEC.

Example:

```
IOTT>sethpb  
IOTT>showhpb
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 0 bytes, maximum = 512 bytes

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

TIMEOUTOFF

The **TIMEOUTOFF** instruction will append the `timeout.off` command to the DAP currently being built. The `timeout.off` causes the device's timeout address to be set false, thus preventing a transfer of control if the device's timer runs down.

Syntax:

TIMEOUTOFF

Example:

```
IOTT>sethpb  
IOTT>timeoutoff  
IOTT>showhpb
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 1 bytes, maximum = 512 bytes.

DAP(0): 68

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

TIMESTAMP

The **TIMESTAMP** instruction will append the `time.stamp` command to be DAP currently being built. The **SINDEX** parameter specifies the index within the user defined status buffer where the card's timer 2-byte value is to be placed; this causes the card's timer to be placed into the specified location in the status buffer. When the timer rolls over it goes from 65536 to 0.

Syntax:

TIMESTAMP {sindex}

Example:

```
IOTT>sethpb  
IOTT timestamp 3  
IOTT>showhpb : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes.

DAP(0): \$46 \$3

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 5 bytes, maximum = 62 bytes.

WARNING: Status Buffer has not been defined.

UNLOCK

The UNLOCK instruction will append the unlock instruction to the DAP currently being built. UNLOCK command causes the lock flag to be reset to permit commands other than case_jump, timeout, and jump to be interpreted.

Syntax:

UNLOCK

Example:

```
IOTT>sethpib  
IOTT>unlock  
IOTT>showhpib
```

Device Adapter Program in Progress.

Device Adapter Program. Current length = 1 bytes, maximum = 512 bytes.

DAP(0): 14

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

WAITPOLL

The WAITPOLL instruction will append a wait_poll or wait_poll_nb command to the DAP currently being built. If the NOBREAK option is given, the wait_poll_nb command will be appended, otherwise the wait_poll command will be used. The wait_poll command indicates that the device adapter should suspend execution of the device adapter control program until the device responds with an affirmative response to an HP-IB parallel poll operation.

Syntax:

WAITPOLL [:nobreak]

Example:

```
IOTT>sethpb  
IOTT>waitpoll  
IOTT>waitpoll : nobreak  
IOTT>showhpb
```

Device Adapter Program Build in Progress

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes

DAP(0): 18 20

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

WBURST

The **WBURST** instruction will append the `write_burst` or the `write_burst.e` command to the DAP currently being built. The **SECONDARY** parameter is device dependent and will be used as the secondary command on the HP-IB. The **BUFFER NAME** parameter will specify the I/O buffer from which the data is to be transferred. The **LENGTH** parameter, which is assumed to be given the buffer units, specifies the length of the transfer. The **#BURST** parameter specifies the number of bursts required by the transfer. If the **EOI** option is given, the `write_burst.e` command will be appended instead of the `write_burst` command. `Write_burst` command causes the device to be addressed and a sequence of data buffers from the host to be sent to the device with a break.

Syntax:

WBURST {secondary},{buffer name},{length},{#burst}[:eoi]

Example:

```
IOTT>sethpb  
IOTT>defbuff buff,512,byte  
IOTT>wburst $15,buff,512,2:eoi  
IOTT>showhpb
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 3 bytes, maximum = 512 bytes.

DAP(0): 30 21 2

Parameter Block. Current length = 0 bytes, maximum = 62 bytes.

DMA Buffer Name = BUFF DMA Transfer length = 512 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

WDATA

The **WDATA** instruction will append the write or write.e command to the DAP currently being built. The **SECONDARY** parameter is device dependent and will be used as the secondary command on the HP-IB. The **BUFFER NAME** parameter gives the name of the buffer containing the data to be sent to the device. This data will be copied into the buffer "PB" at the next available space. The "PB" buffer is the parameter block for the I/O request. The amount of data copied is determined by the **LENGTH** parameter. If the **EOI** option is given the write.e command will be appended instead of the write command. The write command causes the device to be addressed and data from the request buffer to be sent to it.

Syntax:

WDATA {secondary},{buffer name},{length} [:eoi]

Example:

```
IOTT>defbuff buff,2,byte
IOTT>altbuff buff(0),3,5
IOTT>sethpib
IOTT>wdata $5,buff,2:eoi
IOTT>showhpib : hex
```



Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 4 bytes, maximum = 512 bytes.

DAP(0): \$26 \$5 \$2 \$1

Parameter Block. Current length = 2 bytes, maximum = 62 bytes.

PB(0): \$3 \$5

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

WDMA

The **WDMA** instruction will append the **Write_dma** or **Write_dma.e** command to the DAP currently being built. The **SECONDARY** parameter is device dependent and will be used as the secondary command on the HP-IB. The data sent to the device will be obtained from the buffer specified by the **BUFFER NAME** parameter, for the length specified. The **LENGTH** parameter is assumed to be given in buffer defined units. The **Write_dma.e** command will be appended if the **EOI** option is given. **Write_dma** command causes the device to be addressed and a buffer of data from the host to be sent to the device. The data will be terminated with an **EOI**.

Syntax:

WDMA {secondary},{buffer name},{length} [:eoi]

Example:

```
IOTT>defbuff buffer,256,byte
IOTT>fillbuff buffer(0),"a"
IOTT>sethplib
IOTT>wdma $5,buffer,256:eoi
IOTT>showhplib
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 2 bytes, maximum = 512 bytes.

DAP(0): 26 5

Parameter Block. Current length = 2 bytes, maximum = 62 bytes.

DMA Buffer Name = BUFF DMA transfer length = 256 bytes.

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

WINTERF

The **WINTERF** instruction will append the **write.interface** command to the DAP currently being built. The data to be sent to the device is given by the **BUFFER NAME** parameter. The data within the buffer will be copied to the buffer "PB" at the next available space. The "PB" buffer is the parameter block for the request. The length of the copy is given by the **LENGTH** parameter, which is assumed to be given in buffer defined units. **Write.interface** command causes an interface message to be sent from the request buffer to the device. This is used to send a command which has no data message part or has interface message bytes not in the **Write** command interface message to the device.

Syntax:

WINTERF {buffer name},{length}

Example:

```
IOTT>defbuff buff,5,bytes
IOTT>altbuff buff(0),1,2,3,4,5
IOTT>sethpib
IOTT>winterf buff,5
IOTT>showhpib : hex
```

Device Adapter Program Build in Progress.

Device Adapter Program. Current length = 3 bytes, maximum = 512 bytes.

DAP(0): \$28 \$5 \$1

Parameter Block. Current length = 5 bytes, maximum = 62 bytes.

PB(0): \$1 \$2 \$3 \$4 \$5

Status Buffer. Required length = 0 bytes, maximum = 62 bytes.

HP-CIO DMA Chain Instructions

I/O Test Tool provides instructions for the user to create unique HP-CIO DMA chains, which can be used to control the protocol across the HP-CIO. The format of creating the HP-CIO chain is similar to that of the HP-IB device adapter programs. The chain building must begin with a **SETHPCIO** instruction and must end with an **ENDHPCIO** instruction. Since the user will probably wish to modify the HP-CIO chain, it is recommended that the HP-CIO chain instructions be given in program mode. The **SHOWHPCIO** instruction may be used to monitor the progress of the HP-CIO DMA chain build.

The I/O request can be issued by using the **EXECUTE** instruction with the "HPCIO" function identifier. Information specified by the Predefined I/O request instructions (**SETDATA** and others) will be ignored when "EXECUTE HPCIO" is issued. The desired device adapter must be specified by the **SETDEVICE** instruction (i.e., **SETDEVICE PDEV=1.2 [CA 1, DA 2]**). If an error is encountered, the residue count for the DMA transfers will be displayed to the user. Since unique HP-CIO DMA chains have the potential of forcing the device into an abnormal state, such program usage will be available **ONLY IN SINGLE USER MODE**.

ADDQUAD

The ADDQUAD instruction will append a quad with the information given by the user, to the current DMA chain being built. The first parameter given can be a ORDER ID which will specify the HP-CIO order to be used. The list of the ORDER ID is given below. The second parameter is the name of the buffer to be used for that I/O transfer. The LENGTH parameter specifies the length of the transfer which is assumed to be given in buffer units. The user also has the flexibility of specifying a command word in place of an ORDER ID. This will allow the user to give a full 32-bit command word. No alterations will be performed on this value, and all HP-CIO options given will be ignored.

RDORDER - Read Data Order
RSORDER - Read Status Order
WCORDER - Write Control Order
WDORDER - Write Data Order
CLCORDER - Connect Logical Channel Order
IDYORDER - Identify Order

Syntax:

```
ADDQUAD {order ID},{buffer name},{length}[:hpcio options]  
ADDQUAD {cmd value},{buffer name},{length}
```

Optional Parameters:

HPCIO options:

BYTE Indicates that a byte mode transfer is requested (bit 3 of order).
BREAK Indicates that a logical break is to be performed after this quad (bit 10 of command word).
DISCONNECT Indicates that a disconnect of the subchannel or logchannel is to be performed following the transfer (bit 7 of command word).
CONTINUE Indicates that the next quad is a continuation of the same DMA transfer (bit 30 of command word).
EXACT Indicates that if DMA terminates before count is exhausted, an interrupt will be issued to the CPU (bit 31 of command word).

Example:

```
IOTT>sethpcio  
IOTT>defbuff buff,9,byte  
IOTT>addquad idyorder,buff,9  
IOTT>endhpcio  
IOTT>showhpcio : hex
```


HPCIO DMA Chain has been Built. Program Length = 1

HPCIO Statement Number 1

Command word = \$8010000

Count Word = 9 bytes I/O Buffer Name = BUFF

Residue Count = 0 bytes

ENDHPCIO

The ENDHPCIO instruction will indicate to I/O Test Tool that the building of the HP-CIO DMA chain has completed. The user may now execute the HP-CIO DMA chain by using the EXECUTE instruction.

Syntax:

ENDHPCIO

Example:

```
IOTT>endhpcio
IOTT>defbuff idydata,9,byte
IOTT>addquad idyorder,idydata,9
IOTT>endhpcio
IOTT>showhpcio
```

HPCIO DMA Chain has been Built. Program Length = 1

HPCIO Statement Number 1

```
Command word =      134283264
Count Word    =            9 bytes    I/O Buffer Name = IDYDATA
Residue Count =            0 bytes
```

SETHPCIO

The SETHPCIO instruction will indicate to I/O Test Tool that the building of an HP-CIO DMA chain is to begin. Any DMA chains which were created previously will be destroyed. If a chain was in the process of being built when this instruction is given, a warning message will be displayed, and the previous chain will be destroyed.

Syntax:

SETHPCIO

Example:

```
IOTT>sethpcio
IOTT>showhpcio
*** NO HPCIO INSTRUCTIONS HAVE BEEN GIVEN ( IOTERR 4752 )
```

SHOWHPCIO

The SHOWHPCIO instruction will display the contents of the HP-CIO DMA chain. All information given in the ADDQUAD instruction will be displayed. The command word will display the bit pattern which will be the command word of the DMA quad. This is a translation of the ORDER ID and the HPCIO options of the ADDQUAD instruction. The count word indicates the number of bytes requested to be transferred. The I/O buffer name field indicates the name of the buffer involved in the transfer. The residue field will be filled in after the EXECUTE command has been given. The residue field displays the number of bytes not transferred. If a display mode option is given, the command word will be displayed in the given RADIX.

Syntax:

SHOWHPCIO [:display mode]

Optional Parameters:

display mode A mode in which values are to be displayed in HEX, OCT or DEC.

Example:

```
IOTT>sethpcio
IOTT>defbuff buff,9,byte
IOTT>addquad idyorder,buff,9
IOTT>endhpcio
IOTT>showhpcio : hex
```

HPCIO DMA Chain has been Built. Program Length = 1

HPCIO Statement Number 1

Command word	=	\$8010000	
Count Word	=	9 bytes	I/O Buffer Name = BUFF
Residue Count	=	0 bytes	

Program Statement Summary

Program statements are inputs which can only be used as part of a program; these include statements such as GOTO and PRINT. The following is a list and brief description of program statements available in IOTT.

Command Name	Description
COMMENT	Allows user to document the programs created.
DO-LOOP TO	Allows user to loop a group of program statements for a specified number of iterations.
GOTO	Allows user to jump unconditionally to another statement within the program.
IF-THEN/IF-THEN	Allows user to conditionally jump to another statement within the program.
PAUSE	Allows user to enter PAUSE mode during the execution of a program.
PRINT	Displays specified ASCII text to the user.
STOP	Terminates the execution of the program stored in the Program Storage Area.

Program Descriptions

The following pages provide a detailed description of each of the available I/O Test Tool program statements. The description includes information about syntax, options available for each program statement, limitations of each program statement and examples.

Program Statements

The following statements will be available in program mode only. Each of these statements must be preceded by an integer line number. Comments must be enclosed with braces ({ }) if the text is within the input. A left brace ({) alone specifies that the text following is a comment.



COMMENT

The **COMMENT** program statement will allow the user to document the programs created. This statement will be ignored during program execution.

Syntax:

```
{line number} COMMENT [ascii text]
```

Example:

```
IOTT>10 comment {this is an example of the comment statement}  
IOTT>20 comment {so is this}  
IOTT>30 comment {and this too}  
IOTT>run        {Nothing will happen}  
IOTT>
```

DO-LOOPTO

The DO-LOOPTO program statements will allow the user to loop a group of program statements for the specified number of iterations. The count parameter of the DO statement indicates the number of iterations to be performed. The line number parameter of the LOOPTO statement must match the line number of the corresponding DO statement. The program statements between the DO and the LOOPTO statements will be executed for the specified iterations.

Currently, the user can nest DO-LOOPTO constructs for a depth of 10. When entering the RUN command, each DO-LOOPTO construct will be checked to verify that the LOOPTO parameter matches the last unmatched DO statements in the program. If a conflict is encountered, an error message will be displayed to the user and the program will not be executed.

During the execution of the program, any statements which permit jumps, such as GOTO and IF-THEN, will be allowed to reference lines within a DO-LOOPTO construct. But an error message will be displayed during the LOOPTO statement execution indicating that a LOOPTO statement was encountered without executing the corresponding DO statement. The program will continue to the statement following the LOOPTO after the error message has been displayed.

Syntax:

```
{line number} DO {count}
    {group of program statements}
{line number} LOOPTO {line number}
```

Example:

```
IOTT>10 do 3      {Repeat 3 times}
IOTT>20 print "line 20"
IOTT>30 loopto 10
IOTT>40 print "end execution"
IOTT>run
line 20
line 20
line 20
end execution
IOTT>
```


GOTO

The GOTO program statement will allow the user to jump unconditionally to another statement within the program. The parameter given indicates the statement which is the destination of the jump.

When entering the RUN command, each GOTO program statement will be checked to verify that the destination line is present in the Program Storage Area. If not, an error message will be displayed to the user, and the program will not be executed.

Syntax:

```
{line number} GOTO {line number}
```

Example:

```
IOTT>10 print "line 10"  
IOTT>20 print "line 20"  
IOTT>30 goto 50           {Skip line 40}  
IOTT>40 print "line 40"  
IOTT>50 print "line 50"  
IOTT>run  
line 10  
line 20  
line 50  
IOTT>
```

IF-THEN or IFN-THEN

The IF-THEN and IFN-THEN program statements will allow the user to conditionally jump to another statement within the program. The first parameter of the statement is the expression, which will be used to determine whether the jump should be performed. For the IF statement, if the expression results in a non-zero, the jump will be performed. For the IFN statement, if the expression results in a zero value, the jump will be performed. The parameter after the reserved word "THEN" will specify the line number of the statement which is the destination of the jump. If the expression indicates that the jump is not to be performed, the statement following the IF will be executed.

The expression parameter for the IF statement can be given in either of two formats. The first format will allow the user to compare two operands, which can be a single buffer element or a literal value. The literal values can be specified in HEX(\$), OCTAL(%), decimal or ASCII(' '). ASCII can only be used with byte buffers, and in cases where two buffer elements are involved, both buffers must be of the same type. The two operands must be separated by an operator symbol.

Operators available are:

=	Equal
<>	Not equal to
AND	Logical Operator
OR	Logical Operator

For the "=" and "<>" operators, if the two operands are equal, the result will be interpreted as a non-zero value. For the "AND" and "OR" operators, the indicated logical functions will be performed, and the result of the function will determine if the jump should be performed.

The second expression format will determine whether the jump is to be performed depending on the success of the last statement executed. In this case, the "=" operand is given and the last statement executed successfully, then the jump will be performed. If the "<>" operand is given, the jump will be performed if the last statement encountered an error.

When entering the RUN command, each IF and IFN statement will be checked to verify that the destination line exists. If the line is not within the Program Storage Area, an error message will be displayed to the user, and the program will not be executed.

Syntax:

```
{line number} IF {opand}{opor}{opand} THEN {line number}
{line number} IF = THEN {line number}
{line number} IF <> THEN {line number}
```

Example:

```
IOTT>10 defbuff sample1,30,word
IOTT>20 defbuff sample2,40,word
IOTT>30 fillbuff sample1,0      {Fill buffer with zeros}
IOTT>40 altbuff sample1,0,1,2,3,4
IOTT>50 if sample1(0) = sample2(0) then 80
IOTT>60 print "buffers do not match"
IOTT>70 stop
IOTT>80 print "Sample1(0) <> $A"
IOTT>90 if sample1(0) <> $A then 110
IOTT>100 goto 60
IOTT>110 print "Sample1(0) <> $A"
IOTT>120 compbuff sample1(0),sample2(0),3 {Compare 3 values}
IOTT>130 if = then 160
IOTT>140 print "comparison was successful"
IOTT>150 stop
IOTT>160 print "comparison was not successful"
IOTT>run
Sample1(0) = Sample2(0)
Sample1(0) <> $A
SAMPLE1 ( 1 ) = 0
SAMPLE2 ( 1 ) = 1
SAMPLE1 ( 2 ) = 0
SAMPLE2 ( 2 ) = 2
comparison was not successful
IOTT>
```

PAUSE

The **PAUSE** program statement will allow the user to enter PAUSE mode during the execution of a program. In PAUSE mode, the user will be permitted to execute any instruction or command which will not alter the contents of the Program Storage Area. This will give the user a chance to suspend the execution of the program to perform some other task before resuming or aborting the program. Upon entering PAUSE mode, the line number of the PAUSE statement will be displayed to the user, and the prompt **IOTTPAUSE>** will be used until the user leaves PAUSE mode. The program can be continued by issuing a **RESUME** command, or can be terminated by entering the **ABORT** command. After the program has completed, or the **ABORT** command is entered, the prompt will return to the normal **IOTT>**.

Syntax:

{line number} PAUSE

Example:

```
IOTT>10 print "line 10"
IOTT>20 print "line 20"
IOTT>30 pause
IOTT>40 print "line 40"
IOTT>run
line 10
line 20
PAUSE Statement encountered on Line Number 30
IOTTPAUSE>list all
10 print "line 10"
20 print "line 20"
30 pause
40 print "line 40"
IOTTPAUSE>resume
line 40
IOTT>
```



PRINT

The **PRINT** program statement will display the specified ASCII text to the user. If the user specifies a null string (), a carriage return will be performed.

Syntax:

```
{line number} PRINT "[ascii text]"
```

Example:

```
IOTT>10 print "line 10"  
IOTT>20 print "line 20"  
IOTT>30 print "line 30"  
IOTT>run  
line 10  
line 20  
line 30  
IOTT>
```

STOP

The **STOP** program statement will terminate the execution of the program stored in the Program Storage Area. All statements after the **STOP** statement will not be executed. If a **STOP** program statement is not present in a program, execution will terminate after the last statement found in the Program Storage Area. The line number of the **STOP** statement will be displayed to the user when the statement is encountered.

Syntax:

{line number} STOP

Example:

```
IOTT>10 print "line 10"  
IOTT>20 print "line 20"  
IOTT>30 stop  
IOTT>40 print "line 40"  
IOTT>run  
line 10  
line 20  
STOP Statement Encountered on Line Number 30  
IOTT>
```

Predefined I/O Requests

Predefined I/O requests are categorized by device type:

- Tape drive
- Page printer
- CIPER device
- HP-IB Device Adapter
- CIO Channel Adapter
- Six-Channel Terminal Multiplexer

The device commands are listed in alphabetical order. The following pages provide quick reference summary lists (Tables 5-1./5-6.) and descriptions of each of the available I/O requests, along with information needed to issue them successfully. Each description is followed by five categories of tabular information:

MNEMONIC	Indicates the identifier which must be given in the Execute instruction to initiate a particular I/O request.
SETADDR	The word "required" found in this column indicates that a Setaddr instruction must be performed to request the I/O. If not, the reserved disk area will be used.
SETDATA	The symbol "<data1>" found in this column indicates that a Setdata instruction must be performed to request the I/O. The length of the data to be used can be chosen by the user. Special requirements may be specified in the explanation. If the symbol "<data2>" is found in this column, an auxiliary buffer and length must be given in the Setdata instruction.
SETOPTION#1	Indicates that a Setoption instruction with at least one parameter must be performed. If the parameter is to be a reserved word, the possible reserved words will be listed. If a literal value is expected, the possible values will be listed.
SETOPTION#2	Indicates that a Setoption instruction with at least two parameters must be performed. The presentation of this column is identical to that of Setoption1.

Those entries noted with dashes (-&—) are not used by that particular I/O request. If the instruction is entered, the information given will be ignored.

Table 5-1. Tape Drive Device

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
BSF	---	---	---	---
BSR	---	---	---	---
CLEAR	---	---	---	---
REQSTAT	---	---	---	---
REPORTMODE	---	---	enable/disable	---
FSF	---	---	---	---
FSR	---	---	---	---
GAP	---	---	---	---
IDENTIFY	---	<data1>	---	---
IORESULT	---	---	---	---
LOOPBACK	---	<data>	---	---
READ	---	<data>	---	---
REWIND	---	---	---	---
REWINDOFF	---	---	---	---
SELFTTEST	---	<data>	<selftest #>	---
SELSPEED	---	---	100	---
			50	---
DENSITY	---	---	800	---
			1600	---
			6250	---
WRITE	---	<data>	---	---
WFM	---	---	---	---

Table 5-2. Disk Drive

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
CANCEL	---	---	---	---
MC	---	---	---	---
DESCRIBE	---	<data>	---	---
PCHECK	---	---	parityon parityoff	srqon srqoff
IDENTIFY	---	<data>	---	---
SELFTEST	---	---	<loop #>	<selftest #>
READ	required	<data>	---	---
RFS	required	<data>	---	---
RFSI	---	<data>	---	---
RDI	---	<data>	---	---
LOCVER	required	---	<length #>	---
LOCVERI	---	---	<length #>	---
WRITE	required	<data>	---	---
WDI	---	<data>	---	---
LOOPBACK	---	<data>	<read/write>	---
NOP	---	---	---	---
RELEASE	---	---	---	---
RELDEN	---	---	---	---
REQSTAT	---	---	---	---
CLEAR	---	---	---	---
ADDRESS	required	---	---	---
SBLKDISP	---	<data>	---	---
BURST	---	---	last all	<block #>
LENGTH	---	---	<length #>	---
SETOPTS	---	---	<option #>	---
SETREL	---	---	timeon timeoff	autoon autooff
SETRETIM	---	---	<time>	---
RETAMOD	---	---	vector 1 vector 3	---
RPS	---	---	<time 1>	---
MASKSTAT	---	<data1>	---	<time 2>
VOLUME	---	---	<volume #>	---
UNIT	---	---	<unit #>	---
WFM	---	---	---	---

Table 5-3. Page Printer Devices

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
CLEAR	---	---	hard soft	---
IDENTIFY	---	<data>	---	---
LOOPBACK	---	<data>	read write	---
ENVSTAT	---	---	---	---
REQSTAT	---	---	---	---
SELFTEST	---	<data>	read write	---
WRITE	---	<data>	---	---

Table 5-4. CIPER Devices

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
CLEAR	---	<data>	---	---
IDENTIFY	---	<data>	---	---
READ	---	<data>	---	---
WRITE	---	<data>	---	---

Table 5-5. HP-IB Device Adapter

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
IDENTIFY	---	<data>	---	---
LOOPBACK	---	<data>	---	---
REQSTAT	---	---	---	---
SELFTEST	---	<data>	---	---

Table 5-6. CIO Channel Adapter

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
IDENTIFY	---	<data>	---	---
REQSTAT	---	---	---	---
SELFTEST	---	---	---	---

Table 5-7. TMUX Device Adapters

REQUEST	SETADDR	SETDATA	SETOPTION1	SETOPTION2
IDENTIFY	---	<data>	---	---
LOOPBACK	---	<data>	---	---
REQSTAT	---	---	---	---
SELFTEST	---	---	---	---

Examples

Example 1: Read and Write to a CS80 Disk.

This example will write 256 words of data from I/O Buffer SOURCE to the disk, and will then read this data back into I/O Buffer RETURN. The contents of these two buffers are then compared. If the buffers are identical, a message will be displayed to the user indicating that fact.

```
IOTT>setdevice ldev=3
IOTT>showdev
LDEV          PDEV          PRODUCT
  3           5.5.0         hp 7935

Current Device is the following:
  3           5.5.0         hp 7935
IOTT>defbuff status,20,byte:status {Define user status buffer}
IOTT>defbuff source,256,word {Define I/O buffers needed}
IOTT>defbuff return,256,word
IOTT>fincbuff source(0),0,255 {Pattern is data from 0..255}
IOTT>30 setdata source,256
IOTT>40 setaddr 2000 {Use logical sector 2000}
IOTT>45 print "executing write"
IOTT>50 execute write {Write data to disk}
IOTT>60 setdata return,256
IOTT>65 print "executing read"
IOTT>70 execute read {Read data to RETURN buffer}
IOTT>75 print "start comparison"
IOTT>80 compbuff source(0),return(0),256,10 {Compare buffers}
IOTT>90 if <> then 110 {Are the buffers identical?}
IOTT>100 print "comparison successful"
IOTT>110 stop
IOTT run
executing write
executing read
start comparison
comparison successful
STOP Statement Encountered on Line Number 110
IOTT>
```

Example 2: Limited Disk Pattern Search

This example will search the disk for a particular data pattern. In this example, the data pattern to be found is stored in buffer SEARCH. The buffer RETURN will hold the data from the disk. To make sure that the pattern can be found in this example, the search pattern is written to address 2000 (single vectored), and the search process will start at address 1998 (single vectored).

```
IOTT>setdevice ldev=3
IOTT>20 defbuff return,256,byte
IOTT>30 defbuff hardstat,20,byte:status
IOTT>31 defbuff search,20,byte
IOTT>32 altbuff search(0),"search string"
IOTT>33 setaddr 2000
IOTT>34 setdata search,13
IOTT>35 execute write
IOTT>40 setaddr 1998
IOTT>50 eincaddr 1
IOTT>60 setdata return,256
IOTT>70 execute read
IOTT>80   if = then 105
IOTT>90   dstatus
IOTT>100  pause
IOTT>105 do 250
IOTT>111  compbuff search(0),return(0),10,1,diff
IOTT>120  if = then 140
IOTT>130  pause
IOTT>140  copybuff return(1),return(0),255
IOTT>143 loopto 105
IOTT>145 goto 60
IOTT>run
SEARCH ( 0 ) = s
RETURN ( 0 ) = s
***ERROR DETECTED AT LINE NUMBER 111 ( IOTTERR 5999 )
IOTTPAUSE>abort
```



Example 3: Forcing a Protocol Error on a CS80 Disk

This example will force a protocol error (receive a message of the wrong length) to be returned from a CS80 disk. This example also demonstrates how the write data immediate I/O request can be used. The write data immediate I/O request will not send any CS80 complimentary commands with the write request. This means that the user must perform a SET LENGTH and a SET ADDRESS command before issuing the write. The protocol error is generated by issuing a SET LENGTH command specifying a data transfer length of 20 bytes, and then issuing the write request with a data transfer of 1024 bytes.

```
IOTT>setdevice ldev=3
IOTT>showdev
      LDEV          PDEV          PRODUCT
      3            5.5.0          hp 7935

Current Device is the following:
      3            5.5.0          hp 7935
IOTT>defbuff status,20,byte:status
IOTT>defbuff source,256,word
IOTT>setdata source,256          {Data to be used for write request}
IOTT>setaddr 2000                {Write data to logical sector 2000}
IOTT>execute address             {Execute Set Address command}
IOTT>setoption 20                {Data transfer will be 20 bytes}
IOTT>execute length              {Execute Set Length command}
IOTT>execute wdi                 {Write data immediate of 256 words}
TRANSFER COUNT = 1024 BYTES : HARDWARE STATUS COUNT = 20 BYTES
***I/O FAILED DUE TO DEVICE ERROR - CHECK HARDWARE STATUS
(DSSERR 501)
IOTT>dstatus                     {Decode hardware status returned}

Status =

Unit = 0  Volume = 0

No units with status pending

>>>>>>>> REJECT ERRORS <<<<<<<<<
Received a message of the wrong length

New target address is:
Block address = 2001
IOTT>
```

Example 4: Locate and Read Full Sector for a CS80 Disk

This example will perform a locate and read full sector at logical sector 2000. The data will be shifted six (6) bytes to the left, and then written back to sector 2000. This process is used very often by field people to recover data. Remember, read full sector will return a header of 6 bytes, followed by 256 bytes of data, and then a trailer of 7 bytes.

```
IOTT>setdevice ldev=3
IOTT>defbuff buff,269,byte
IOTT>defbuff status,20,byte:status
IOTT>setaddr 2000 {Use logical sector 2000}
IOTT>setdata buff,269 {Specify read buffer}
IOTT>execute rfs {Read full sector immediate}
IOTT>db buff {Display data read}
0-----1-----2-----3-----4-----5-----6-----
me="ijkl";mf="abcd";mg="stuv";mh="opqr";mj="ghij";mk="yzab"
BUFF( 0):.... .abcdefghijklmnopqrstuvwxyzabcdefgh
BUFF( 70):mnopqrstuvwxyzabcdefghijklmnopqrstuvwxy
BUFF(140):efghijklmnopqrstuvwxyzabcdefghijklmnopqr
BUFF(210):wxyzabcdefghijklmnopqrstvwxyzabcdefghijklmnopqrstuvQgxga).

IOTT>copy buff(6),buff(0),256 {Shift data left by 6 bytes}
IOTT>db buff {Confirm data movement}
0-----1-----2-----3-----4-----5-----6-----
BUFF( 0):abcdefghijklmnopqrstvwxyzabcdefghijklmnopqrstvwxyzabcdefghijklmnop
BUFF( 70):stuvwxyzabcdefghijklmnopqrstvwxyzabcdefghijklmnopqrstvwxyzabcde
BUFF(140):klmnopqrstuvwxyzabcdefghijklmnopqrstvwxyzabcdefghijklmnopqrstuvw
BUFF(210):cdefghijklmnopqrstvwxyzabcdefghijklmnopqrstvwqrstuvQgxga).

IOTT>setdata buff,256 {Change transfer count to 256}
IOTT>execute write {Write data to disk}
IOTT>
```

Example 5: Unblocked Write to a Streaming Tape

This example will write 16 blocks of data, and will verify the data by reading the blocks. The sequence for the reads will be read one block, rewind, read 2 blocks, rewind, until all 16 blocks are read. If any READ requests fail, the program will stop.

```
IOTT>10 setdevice ldev=7 {Device must be a tape}
IOTT>20 defbuff status,6,byte:status
IOTT>30 defbuff pattern,256,word
IOTT>40 fincbuff pattern(0),0,256
IOTT>50 setdata pattern,256
IOTT>60 execute write,16:unblock {Write data in streaming mode}
IOTT>65 execute ioreult
IOTT>66 if <> then 65
IOTT>70 defbuff index,1,word
IOTT>80 defbuff counter,1,word
IOTT>90 altbuff index(0),0
IOTT>100 defbuff return,256,word
IOTT>110 setdata return,256
IOTT>120 altbuff counter(0),0
IOTT>130 adjbuff index (0),1
IOTT>140 execute rewind
IOTT>150 adjbuff counter(0),1
IOTT>160 execute read
IOTT>170 if = then 210
IOTT>180 print "read failed"
IOTT>190 dstatus
IOTT>200 stop
IOTT>210 if index(0) <> counter(0) then 150
IOTT>220 compbuff pattern(0),return(0),256,10
IOTT>230 if = then 120
IOTT>240 pause
IOTT>250 if index(0) <> 16 then 120
IOTT>260 execute rewind
IOTT>270 stop
IOTT>execute rewind
IOTT>run
TRANSFER COUNT = 0 BYTES : HARDWARE STATUS COUNT = 0 BYTES
***I/O HAS NOT YET COMPLETED (DSSERR 510)
***ERROR DETECTED AT LINE NUMBER 65 ( IOTTERR 5999 )

STOP Statement Encountered on Line Number 270
IOTT>
```


Example 6: Write Lines of Text to EPOC

This example will write 80 lines of text to the EPOC printer. Before the text can be sent to the device, a job open record must be sent to the device. After this record is sent, the text itself is sent. The job is terminated by a clear environment record, and a job close. All records sent to the printer must be 1024 bytes long.

```
IOTT>10 setdevice ldev=14           {Device must be PP}
IOTT> 20 defbuff buff,1024,byte
IOTT> 30 defbuff status,32,byte:status {Store device status}
IOTT> 40 comment -- build job open record
IOTT> 50 altbuff buff(0),0,8         {Length of record}
IOTT> 60 altbuff buff(2),0,0         {No data transfer}
IOTT> 70 altbuff buff(4),0,142       {Function code: job open}
IOTT>80 altbuff buff(6),0,0,0,0     {P1 and P2 not used}
IOTT>90 altbuff buff(10),255,255    {Mark end of data}
IOTT>100 setdata buff,1024         {Issue job open}
IOTT>110 execute write
IOTT>120 comment -- build data record
IOTT>130 altbuff buff(0),0,88        {Length of record}
IOTT>140 altbuff buff(2),0,80        {Transfer 80 bytes}
IOTT>150 altbuff buff(4),0,1         {Function code: write}
IOTT>160 altbuff buff(6),0,32        {P1 - single space}
IOTT>170 altbuff buff(8),0,0         {P2 - not used}
IOTT>180 fillbuff buff(10),"",80    {Fill pattern}
IOTT>190 altbuff buff(40),"I/O Test Tool"
IOTT>200 altbuff buff(90),255,255    {Mark end of data}
IOTT>210 setdata buff,1024
IOTT>220 execute write,80           {Write 80 lines of data}
IOTT>230 comment -- format print environment
IOTT>240 altbuff buff(0),0,8         {Length of record}
IOTT>250 altbuff buff(2),0,0         {No data transfer}
IOTT>260 altbuff buff(4),0,141       {Clear environment}
IOTT>270 altbuff buff(6),0,0,0,0     {P1 and P2 not used}
IOTT>280 altbuff buff(10),255,255    {Mark end of data}
IOTT>290 setdata buff,1024
IOTT>300 execute write
IOTT>310 setoption soft              {Job close}
IOTT>320 execute clear
IOTT>330 execute envstat
```

Example 7: Ripple Print of Test to a CIPER Printer

The example will initiate 80 lines of ripple print on a CIPER line printer. This is performed by sending an ESC z pattern to the printer. Before this is done, the user must synchronize with the printer by performing a device clear, and receiving appropriate receive ready messages from the printer.

```
IOTT>5  comment - Format clear record
IOTT>10 debuff clear,9,byte           {Will hold clear device request}
IOTT>20 altbuff clear(0),4,1,0,0      {Packet header}
IOTT>30 altbuff clear(4),4,1,0,0      {Record header}
IOTT>40 altbuff clear(8),1            {Data: purge internal buffer}
IOTT>45 comment -- Format data record
IOTT>50 debuff data,1024,byte
IOTT>60 altbuff data(0),4,1,0,0       {Packet header}
IOTT>70 altbuff data(4),4,0,16,96     {Record header}
IOTT>80 altbuff data(8),6,0,0,0,0,0   {Block header:necessary for write}
IOTT>90 altbuff data(14),27,38,112,49,51,50,88{Enable display functions}
IOTT>100 fincbuff data(21),0,127,132  {Fill 132 bytes of data pattern}
IOTT>110 altbuff data(154),10,13      {Append CR and LF}
IOTT>115 comment -- define buffer for reply from printer
IOTT>120 debuff return,32,byte        {Will hold replies from the printer}
IOTT>130 fillbuff return(0),0         {zero out buffer}
IOTT>132 comment -- issue clear request
IOTT>135 debuff counter,1,byte
IOTT>150 setdata clear,9
IOTT>152 execute write                {Issue device clear}
IOTT>151 comment -- obtain clear response record from printer
IOTT>154 setdata return,32
IOTT>156 execute read                 {Obtain reply}
IOTT>158 if then 500                 {If I/O was not successful, stop}
```

```

IOTT>160 if return(6) 1 then 156      {Look for clear response}
IOTT>162 if return(8) 100 then 156    {Clear response match request}
IOTT>164 print "clear response obtained"
IOTT>166 fillbuff return(0),0
IOTT>167 altbuff counter(0),0
IOTT>170 do 80                        {Print 80 lines}
IOTT>171   if return(8) 0 then 182    {Get next receive ready response}
IOTT>172     setdata return,32
IOTT>174     execute read              {Obtain receive ready response}
IOTT>176     if then 500
IOTT>178     if return(6) 0 then 172  {Loop for receive ready}
IOTT>180     print "receive ready obtained"
IOTT>182     setdata data,156
IOTT>184     execute write             {Send data to printer}
IOTT>186     adjbuff return(8),-1
IOTT>190     adjbuff data(5),1         {Update record number}
IOTT>200     adjbuff data(13),1       {Update block number}
IOTT>201     copybuff data(22),data(21),132 {Shift data one char over}
IOTT>202     copybuff counter(0),data(253),1
IOTT>203     adjbuff counter(0),-1
IOTT>204 loopto 170
IOTT>205 stop
IOTT>500 print "I/O failed"
IOTT>501 stop

```

Example 8: Initiating Selftest on a CIPER Printer

The example will initiate power-on selftest on a CIPER line printer. This is performed by sending an ESC z pattern to the printer. Before this is done, the user must synchronize with the printer by performing a device clear, and receiving appropriate receive ready messages from the printer.

```
IOTT>5  comment - Format clear record
IOTT>10 defbuff clear,9,byte           {Will hold clear device request}
IOTT>20 altbuff clear(0),4,1,0,0       {Packet header}
IOTT>30 altbuff clear(4),4,100,1,96    {Record header}
IOTT>40 altbuff clear(8),1             {Data: purge internal buffer}
IOTT>45 comment -- Format data record
IOTT>50 defbuff data,1024,byte
IOTT>60 altbuff data(0),4,1,0,0        {Packet header}
IOTT>70 altbuff data(4),4,0,16,96      {Record header}
IOTT>80 altbuff data(8),6,0,0,0,0,0    {Block header: necessary for write}
IOTT>90 altbuff data(14),27,122        {Characters (esc) z}
IOTT>115 comment -- define buffer for reply from printer
IOTT>120 defbuff return,32,byte        {Will hold replies from the printer}
IOTT>130 fillbuff return(0),0          {Zero out buffer}
IOTT>132 comment -- issue clear request
IOTT>150 setdata clear,9
IOTT>152 execute write                  {Issue device clear}
IOTT>151 comment -- obtain clear response record from printer
IOTT>154 setdata return,32
IOTT>156 execute read                  {Obtain reply}
```

```

IOTT>158 if <> then 500           {If I/O was not successful, stop}
IOTT>160 if return(6) <> 1 then 156 {Look for clear response}
IOTT>162 if return(8) <> 100 then 156 {Clear response match request}
IOTT>164 print "clear response obtained"
IOTT>166 fillbuff return(0),0
IOTT>171 if return(8) <> 0 then 182 {Get next receive ready response}
IOTT>172   setdata return,32
IOTT>174   execute read           {Obtain receive ready response}
IOTT>176   if <> then 500
IOTT>178   if return(6) <> 0 then 172 {Loop for receive ready}
IOTT>180   print "receive ready obtained"
IOTT>182 setdata data,16         {Send selftest record}
IOTT>184 execute write           {Send data to printer}
IOTT>185 settimer 25             {Reduce timeout from 60 to 25 seconds}
IOTT>186 setdata return,32
IOTT>187 execute read
IOTT>188 if <> then 200           {Timeout error}
IOTT>189   print "device request timeout: assume selftest successful"
IOTT>190   stop
IOTT>200 if return(6) <> 2 then 186 {Find device status}
IOTT>201   copybuff data(8),data(0),6 {Shift device status over headers}
IOTT>202   dstatus                   {Decode device status}
IOTT>205 stop
IOTT>500 print "I/O failed"
IOTT>501 stop

```

Error and Warning Messages

This section gives a complete list of the error messages that may be generated by I/O Test Tool, listed in numerical order. Other diagnostic messages may appear at any time. Messages that appear without the (IOTERR #) trailer are generated by the Online Diagnostic Subsystem or the operating system. Listed below each message are a probable cause and a recommended action statement. The actual cause and action may differ from this list, depending upon the particular situation. The "!" indicates that a parameter of some sort will replace the exclamation point when the message is displayed.

1000	BUFFER ! HAS NOT BEEN ALLOCATED (IOTERR 1000)	
CAUSE	The user issued an instruction requiring an I/O buffer, but the user has not performed a DEFBUFF instruction to allocate the I/O buffer.	
ACTION	Please consult the I/O Test Tool help facility for further information on the DEFBUFF instruction.	
1001	INDEX VALUE OF ! WILL ACCESS BEYOND THE BOUNDARY OF BUFFER ! (IOTERR 1001)	
CAUSE	The user has issued an instruction where the buffer index value given is larger than the buffer size.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued. The SHOWBUFF instruction will display the size and type of all buffers defined. All buffer index values start with value zero.	
1002	PERFORMING FUNCTION WILL EXCEED THE MAXIMUM LENGTH OF THE DATA BUFFER (IOTERR 1002)	
CAUSE	The user issued an instruction where the buffer length parameter and buffer index given will force the instruction to access beyond the limit of the buffer.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued. The SHOWBUFF instruction will display the size and type of all buffers defined. All buffer index values start with value zero.	
1003	INTERNAL ERROR: TABLE ACCESS PROCEDURE !. (IOTERR 1003)	
CAUSE	I/O Test Tool encountered an error that it could not recover from. This is a software error.	
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.	
1004	UNABLE TO OBTAIN DATA OBJECT FOR INTERNAL WORK AREA I/O TEST TOOL CANNOT CONTINUE EXECUTING (IOTERR 1004)	
CAUSE	I/O Test Tool was not able to obtain data space from the operating system to store internal information. I/O Test Tool cannot continue execution. MPE XL Operating System error information will be displayed with this message.	
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.	

2000	WARNING: SECTION NUMBERS ARE NOT USED BY I/O TEST TOOL
CAUSE	The user gave the SECTION parameter when initiating I/O Test Tool through the Diagnostic User Interface. I/O Test Tool does not use this parameter.
ACTION	I/O Test Tool will ignore the SECTION parameter and will continue execution.
2001	WARNING: STEP NUMBERS ARE NOT USED BY I/O TEST TOOL
CAUSE	The user gave the STEP parameter when initiating I/O Test Tool through the Diagnostic User Interface. I/O Test Tool does not use this parameter.
ACTION	I/O Test Tool will ignore the STEP parameter and will continue execution.
2002	WARNING: LOOP COUNT IS NOT USED BY I/O TEST TOOL
CAUSE	The user gave the LOOPCOUNT parameter when initiating I/O Test Tool through the Diagnostic User Interface. I/O Test Tool does not use this parameter.
ACTION	I/O Test Tool will ignore the LOOPCOUNT parameter and will continue execution.
2003	WARNING:ERRONLY PARAMETER IS NOT USED BY I/O TEST TOOL
CAUSE	The user gave the ERRONLY parameter when initiating I/O Test Tool through the Diagnostic User Interface. I/O Test Tool does not use this parameter.
ACTION	I/O Test Tool will ignore the ERRONLY parameter and will continue execution.
2004	WARNING: ERRCOUNT PARAMETER IS NOT USED BY I/O TEST TOOL
CAUSE	The user gave the ERRCOUNT parameter when initiating I/O Test Tool through the Diagnostic User Interface. I/O Test Tool does not use this parameter.
ACTION	I/O Test Tool will ignore the ERRCOUNT parameter and will continue execution.
2007	WARNING: CONTROL-Y MAY NOT WORK: INITIALIZATION ERROR ENCOUNTERED
CAUSE	I/O Test Tool could not enable control-Y during initialization. I/O Test Tool will continue execution, but control-Y will be ignored.
ACTION	User can still use I/O Test Tool, but please report all error messages displayed to support personnel.
2008	I/O TEST TOOL INITIALIZATION CANNOT BE COMPLETED ERROR ENCOUNTERED WHILE PROCESSING MESSAGE NUMBER ! MESSAGE CATALOG CONTAINS INCORRECT COMMAND FORMAT (IOTERR 2008)
CAUSE	I/O Test Tool was not able to initialize because the command strings read from the diagnostic message catalog were not in the correct format.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated.



2009	I/O TEST TOOL INITIALIZATION CANNOT BE COMPLETED INTERNAL DATA BUFFER COULD NOT BE ALLOCATED (IOTERR 2009)	
CAUSE	I/O Test Tool could not complete initialization because it was not able to obtain data space for internal data buffers.	
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.	
3000	PROGRAM STORAGE AREA IS EMPTY (IOTERR 3000)	
CAUSE	User tried to perform a command which required a program to exist in the Program Storage Area, but the area was empty.	
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.	
3001	STARTING LINE COULD NOT BE FOUND (IOTERR 3001)	
CAUSE	User has issued a command to access the Program Storage Area, but the starting line number given does not exist.	
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.	
3002	DESTINATION LINE NUMBER ALREADY EXISTS (IOTERR 3002)	
CAUSE	User has issued a command which will access the Program Storage Area, but the destination line is already occupied.	
ACTION	Please delete the destination line and repeat the command.	
3003	TOO MANY LINES TO MOVE, PLEASE RENUMBER PROGRAM (IOTERR 3003)	
CAUSE	The destination area is too small to move the requested number of lines.	
ACTION	Renumber the program using an increment factor large enough to hold the number of lines to be moved. Then issue the MOVE command again.	
3004	EXCEEDED MAXIMUM LINE NUMBER, RENUMBERING NOT PERFORMED (IOTERR 3004)	
CAUSE	Performing the RENUMBER command would have exceeded the maximum line number of 9999.	
ACTION	Please select an smaller increment factor and repeat the RENUMBER command.	
3005	I/O TEST TOOL IS NOT IN PAUSE MODE (IOTERR 3005)	
CAUSE	User tried to perform an instruction in normal I/O Test Tool mode which requires the user to be in PAUSE mode.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction given.	

3006	PROGRAM STORAGE AREA MUST BE EMPTY BEFORE LOAD CAN BE PERFORMED (IOTERR 3006)
CAUSE	The LOAD command can only be performed when the Program Storage Area is empty. This is a precautionary requirement to guarantee that the user does not accidentally overwrite the current program.
ACTION	Perform a DELETE ALL to empty the Program Storage Area, and issue the LOAD command again.
3007	DO-LOOP TO NESTING LIMIT (10) HAS BEEN EXCEEDED (IOTERR 3007)
CAUSE	User has created a program which contains many nested DO-LOOP TO statements. The user has exceeded the nesting limit for the DO-LOOP TO construct.
ACTION	If deeper nesting is required, please report enhancement request to support personnel.
3008	LOOP TO STATEMENT WAS ENCOUNTERED BUT NO DO STATEMENT WAS FOUND PREVIOUSLY (IOTERR 3008)
CAUSE	While I/O Test Tool was performing program verification, a LOOP TO statement was encountered before a DO statement. The program will not be executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the LOOP TO statement.
3009	LOOP TO STATEMENT DOES NOT MATCH THE LAST DO STATEMENT ENCOUNTERED (IOTERR 3009)
CAUSE	While I/O Test Tool was performing program verification, a LOOP TO statement parameter indicating a line number which did not match the line number of the last DO statement was found. The program will not be executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the LOOP TO statement.
3010	THE GOTO DESTINATION LINE COULD NOT BE FOUND (IOTERR 3010)
CAUSE	While I/O Test Tool was performing program verification, a GOTO statement with a non-existent line number was encountered. Program will not be executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the GOTO statement.
3011	THE IF DESTINATION LINE COULD NOT BE FOUND (IOTERR 3011)
CAUSE	While I/O Test Tool was performing program verification, an IF or IFN statement with a non-existent destination line number was found. Program will not be executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the IF or IFN statement.

3013	THE JUMP DESTINATION LINE COULD NOT BE FOUND (IOTERR 3013)
CAUSE	While I/O Test Tool was performing program verification, a JUMP instruction with a non-existent destination line number was found. Program will not be executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the JUMP instruction.
3014	THE DESTINATION LINE IS NOT AN HP-IB INSTRUCTION (IOTERR 3014)
CAUSE	While I/O Test Tool was performing program verification, an HP-IB instruction which had a destination line number to an instruction which was not HP-IB was found. Program was not executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
3015	DESTINATION LINE NUMBER ! COULD NOT BE FOUND (IOTERR 3015)
CAUSE	While I/O Test Tool was performing program verification, a HP-IB instruction with a non-existent destination line number was found. Program was not executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
3016	INTERNAL ERROR OCCURRED: COMMAND MODULE ! (IOTERR 3016)
CAUSE	I/O Test Tool encountered an error it could not recover from. This is a software error.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.
3017	WARNING: LINE WAS TRUNCATED, PLEASE RE-ENTER THIS LINE (IOTERR 3017)
CAUSE	The RENUMBER or MOVE command caused the input line to be extended beyond the limit of the input line text. The line number at the beginning of the line has been extended by one or more characters.
ACTION	Please shorten the comments on the input line. I/O Test Tool limits user input to 80 characters.
3018	LINE REFERENCED COULD NOT BE FOUND, LINE SET TO 9999 (IOTERR 3018)
CAUSE	During the MOVE or RENUMBER command, I/O Test Tool encountered line references to lines which do not exist. These line references have been changed to reference line 9999.
ACTION	Please modify all lines referencing line 9999 to the correct line numbers.
3021	PROGRAM EXECUTION WAS NOT INITIATED (IOTERR 3021)
CAUSE	Execution of the program in the program storage area was not initiated due to errors.
ACTION	Error messages indicating why the program was not initiated will be displayed with this error message. Please consult the ACTION clause for those error messages for further information.

3022	BACKWARD REFERENCE TO LINE ! IS ILLEGAL FOR THIS HP-IB INSTRUCTION (IOTERR 3022)
CAUSE	While I/O Test Tool was performing program verification, an HP-IB instruction with backward references to other HP-IB instruction, was found. Backward references are not allowed in HP-IB device adapter programs. Program was not executed.
ACTION	Please consult the I/O Test Tool help facility for further information on the HP-IB instruction issued.
3023	ERROR OCCURRED DURING SUSPEND, BUT EXECUTION WILL RESUME (IOTERR 3023)
CAUSE	I/O Test Tool encountered an error while processing a SUSPEND command.
ACTION	I/O Test Tool will continue execution. Please report all error messages displayed to support personnel.
3024	NO LOOP TO FOUND FOR DO STATEMENT (IOTERR 3024)
CAUSE	While performing program verification, I/O Test Tool was not able to locate the LOOP TO statement which corresponds to the DO statement displayed.
ACTION	Please consult the I/O Test Tool help facility for further information on the DO statement.
3025	MODIFIED INPUT TOO LONG, PLEASE TRY AGAIN (IOTERR 3025)
CAUSE	The resulting line from the MODIFY command will exceed the line limit in I/O Test Tool.
ACTION	Please modify the line, but the limit for I/O Test Tool is 80 characters.
3026	CATALOG CANNOT BE ACCESSED, LOAD CANNOT BE PERFORMED (IOTERR 3026)
CAUSE	The I/O Test Tool file header text could not be obtained from the diagnostic message catalog. I/O Test Tool will not be able to load the file.
ACTION	I/O Test Tool will continue execution, but please report all error messages displayed to support personnel.
3027	FILE IS NOT AN I/O TEST TOOL LOAD FILE (IOTERR 3027)
CAUSE	The file given in the load command was not created by I/O Test Tool.
ACTION	Please consult the I/O Test Tool help facility for further information on the LOAD command.
3028	DESTINATION LINE NUMBER IS WITHIN THE RANGE OF THE LINES TO BE MOVED (IOTERR 3028)
CAUSE	The user tried to move lines where the source lines and destination lines are the within the same range.
ACTION	Please verify that the line numbers specified in the MOVE command are correct.

3029	REDO LINE WAS TRUNCATED (IOTERR 3029)
CAUSE	The resulting line from the REDO command exceeds 80 characters. The line was truncated to 80 characters.
ACTION	Please issue the line with shorter comments.
3030	UNABLE TO READ IOTT HEADER (IOTERR 3030)
CAUSE	I/O Test Tool was not able to read the I/O Test Tool header from the load file specified by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3031	UNABLE TO CLOSE LOAD FILE (IOTERR 3031)
CAUSE	I/O Test Tool was not able to close the load file specified by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3032	UNABLE TO OPEN LOAD FILE (IOTERR 3032)
CAUSE	I/O Test Tool was not able to open the load file specified by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3033	UNABLE TO COMPLETE FILE NAME (IOTERR 3033)
CAUSE	I/O Test Tool was not able to fully qualify the load file name given by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3034	UNABLE TO OPEN OLD SAVE FILE (IOTERR 3034)
CAUSE	I/O Test Tool was not able to open the old copy of the save file given by the user. I/O Test Tool must verify that the old file was created by I/O Test Tool before purging that file, and replacing it with a new file.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3035	UNABLE TO CLOSE OLD SAVE FILE (IOTERR 3035)
CAUSE	I/O Test Tool was not able to close the old copy of the save file given by the user. I/O Test Tool must verify that the old file was created by I/O Test Tool before purging the file, and replacing it was a new file.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3036	UNABLE TO WRITE IOTT HEADER TO SAVE FILE (IOTERR 3036)
CAUSE	I/O Test Tool was not able to write the I/O Test Tool file header to the save file.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.

3037	UNABLE TO OPEN NEW SAVE FILE (IOTERR 3037)
CAUSE	I/O Test Tool was not able to open the new save file created for the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3038	UNABLE TO CLOSE NEW SAVE FILE (IOTERR 3038)
CAUSE	I/O Test Tool was not able to close the save file specified by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
3039	UNABLE TO WRITE TO SAVE FILE (IOTERR 3039)
CAUSE	I/O Test Tool was not able to write the contents of the program storage area to the save file specified by the user.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
4000	DEFBUFF INSTRUCTION UNABLE TO ALLOCATE DATA BUFFER (IOTERR 4000)
CAUSE	I/O Test Tool was not able to allocate a data buffer. Therefore, the DEFBUFF instruction was not performed.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
4001	RELBUFF INSTRUCTION UNABLE TO DEALLOCATE DATA BUFFER (IOTERR 4001)
CAUSE	I/O Test Tool was not able to deallocate a data buffer. Therefore, the RELBUFF instruction was not performed.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.
4003	THE START PARAMETER IS NOT COMPATIBLE WITH A BYTE BUFFER (IOTERR 4003)
CAUSE	The user has issued a buffer instruction involving a byte buffer, but the starting value is beyond the limits of a byte (0 - 255).
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
4004	THE END PARAMETER IS NOT COMPATIBLE WITH A BYTE BUFFER (IOTERR 4004)
CAUSE	The user has issued a buffer instruction involving a byte buffer, but the ending value is beyond the limits of a byte (0 - 255).
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

4005	INC PARAMETER INCOMPATIBLE WITH START AND STOP VALUES GIVEN (IOTERR 4005)
CAUSE	The user has issued a buffer instruction where the increment parameter plus the starting buffer index will be greater than the stop index value.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
4007	BUFFERS MUST BE OF THE SAME TYPE (IOTERR 4007)
CAUSE	User has issued an instruction involving two data buffers, but the type of buffers given were not identical.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
4008	PARAMETER NUMBER ! IS NOT A VALID DATA VALUE FOR A BYTE BUFFER (IOTERR 4008)
CAUSE	The nth data value given is outside the limits of a byte buffer (0 - 255).
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
4011	DATA VALUE IS NOT COMPATIBLE WITH A BYTE BUFFER (IOTERR 4011)
CAUSE	The data value given is either too large or too small to be stored into the buffer of type byte.
ACTION	Please limit the data value to between 0 and 255.
4012	VALUE EXCEEDED SPAN OF THE BYTE BUFFER (IOTERR 4012)
CAUSE	The value given is outside the limits of a byte buffer.
ACTION	Please limit data values to between 0 and 255.
4250	SETADDR WAS NOT PERFORMED (IOTERR 4250)
CAUSE	The user requested an I/O function which requires the information supplied by the SETADDR instruction.
ACTION	Please consult the I/O Test Tool help facility for a list of items needed to perform the requested I/O.
4251	ADDRESS INCREMENTATION EXCEEDS MAXIMUM FOR DEVICE (IOTERR 4251)
CAUSE	The address given through the SETADDR instruction was larger than the maximum address for the device, or the address incrementation function has reached the maximum for the device.
ACTION	Please consult the SHOWPARM instruction for maximum address for the device.

4252	UNEXPECTED INTERNAL ERROR IN PREDEFINED I/O MODULE PROCEDURE ! (IOTERR 4252)
CAUSE	I/O Test Tool encountered an error that it could not recover from. I/O Test Tool cannot continue execution.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.
4254	SETDEVICE HAS NOT BEEN PERFORMED, INSTRUCTION NOT PERFORMED (IOTERR 4254)
CAUSE	User tried to perform an instruction which requires that a current device be defined.
ACTION	Please perform a SETDEVICE instruction to define a current device.
4255	CURRENT DEVICE DOES NOT USE ADDRESSES (IOTERR 4255)
CAUSE	User has tried to perform an INCADDR instruction for a device which does not use addresses.
ACTION	Please consult the I/O Test Tool help facility for further information on the INCADDR instruction.
4256	DEVICE HAS NO I/O REQUESTS PENDING (IOTERR 4256)
CAUSE	User requested abortio for a device which cannot perform unblocked I/O requests or has no pending unblock I/O requests.
ACTION	Please consult the I/O Test Tool help facility for further information on I/O requests.
4257	"PROTECT ON" ALREADY PERFORMED (IOTERR 4257)
CAUSE	User has tried to execute the PROTECT ON instruction twice.
ACTION	Please consult the I/O Test Tool help facility for further information on the PROTECT command.
4258	"PROTECT OFF" ALREADY PERFORMED (IOTERR 4258)
CAUSE	User has tried to execute the PROTECT OFF instruction twice.
ACTION	Please consult the I/O Test Tool help facility for further information on the PROTECT command.
4500	WARNING: Previous HP-IB DAP has been deleted.
CAUSE	The user has issued a SETHPIB instruction after creating an HP-IB device adapter program. At this point, the old HP-IB device adapter program will be deleted.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETHPIB instruction.
4501	PREVIOUS HP-CIO PROGRAM HAS NOT BEEN COMPLETED (IOTERR 4501)
CAUSE	The user has issued an EXECUTE HPCIO before completing the HPCIO DMA CHAIN.
ACTION	Please consult the I/O Test Tool help facility for further information on the ENDHPCIO instruction.

4502	DATA AREA FOR COMMAND BLOCK COULD NOT BE ALLOCATED (IOTTERR 4502)
CAUSE	I/O Test Tool was not able to obtain data space to use in building a HP-IB device adapter program.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause for those error messages for further information.
4503	SETHPIB INSTRUCTION HAS NOT BEEN ISSUED (IOTTERR 4503)
CAUSE	User has issued HP-IB device adapter program instructions before performing a SETHPIB instruction.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETHPIB instruction.
4504	HP-IB DEVICE ADAPTER PROGRAM IS TOO LARGE TO APPEND INSTRUCTION GIVEN (IOTTERR 4504)
CAUSE	User is trying to build a HP-IB device adapter program which will be larger than the limits set by hardware.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of an HP-IB DAP program.
4506	PARAMETER BLOCK LIMIT HAS BEEN EXCEEDED (IOTTERR 4506)
CAUSE	User has tried to create an HP-IB device adapter program which will result in a parameter block exceeding the limit set by hardware.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.
4507	UNEXPECTED INTERNAL ERROR: !
CAUSE	I/O Test Tool encountered an error it could not recover from. This is a software error.
ACTION	Please report all error message displayed to support personnel.
4508	PREVIOUS DMA BUFFER WILL BE IGNORED (IOTTWARN 4508)
CAUSE	The user has issued an HP-IB instruction which requires a DMA buffer. Therefore, the DMA buffer specified previously will be ignored.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of an HP-IB device adapter program.
4509	LINE NUMBER ! WAS REFERENCED BUT NOT EXECUTED (IOTTERR 4509)
CAUSE	I/O Test Tool has encountered an HP-IB instruction which references another HP-IB instruction, but that instruction has not been executed.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.

4510	DSJ JUMP LOCATIONS EXCEED LIMIT (IOTERR 4510)
CAUSE	The user has exceed the limit of DSJ values allowed.
ACTION	Please consult the I/O Test Tool help facility for further information on the DSJ instruction.
4512	HP-IB DEVICE ADAPTER PROGRAM HAS NOT BEEN BUILT (IOTERR 4512)
CAUSE	The user has tried to issue an EXECUTE HPIB instruction without building an HP-IB device adapter program.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.
4513	WARNING: Status Buffer has not been defined.
CAUSE	The user has issued the SHOWHPIB instruction, but has not designated a status buffer.
ACTION	Please consult the I/O Test Tool help facility for further information on the DEFBUFF instruction.
4514	PARAMETER BLOCK COULD NOT BE ALLOCATED (IOTERR 4514)
CAUSE	I/O Test Tool was not able to obtain data space to be used as the parameter block of the HP-IB device adapter program.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause for those error message for further information.
4515	FIRST BYTE OF THE STATUS BUFFER IS RESERVED FOR HSTAT (IOTERR 4515)
CAUSE	The first byte of the status buffer is reserved for return of HSTAT from the HP-IB device adapter. This hYTE cannot be used for any other purpose.
ACTION	Please consult the I/O Test Tool manual for furtber information on the structure of a HP-IB Device adapter program.
4516	WARNING: DAP BUFFER HAS BEEN REDEFINED (IOTERR 4516)
CAUSE	I/O Test Tool has redefined the buffer named DAP. This buffer is required for an HP-IB device adapter program.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETHPIB instruction.
4517	EXCEEDED MAXIMUM LENGTH OF HP-IB STATUS BUFFER (IOTERR 4517)
CAUSE	User has tried to build a HP-IB device adapter program with a SINDE ~~X~~ parameter larger than the hardware will allow.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.



4518	WARNING: Status buffer is too small for DAP execution.	
CAUSE	User is trying to build a HP-IB device adapter program which contains a SINDEXX larger than the size of the status buffer.	
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.	
4519	WARNING: Non-language localized buffer names will be used.	
CAUSE	I/O Test Tool was not able to obtain the buffer names for the buffer used in HP-IB device adapter programs. I/O Test Tool will continue execution using non-language localized buffer names.	
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.	
4520	PINDEX PARAMETER IS OUT OF RANGE (IOTERR 4520)	
CAUSE	User has tried to build an HP-IB device adapter program containing PINDEXX parameter larger than allowed.	
ACTION	Please consult the I/O Test Tool manual for further information on the structure of an HP-IB device adapter program.	
4521	CALCULATED RELATIVE OFFSET IS OUT OF RANGE (IOTERR 4521)	
CAUSE	User has tried to create a HP-IB device adapter program where the relative offset was beyond the range of a byte.	
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-IB device adapter program.	
4750	SETHPCIO INSTRUCTION HAS NOT BEEN PERFORMED (IOTERR 4750)	
CAUSE	The user has tried to issue an HPCIO instruction before a SETHPCIO instruction was given.	
ACTION	Please consult the I/O Test Tool help facility for further information on the SETHPCIO command.	
4752	NO HP-CIO INSTRUCTIONS HAVE BEEN GIVEN (IOTERR 4752)	
CAUSE	User has issued a ENDHPCIO instruction but there were no HPCIO instructions given.	
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HPCIO DMA chain.	
4753	QUADS COULD NOT RELEASED (IOTERR 4753)	
CAUSE	I/O Test Tool was not able to return data space to the operating system.	
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.	

4754	COULD NOT OBTAIN QUAD. SYSTEM ERROR = ! (IOTERR 4754)
CAUSE	I/O Test Tool could not obtain data space for an HPCIO quad.
ACTION	The system error returned from the operating system will be displayed with this message. Please report all error messages displayed to support personnel.
4755	INVALID HP-CIO ORDER WAS GIVEN. (IOTERR 4755)
CAUSE	User issued an ADDQUAD instruction with an unknown order identifier.
ACTION	Please consult the I/O Test Tool help facility for further information on the ADDQUAD instruction.
4756	HP-CIO DMA CHAIN WILL BE OVERWRITTEN (IOTERR 4756)
CAUSE	The user has issued an SETHPCIO instruction after a ENDHPCIO instruction has been processed. The current HP-CIO DMA chain will be deleted.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETHPCIO instruction.
4757	INTERNAL ERROR IEMHPIO : ! (IOTERR 4757)
CAUSE	I/O Test Tool has encountered an error it could not recover from. I/O Test Tool cannot continue execution.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.
4758	HP-CIO OPTIONS NOT ALLOWED HERE (IOTERR 4758)
CAUSE	The HP-CIO options given are not allowed with the HP-CIO order given.
ACTION	Please consult the I/O Test Tool help facility for further information on the ADDQUAD instruction.
4759	UNABLE TO OBTAIN HPCIO DATA OBJECT (IOTERR 4759)
CAUSE	An error status was returned when I/O Test Tool tried to obtain a data object to store state information for the DMA chain being built.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause for those error messages for further information.
4760	UNABLE TO RELEASE HPCIO OBJECT (IOTERR 4760)
CAUSE	An error status was returned when I/O Test Tool tried to return a data object used to store state information for the DMA chain being built.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause for those error messages for further information.
5250	SETDEVICE INSTRUCTION HAS NOT BEEN PERFORMED (IOTERR 5250)
CAUSE	The user has requested a function which requires a device to be allocated.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETDEVICE instruction.

5253	I/O DEVICE UNKNOWN TO THIS UTILITY (IOTTERR 5253)	
CAUSE	The device given is unknown to the utility and therefore cannot be accessed.	
ACTION	Please consult the I/O Test Tool manual for further information on the I/O devices supported by I/O Test Tool.	
5258	WARNING: EXCLUSIVE ACCESS COULD NOT BE OBTAINED, BUT DEVICE WAS ALLOCATED TO I/O TEST TOOL (IOTTWARN 5258)	
CAUSE	The diagnostic monitor was unable to lock the device. The user was allocated the device, but can perform non-destructive functions only.	
ACTION	Please consult the I/O Test Tool help facility for further information on the SETDEVICE instruction.	
5259	WARNING: TERMINATION CONDITION SETTING IGNORED (IOTTWARN 5259)	
CAUSE	A request to set the termination condition for the current device resulted in an error. Therefore, when the user exits or releases the device, no re-initialization of the device will be performed.	
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.	
5260	NO STATUS BUFFER HAS BEEN DEFINED (IOTTERR 5260)	
CAUSE	The user requested an I/O to be performed without specifying a status buffer. The status buffer is used to store the hardware status when an error is encountered.	
ACTION	Please consult the I/O Test Tool help facility for further information on the DEFBUFF instruction.	
5261	CURRENT DEVICE NOT DEFINED (IOTTERR 5261)	
CAUSE	The user requested an I/O to be performed before a device was specified.	
ACTION	Please consult the I/O Test Tool help facility for further information on the SETDEVICE instruction.	
5262	WARNING: DEVICE IS ON THE COLD LOAD PATH. EXCLUSIVE ACCESS WILL BE OBTAINED WHEN A DATA DESTRUCTIVE I/O IS REQUESTED. (IOTTWARN 5262)	
CAUSE	The user has selected a device on the load path. The device will not be placed into a locked state until an I/O request is issued through the EXECUTE command. This means each I/O to this device will be bounded with a lock/unlock.	
ACTION	Please consult the I/O Test Tool manual for further information on accessing devices on the load path.	

5263	ABORTIO OF UNBLOCKED I/O'S FAILED. BUT SETDEVICE WILL CONTINUE (IOTERR 5263)
CAUSE	I/O Test Tool encountered an error while trying to abort any unblocked I/O requests before processing the SETDEVICE instruction for a new device.
ACTION	SETDEVICE will still be performed.
5264	WARNING: EXCLUSIVE ACCESS CANNOT BE OBTAINED IN MULTI-USER MODE, BUT DEVICE WAS ALLOCATED TO I/O TEST TOOL. (IOTTWARN 5264)
CAUSE	The diagnostic system was not able to allocate the device to I/O Test Tool. I/O Test Tool will allow the user to access the device, but the user will not be able to perform data destructive functions.
ACTION	To perform data destructive functions the user will need to enter single user mode, or single disk mode.
5500	I/O REQUEST INVALID FOR DEVICE (IOTERR 5500)
CAUSE	The device is incapable of performing the function requested.
ACTION	Please consult the I/O Test Tool help facility for a list of all I/O requests appropriate for the desired device.
5501	I/O REQUEST CAN ONLY BE PERFORMED IF DEVICE WAS LOCKED (IOTERR 5501)
CAUSE	The device must be in a locked state before I/O Test Tool can issue the type of I/O requested.
ACTION	To perform data destructive functions the user will need to enter single user mode, or single disk mode.
5502	SETOPTION OF OPTION 2 IS REQUIRED (IOTERR 5502)
CAUSE	The user has issued an EXECUTE instruction for a I/O request which requires a SETOPTION instruction with two parameters. The user has either not given the SETOPTION command, or has given SETOPTION instruction with one parameter.
ACTION	The SHOWPARM instruction will display the current settings from the SETOPTION instruction. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5503	SETOPTION OF OPTION 1 IS REQUIRED (IOTERR 5503)
CAUSE	The user has issued an EXECUTE instruction for a I/O request which requires a SETOPTION instruction with one parameter. The user has not given the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5504	SETDATA REQUIRED FOR I/O REQUEST (IOTERR 5504)
CAUSE	The I/O requested by the user requires that a SETDATA instruction to be performed.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETDATA instruction.
5505	WARNING: SETADDR not performed, reserved disc area will be used
CAUSE	The user has issued an I/O request which requires an address, but the user did not issue the SETADDR instruction. Therefore, the reserved disk area will be used.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETADDR instruction.
5506	HARDWARE STATUS COULD NOT BE OBTAINED (IOTERR 5506)
CAUSE	The user has issued an I/O request, but the request failed and I/O Test Tool was not able to obtain hardware status information from the device.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause of those error messages for further information.
5509	REQUEST CAN ONLY BE PERFORMED IN SINGLE USER MODE (IOTERR 5509)
CAUSE	The I/O requested by the user can only be executed in single user mode.
ACTION	Please return to the diagnostic user interface and enter single user mode before issuing the request again.
5510	STATUS BUFFER HAS NOT BEEN DEFINED (IOTERR 5510)
CAUSE	The user has issued an I/O request, but there is no status buffer defined.
ACTION	Please consult the I/O Test Tool help facility for further information on the DEFBUFF instruction.
5511	INTERNAL ERROR: - EXECUTE ! (IOTERR 5511)
CAUSE	I/O Test Tool has encountered an error it cannot recover from. This is a software error.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.
5512	A SETDEVICE INSTRUCTION WAS NOT EXECUTED (IOTERR 5512)
CAUSE	The user has issued an I/O request, but the SETDEVICE instruction was not given.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETDEVICE instruction.
5513	EXPECTING SELFTEST NUMBER IN SETOPTION 1 (IOTERR 5513)
CAUSE	The user has issued an I/O request, but the I/O requires that a selftest number be given in the first parameter of the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5514	EXPECTING LOOP COUNT IN SETOPTION 2 (IOTERR 5514)
CAUSE	The user has issued an I/O request, but that request requires that a loop count be given as the second parameter of the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions need to be issued for a particular I/O.
5515	EXPECTING "READ" OR "WRITE" OPTION IN SETOPTION 1 (IOTERR 5515)
CAUSE	The user has issued an I/O request, but that request requires that the reserved word READ or WRITE be given as the first parameter of the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions need to be issued for a particular I/O.
5516	EXPECTING UNIT NUMBER IN SETOPTION 1 (IOTERR 5516)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be the unit number of the device.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions need to be issued for a particular I/O.
5517	EXPECTING BURST NUMBER IN SETOPTION 1 (IOTERR 5517)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction to be the number of bursts.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions need to be issued for a particular I/O.
5518	EXPECTING LENGTH COUNT IN SETOPTION 1 (IOTERR 5518)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction to specify a length.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions need to be issued for a particular I/O.
5519	EXPECTING "TIMEON" OR "TIMEOFF" OPTION IN SETOPTION 1 (IOTERR 5519)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction to be the reserved word TIMEON or TIMEOFF.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5520	EXPECTING "AUTOON" OR "AUTOOFF" OPTION IN SETOPTION 2 (IOTERR 5520)
CAUSE	The user has issued an I/O request, but that request requires that the second parameter of the SETOPTION instruction be the reserved word AUTOON or AUTOOFF.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5521	EXPECTING "VECTOR1" OR "VECTOR2" OPTION IN SETOPTION 1 (IOTERR 5521)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be the reserved word VECTOR1 or VECTOR2.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5522	EXPECTING RETRY COUNT IN OPTION 1 (IOTERR 5522)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be set to the retry count.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5523	EXPECTING VOLUME NUMBER IN OPTION 1 (IOTERR 5523)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be set to the volume number.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5524	EXPECTING "LAST" OR "ALL" OPTION IN OPTION 1 (IOTERR 5524)
CAUSE	The user has issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be set to the reserved words LAST or ALL.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5525	EXPECTING OPTION NUMBER IN OPTION 1 (IOTERR 5525)
CAUSE	The user issued an I/O request, but that request requires that the first parameter of the SETOPTION instruction be the option number.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5526	INTERNAL ERROR - INSTRUCTION EXECUTION MODULE : CS80 (IOTERR 5526)
CAUSE	I/O Test Tool has encountered an error it cannot recover from. This is a software error.
ACTION	Please submit a service request indicating the following: a) Any other error messages displayed with this error message. b) The sequence of IOTT commands which were being performed when this error message was encountered. c) The version (A.nn.nn) of the IOTT being executed.
5527	EXPECTING "PARITYON" OR "PARITYOFF" IN SETOPTION 1 (IOTERR 5527)
CAUSE	I/O Test Tool was expecting either the reserved words PARITYON or PARITYOFF as the first parameter of the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5528	EXPECTING "SRQON" OR "SRQOFF" SETOPTION 2 (IOTERR 5528)
CAUSE	I/O Test Tool was expecting either the reserved words SRQON or SRQOFF as the second parameter of the SETOPTION instruction.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5529	TO EXECUTE SELFTEST - USE WRITE WITH ESCAPE - Z (IOTERR 5529)
CAUSE	I/O Test Tool cannot initiate selftest on a line printer.
ACTION	Issue a write I/O request with a data buffer containing a CIPER data packet with ESCAPE z as data. Please consult the I/O Test Tool manual for further information on protocol to a line printer.
5530	DATA BUFFER TOO SMALL TO PERFORM FUNCTION (IOTERR 5530)
CAUSE	The user has specified a data buffer too small to perform the I/O requested.
ACTION	Please consult the I/O Test Tool help facility for any special size requirements for the I/O requested.
5531	EXPECTING TIME1 PARAMETER IN OPTION 1 (IOTERR 5531)
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to contain a time value.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5532	EXPECTING TIME2 PARAMETER IN OPTION 2 (IOTERR 5532)
CAUSE	I/O Test Tool was expecting the second parameter of the SETOPTION instruction to contain a time value.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5533	UNBLOCK OPTION IGNORED, I/O PERFORMED IN BLOCK MODE (IOTERR 5533)
CAUSE	The I/O device the user has selected does not handle unblocked I/O requests.
ACTION	Please issue the I/O request again without the UNBLOCK option.
5534	EXPECTING VERIFY LENGTH IN OPTION 1 (IOTERR 5534)
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to contain the length of transfer.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5535	INTERNAL ERROR: PDEV WAS NOT OF VALID FORMAT (IOTERR 5535)
CAUSE	I/O Test Tool encountered an error it could not recover from. This is a software error.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
5539	DEVICE ADAPTER IS NOT HP-IB (IOTERR 5539)
CAUSE	The user tried to issue an EXECUTE HPIB instruction against a device adapter which is not HPIB.
ACTION	Please select a device which is connected to an HP-IB device adapter.
5542	RELEASE OF HP-IB DEVICE ADAPTER MANAGER FAILED (IOTERR 5542)
CAUSE	I/O Test Tool was not able to release the HP-IB device adapter manager. Additional error messages containing further information will be displayed with this message.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
5543	BUILDING OF HP-IB DAP HAS NOT BEEN COMPLETED (IOTERR 5543)
CAUSE	The user has tried to issue a EXECUTE HPIB instruction before the building of the HP-IB device adapter program has been completed.
ACTION	Please consult the I/O Test Tool help facility for further information on the ENDPPIB instruction.



5553	HP-CIO DMA CHAIN HAS NOT BEEN BUILT (IOTERR 5553)
CAUSE	The user has tried to issue an EXECUTE HPCIO instruction but a HP-CIO DMA chain has not been built.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a HP-CIO DMA chain.
5557	STATUS BUFFER NOT LARGE ENOUGH TO EXECUTE HP-IB DEVICE ADAPTER PROGRAM (IOTERR 5557)
CAUSE	One instruction within the HP-IB device adapter program has specified a SINDEIX larger than the status buffer length.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of an HP-IB device adapter program.
5558	I/O TEST TOOL GENERATED IORESULT REQUEST FAILED (IOTERR 5558)
CAUSE	I/O Test Tool encountered a failure while processing the result of an unblocked I/O.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause of those error messages for further information.
5559	! INSTRUCTIONS WERE EXECUTED (IOTERR 5559)
CAUSE	Indicates the number of I/O requests which were executed before a failure was detected.
ACTION	Error messages containing further information on the failure will be displayed with this message. Please consult the ACTION clause of those error messages for further information.
5560	BLOCKED I/O REQUEST WAS NOT ISSUED (IOTERR 5560)
CAUSE	I/O Test Tool has not obtained all the information it needs to perform the I/O request.
ACTION	Error messages indicating the information lacking will be displayed with this message. Please refer to the ACTION clause for those error messages for further information.
5561	HP-IB ADDRESS IS OUT OF RANGE (IOTERR 5561)
CAUSE	I/O Test Tool has encountered a HP-IB address which is beyond the range of 0 to 7. This is a software error.
ACTION	Please submit a service request indicating the following: a) Any other error messages displayed with this error message. b) The sequence of IOTT commands which were being performed when this error message was encountered. c) The version (A.nn.nn) of the IOTT being executed.
5566	EXPECTING TAPE DENSITY IN OPTION 1 (IOTERR 5566)
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to be a tape density value.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.

5567	INVALID TAPE DENSITY VALUE GIVEN (IOTERR 5567)	
CAUSE	The tape density value given by the user though the SETOPTION instruction is invalid. Tape density values must be 800 or 1600 or 6250.	
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.	
5568	EXPECTING TAPE SPEED SELECT IN OPTION 1 (IOTERR 5568)	
CAUSE	I/O Test Tool is expecting the first parameter of the SETOPTION instruction to be the tape speed select value.	
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.	
5569	INVALID TAPE SPEED SELECT GIVEN (IOTERR 5569)	
CAUSE	The tape speed parameter given in the SETOPTION instruction is invalid. Tape speed values can be 100 or 50.	
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.	
5570	EXPECTING "ENABLE" OR "DISABLE" IN OPTION 1 (IOTERR 5570)	
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to be either of the reserved words ENABLE or DISABLE.	
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.	
5571	EXPECTING "HARD" OR "SOFT" IN OPTION 1 (IOTERR 5571)	
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to contain either of the reserved words HARD or SOFT.	
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.	
5572	UNABLE TO LOCK NEXT LEVEL I/O MANAGER (IOTERR 5572)	
CAUSE	I/O Test Tool was not able to lock the next level I/O manager to perform the EXECUTE instruction.	
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.	

5573	ORIGINAL DEVICE IS WAS NOT LOCKED (IOTERR 5573)
CAUSE	I/O Test Tool cannot perform an EXECUTE HPIB or an EXECUTE HPCIO unless the device is in a locked state.
ACTION	It may be necessary to enter single user mode to perform this request.
5574	INVALID OPTION 1 FOR GENERIC I/O REQUEST (IOTERR 5574)
CAUSE	I/O Test Tool was expecting the first parameter of the SETOPTION instruction to be a literal value.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5575	INVALID OPTION 2 FOR GENERIC I/O REQUEST (IOTERR 5575)
CAUSE	I/O Test Tool was expecting the second parameter of the SETOPTION instruction to be a literal value.
ACTION	The SHOWPARM instruction will display the current settings from SETOPTION. Please consult the I/O Test Tool help facility for further information on what instructions needs to be issued for a particular I/O.
5576	UNABLE TO DOWNLOAD DEVICE ADAPTER PROGRAM (IOTERR 5576)
CAUSE	I/O Test Tool was not able to download the user's device adapter program onto the HP-IB device adapter.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause of those error messages for further information.
5577	! STATUS BYTES EXCEED MAXIMUM ! BYTES ALLOCATED BY I/O TEST TOOL. STATUS BUFFER ALTERATION NOT PERFORMED (IOTERR 5577)
CAUSE	The status buffer allocated by I/O Test Tool is too small for the hardware status which will be returned by the device. This is a software error.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.
5578	SETADDR REQUIRED FOR I/O REQUEST (IOTERR 5578)
CAUSE	User trying to request an I/O, but did not issue a SETADDR instruction to specify an address. An address is required to perform the I/O request.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETADDR instruction.
5579	IOTT GENERATED SET ADDRESS COMMAND FAILED (IOTERR 5579)
CAUSE	I/O Test Tool issued a CS/80 set address command to the current device, and that I/O failed.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause of those error messages for further information.

5750	BUFFER TYPES MUST BE IDENTICAL FOR IF STATEMENTS (IOTERR 5750)	
CAUSE	For the IF and IFN program statements, buffer elements used as operands must be of the same type.	
ACTION	Please consult the I/O Test Tool help facility for further information on the IF or IFN program statement.	
5751	BUFFER GIVEN IS OF TYPE BYTE AND THE VALUE GIVEN IS INTEGER (IOTERR 5751)	
CAUSE	The user has tried to perform integer functions with a buffer of type BYTE.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.	
5752	LOOP TO STATEMENT ENCOUNTERED BEFORE DO STATEMENT WAS EXECUTED (IOTERR 5752)	
CAUSE	During execution of the program, a LOOP TO program statement was encountered before a DO program statement.	
ACTION	Please consult the I/O Test Tool help facility for further information on the DO-LOOP TO program statements.	
5753	INTERNAL ERROR IEMPGM : ! (IOTERR 5753)	
CAUSE	I/O Test Tool encountered an error it could not recover from. I/O Test Tool cannot continue execution.	
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.	
5998	INTERNAL ERROR: ! (IOTERR 5998)	
CAUSE	I/O Test Tool encountered an error it could not recover from. This is a software error.	
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.	
5999	ERROR DETECTED AT LINE NUMBER ! (IOTERR 5999)	
CAUSE	Indicates which line within a program produced an error. When displayed, the ! symbol will be replaced with the actual line number.	
ACTION	Error messages indicating the instruction error will be displayed with this message. Please consult the ACTION clause of those error messages for further information.	
6000	COMMANDS CANNOT BE USED WITH A LOAD FILE (IOTERR 6000)	
CAUSE	The user tried to load a file through the LOAD command which contained commands. Only instructions and program statements can be loaded from a file.	
ACTION	Please consult the I/O Test Tool help facility for further information on the LOAD command.	

6001	COMMANDS CANNOT BE PRECEDED BY A LINE NUMBER (IOTERR 6001)
CAUSE	The user gave a command preceded by a line number. Only instructions and program statements can be stored into the Program Storage Area.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a program.
6002	ALL INPUT FROM THE LOAD FILE MUST BE PRECEDED BY A LINE NUMBER (IOTERR 6002)
CAUSE	The file designated by the LOAD command contained instructions or program statements which were not preceded by a line number.
ACTION	Please consult the I/O Test Tool help facility for further information on the LOAD command.
6003	PROGRAM STATEMENTS MUST BE PRECEDED BY LINE NUMBERS (IOTERR 6003)
CAUSE	The user has given a program statement without a leading line number. Since program statements must be inserted into the Program Storage Area, these statements must be preceded by a line number.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a program.
6004	INVALID LINE NUMBER ENCOUNTERED (IOTERR 6004)
CAUSE	The line number given was outside the range of a valid line number. A valid line number is between 1 and 9999.
ACTION	Please consult the I/O Test Tool manual for further information on the structure of a program.
6005	UNRECOGNIZED COMMAND (IOTERR 6005)
CAUSE	The input given by the user is not recognized as an I/O Test Tool command, instruction, or program statement.
ACTION	Please consult the I/O Test Tool help facility for a list of all commands available within I/O Test Tool.
6006	EXPECTING COMMA HERE (IOTERR 6006)
CAUSE	A mandatory comma was not found during syntax checking. The caret (^) will indicate the location where the comma was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6007	EXPECTING COLON HERE (IOTERR 6007)
CAUSE	A mandatory colon was not found during syntax checking. The caret (^) will indicate the location where the colon was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

6008	UNEXPECTED INPUT (IOTERR 6008)
CAUSE	Extra text was found after a syntactically correct input was found.
ACTION	Please enter all extra text as comments. To do this, enclose text within brackets.
6009	EXPECTING OPTION HERE (IOTERR 6009)
CAUSE	A mandatory option parameter was not given. Consult the HELP facility for information concerning the correct option for the input requested.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6010	EXPECTING BUFFER NAME HERE (IOTERR 6010)
CAUSE	A mandatory buffer name was not given. The caret (^) will indicate where the buffer name was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6011	BUFFER NAME EXCEEDS CHARACTER LIMIT OF 8 (IOTERR 6011)
CAUSE	The user gave a buffer name which exceeded the length of eight characters.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6012	EXPECTING LEFT PARENTHESIS FOR BUFFER INDEX HERE (IOTERR 6012)
CAUSE	A mandatory left parenthesis was not given. The caret (^) will indicate the location where the parenthesis was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6013	EXPECTING BUFFER INDEX HERE (IOTERR 6013)
CAUSE	A mandatory buffer index was not given. A buffer index must be a number. The caret (^) will indicate the location where the buffer index was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6014	EXPECTING RIGHT PARENTHESIS FOR BUFFER INDEX HERE (IOTERR 6014)
CAUSE	A mandatory right parenthesis was not given. The caret (^) will indicate the location where the parenthesis was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

6015	EXPECTING DATA VALUES (IOTERR 6015)
CAUSE	The input found was not a legal data value. Legal data values are numbers, constant expressions, and single ASCII characters.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6016	LINE NUMBERED INPUT CANNOT BE GIVEN IN PAUSE MODE (IOTERR 6016)
CAUSE	The user tried to enter new lines into the Program Storage Area while in PAUSE mode. This is illegal.
ACTION	Please consult the I/O Test Tool help facility for further information on the PAUSE instruction.
6017	DESTINATION NUMBER IN RANGE (IOTERR 6017)
CAUSE	The user has specified a destination line number which is within the range to be moved.
ACTION	Please consult the I/O Test Tool help facility for further information on the MOVE command.
6018	STARTING NUMBER IS GREATER THAN ENDING NUMBER (IOTERR 6018)
CAUSE	The user has given a line number range where the second value given is less than the first value.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.
6019	EXPECTING "SYNTAX" OPTION HERE (IOTERR 6019)
CAUSE	The user has given an illegal option for the HELP command.
ACTION	Please consult the I/O Test Tool manual for further information on the help facility.
6021	EXPECTING LINE RANGE OR ALL OPTION (IOTERR 6021)
CAUSE	An input other than a line range or the reserved word ALL has been given.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.
6022	EXPECTING DESTINATION LINE NUMBER (IOTERR 6022)
CAUSE	The user tried to perform a MOVE command without specifying where the lines are to be moved.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.

6023	EXPECTING DISPLAY OPTIONS HERE (IOTERR 6023)
CAUSE	The user gave a colon without the option identifier. In this case, only display options are valid. The options are HEX, DEC, or OCT.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6024	EXPECTING INDEX FOR THE BUFFER GIVEN (IOTERR 6024)
CAUSE	I/O Test Tool was expecting the next token to be a buffer index value. The caret (') will indicate the location where the buffer index value was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6027	EXPECTING DESTINATION LINE NUMBER (IOTERR 6027)
CAUSE	I/O Test Tool was expecting the next token to be a destination line number. The caret (') will indicate the location where the line number was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.
6028	LENGTH PARAMETER GIVEN MUST BE GREATER THAN ZERO (IOTERR 6028)
CAUSE	The length parameter given was less than or equal to zero.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction given.
6029	EXPECTING LENGTH PARAMETER HERE (IOTERR 6029)
CAUSE	A mandatory length parameter was not given. The caret (') indicates where the length parameter was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6030	COUNT PARAMETER MUST BE GREATER THAN ZERO (IOTERR 6030)
CAUSE	The user gave a count parameter which was less than zero.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6031	ILLEGAL COUNT PARAMETER WAS GIVEN (IOTERR 6031)
CAUSE	I/O Test Tool was expecting the next token to be a count parameter. The caret (') will indicate the location where the count parameter was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6032	UNEXPECTED INTERNAL ERROR - GET INPUT MODULE ! (IOTERR 6032)
CAUSE	I/O Test Tool encountered an error it could not recover from. This is a software error.
ACTION	Please submit a service request indicating the following: a) The exact text of the above error message including the parameter. b) Any other error messages displayed with this error message. c) The sequence of IOTT commands which were being performed when this error message was encountered. d) The version (A.nn.nn) of the IOTT being executed.

6033	EXPECTING "TO" RESERVED WORD (IOTERR 6033)
CAUSE	The caret (^) will indicate where the reserved word should be located.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6034	EXPECTING "ALL" RESERVED WORD (IOTERR 6034)
CAUSE	The caret (^) will indicate where the reserved word should be located.
ACTION	Please consult the I/O Test Tool help facility for further information on the requirements for the I/O request issued.
6035	EXPECTING "OFF" RESERVED WORD (IOTERR 6035)
CAUSE	The caret (^) will indicate where the reserved word should be located.
ACTION	Please consult the I/O Test Tool help facility for further information on the requirements for the I/O request issued.
6036	EXPECTING "ON" RESERVED WORD (IOTERR 6036)
CAUSE	The caret (^) will indicate where the reserved word should be located.
ACTION	Please consult the I/O Test Tool help facility for further information on the requirements for the I/O request issued.
6037	EXPECTING A RESERVED WORD (IOTERR 6037)
CAUSE	A mandatory reserved word was not given by the user.
ACTION	Please consult the I/O Test Tool help facility for further information on the requirements for the I/O request issued.
6038	EXPECTING "THEN" RESERVED WORD (IOTERR 6038)
CAUSE	The caret (^) will indicate where the reserved word should be located.
ACTION	Please consult the I/O Test Tool help facility for further information on the requirements for the I/O request issued.
6039	EXPECTING FILE NAME (IOTERR 6039)
CAUSE	I/O Test Tool was expecting the next token to be a file name. The caret (^) will indicate the location where the file name was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6040	DISPLAY OPTION ALREADY GIVEN (IOTERR 6040)
CAUSE	I/O Test Tool was expecting the next token to be a display option. The caret (^) will indicate the location where the display option was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

6041		EXPECTING LINE NUMBER RANGE (IOTERR 6041)
CAUSE	I/O Test Tool was expecting the next token to be a line number range. The caret (^) will indicate the location where the line number range was expected.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.	
6042		EXPECTING "BYTE" OR "WORD" BUFFER TYPE (IOTERR 6042)
CAUSE	I/O Test Tool was expecting the next token to be either of the reserved word BYTE or WORD. The caret (^) will indicate the location where the reserved word was expected.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.	
6043		EXPECTING RESERVED WORD "STATUS" (IOTERR 6043)
CAUSE	I/O Test Tool was expecting the next token to be the reserved word STATUS. The caret (^) will indicate the location where the reserved word was expected.	
ACTION	Please consult the I/O Test Tool help facility for further information on the DEFBUFF instruction.	
6044		EXPECTING OPERATOR FOR IF STATEMENT (IOTERR 6044)
CAUSE	I/O Test Tool was expecting the next token to be an operator symbol. Legal symbols are "=" or "<>" or "AND" or "OR". The caret (^) will indicate the location where the operator was expected.	
ACTION	Please consult the I/O Test Tool help facility for further information on the IF or IFN program statement.	
6045		EXPECTING PRINT TEXT (IOTERR 6045)
CAUSE	Mandatory display text was not found.	
ACTION	Please consult the I/O Test Tool help facility for further information on the PRINT program statement.	
6046		"CYL" PARAMETER ALREADY GIVEN (IOTERR 6046)
CAUSE	The user has specified the CYL= key parameter twice. The caret (^) will indicate the location where the second occurrence of the parameter was found.	
ACTION	Please consult the I/O Test Tool help facility for further information.	
6047		"CYL" PARAMETER EXPECTED (IOTERR 6047)
CAUSE	I/O Test Tool was expecting the CYL= key parameter.	
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.	
6048		"HEAD" PARAMETER ALREADY GIVEN (IOTERR 6048)
CAUSE	The user has specified the HEAD= key parameter twice. The caret (^) will indicate the location where the second occurrence of the parameter was found.	
ACTION	Please consult the I/O Test Tool help facility for	

6049	"HEAD" PARAMETER EXPECTED (IOTERR 6049)
CAUSE	I/O Test Tool was expecting the HEAD= key parameter.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6050	"SECT" PARAMETER ALREADY GIVEN (IOTERR 6050)
CAUSE	The user has specified the SECT= key parameter twice. The caret (^) will indicate the location where the second occurrence of the parameter was found.
ACTION	Please consult the I/O Test Tool help facility for
6051	"SECT" PARAMETER EXPECTED (IOTERR 6051)
CAUSE	I/O Test Tool was expecting the SECT= key parameter.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6052	TIMEOUT VALUE (IN SECONDS) EXPECTED (IOTERR 6052)
CAUSE	I/O Test Tool was expecting the next token to be a timeout value. The caret (^) will indicate the location where the timeout value was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6053	IMPROPER OPTION GIVEN (IOTERR 6053)
CAUSE	I/O Test Tool has encountered an unknown option reserved word. The caret (^) will indicate the location where the unknown reserved word was found.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6054	INCREMENT VALUE EXPECTED (IOTERR 6054)
CAUSE	I/O Test Tool was expecting the next token to be a increment value. The caret (^) will indicate the location where the increment value was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6055	INCREMENT MUST BE POSITIVE (IOTERR 6055)
CAUSE	User has given a increment value which was less than or equal to zero. All increment values must be positive.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6056	TIME VALUE MUST BE POSITIVE (IOTERR 6056)
CAUSE	The time value given must be greater that zero.
ACTION	Please consult the I/O Test Tool help facility for further information on the SETTIMER instruction.

6057	INCREMENT CANNOT BE ZERO (IOTERR 6057)
CAUSE	All increment values must be a value other than zero.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6058	COMMAND CANNOT BE ISSUED IN PAUSE MODE (IOTERR 6058)
CAUSE	Users can issue instructions in pause mode. Commands are not allowed.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.
6059	BUFFER INDEX MUST NOT BE NEGATIVE (IOTERR 6059)
CAUSE	The user has given a negative buffer index. All buffer index values start at zero.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6060	EXPECTING "UNBLOCK" OPTION (IOTERR 6060)
CAUSE	I/O Test Tool was expecting the next token to be the reserved word UNBLOCK.
ACTION	Please consult the I/O Test Tool help facility for further information for the EXECUTE instruction.
6061	INVALID PREDEFINED I/O IDENTIFIER (IOTERR 6061)
CAUSE	This I/O identifier is unknown to the program.
ACTION	Please consult the I/O Test Tool help facility for a list of valid I/O identifiers.
6062	EXPECTING A DEVICE SPECIFIER (IOTERR 6062)
CAUSE	I/O Test Tool was expecting a LDEV= or PDEV= parameter. The caret (^) will indicate the location where the device specifier was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6063	EXPECTING A PHYSICAL DEVICE NUMBER (PDEV) (IOTERR 6063)
CAUSE	I/O Test Tool was expecting a physical device number after the PDEV= key word.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6064	EXPECTING A LOGICAL DEVICE NUMBER (LDEV) (IOTERR 6064)
CAUSE	I/O Test Tool was expecting a logical device number after the LDEV= key word.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

6065	NOT ALL ADDRESS PARAMETERS WERE GIVEN (IOTERR 6065)
CAUSE	The user has given an incomplete address parameter. The address consists of a CYL=, HEAD=, and SECT= key parameters.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6066	EXPECTING "ON" OR "OFF" OPTION (IOTERR 6066)
CAUSE	I/O Test Tool was expecting the reserved word ON or OFF. The caret (^) will indicate the location where the reserved word was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6067	EXPECTING LINE NUMBER HERE (IOTERR 6067)
CAUSE	I/O Test Tool was expecting the next token to be a line number. The caret (^) will indicate the location where the line number was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6068	VALUE CANNOT BE NEGATIVE (IOTERR 6068)
CAUSE	The user has given a negative value parameter, but the instruction will only access positive values.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6070	ILLEGAL KEY WORD (IOTERR 6070)
CAUSE	The key word given is unknown to this program.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6071	FILE NAME TOO LONG (IOTERR 6071)
CAUSE	The file name given exceeded the maximum character length. Remember the file must reside in the diagnostic group and and account.
ACTION	Please consult the I/O Test Tool help facility for further information on the command issued.
6072	FILE SELECTOR ID CONTAINS INVALID CHARACTERS OR IS TOO LONG (IOTERR 6072)
CAUSE	The file selector parameter given to the SHOWFILE command was too long.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.

6073	EXPECTING VALUE PARAMETER (IOTERR 6073)
CAUSE	The required parameter for a HP-IB device adapter program instruction was not given.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6074	VALUE MUST BE WITHIN THE RANGE OF 0 to 255 (IOTERR 6074)
CAUSE	All HP-IB device adapter program parameters must be contained within 8-bits.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6075	EXPECTING "EOI" OPTION (IOTERR 6075)
CAUSE	I/O Test Tool was expecting the reserved word EOI as an option. The caret (^) will indicate the location where the reserved word was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6076	EXPECTING "BREAK" OPTION (IOTERR 6076)
CAUSE	I/O Test Tool was expecting the next token to be the reserved word BREAK. The caret (^) will indicate the location where the reserved word was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the instruction issued.
6077	HP-CIO OPTION ALREADY GIVEN (IOTERR 6077)
CAUSE	User has already specified the HP-CIO option in the list. The caret (^) will indicate the location where the duplicate option was found.
ACTION	Please consult the I/O Test Tool help facility for further information on the ADDQUAD instruction.
6078	EXPECTING HP-CIO OPTION (IOTERR 6078)
CAUSE	I/O Test Tool was expecting a HP-CIO option list. The caret (^) will indicate the location where the HP-CIO option list was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the ADDQUAD instruction.
6079	EXPECTING HP-CIO ORDER (IOTERR 6079)
CAUSE	I/O Test Tool was expecting a HP-CIO order identifier. The caret (^) will indicate the location where the HP-CIO order identifier was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the ADDQUAD instruction.

6080	EXPECTING "DIFF" OR "SIMILAR" WORD (IOTERR 6080)
CAUSE	I/O Test Tool was expecting the reserved word DIFF or SIMILAR. The caret (^) will indicate the location where the reserved word was expected.
ACTION	Please consult the I/O Test Tool help facility for further information on the COMPBUFF command.
8000	OBTAINED UNEXPECTED MESSAGE DURING STATUS DECODING (IOTERR 8000)
CAUSE	I/O Test Tool obtained an unexpected inter-process communication message from the diagnostic program decoding status.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
8001	DIAGNOSTIC MONITOR WAS NOT ABLE TO TERMINATE DECODING PROCESS (IOTERR 8001)
CAUSE	I/O Test Tool encountered an error while trying to terminate a diagnostic program which was launched for decode hardware status.
ACTION	Error messages containing further information will be displayed with this message. Please consult the ACTION clause for those error messages for further information.
8002	BUFFER DISPLAY TERMINATED DUE TO NUMBER CONVERSION PROBLEMS (IOTERR 8002)
CAUSE	I/O Test Tool was not able to convert an integer to a string.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
8003	UNABLE TO OBTAIN MESSAGE FROM CATALOG I/O TEST TOOL CANNOT CONTINUE EXECUTION (IOTERR 8003)
CAUSE	I/O Test Tool was not able to obtain a message from the message catalog.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
8004	UNABLE TO DISPLAY ASCII HEADER (IOTERR 8004)
CAUSE	I/O Test Tool was not able to display the header when a DBUFF instruction is given for a byte buffer.
ACTION	Enable error tracing by rerunning the program with the TRACE=ERROR parameter at the DUI command line, and file a service request with the trace information generated when performing the IOTT command again.
8005	MAXIMUM ADDRESS COULD NOT BE OBTAINED. INCREMENTATION NOT PERFORMED (IOTERR 8005)
CAUSE	I/O Test Tool will issue a DESCRIBE command to the CS80 disk to obtain the maximum address possible for the particular disk. In this instance, the DESCRIBE request failed.
ACTION	Error messages containing further information will be displayed with this message. Please refer to the ACTION clause of those error messages for further information.



Contents

6. MPE XL Online Diagnostic Installer	
Introduction	6-1
Defects and Enhancements	6-1
Minimum Configuration	6-2
Operating Instructions	6-2
RUN Command	6-2
Test Execution	6-3
Command Summary	6-5
ADD	6-6
CORRECT	6-8
EXIT	6-11
LIST	6-12
REMOVE	6-13
SHOWMSG	6-14
SYSTEM	6-15
XCHECK	6-16
Warning Messages	6-18
Error Messages	6-22

MPE XL Online Diagnostic Installer

Introduction

The Online Diagnostic Installer is part of the MPE XL Online Diagnostic Subsystem. It allows the user to add, remove, and correct program information entries in the directory, and add, remove, and replace message sets in the message catalog (including those for the Diagnostic User Interface - DUI, System Monitor, and diagnostic system services). Updatable directory information includes:

- Security clearance
- Single user mode (SUM)
- Device requirements
- Type of program
- Program source code
- Autodiagnostic/decode ability of diagnosable devices.

The Installer updates and maintains other structures in the directory which are transparent to the system user but which are needed by the DUI and the system monitor. The Installer is written in menu driven form for simple operation. This utility will serve both as a remote and on-site support tool.

Updating the Online Diagnostic Subsystem requires the user to be logged onto the SYS account. Diagnostic security level 0 is also required. Since the Installer modifies core system files, actual installation will not occur (disallowed by the subsystem monitor) unless all other diagnostic users, including auto-diagnostics, are logged off the system.

The above restrictions do not apply in those instances where the installer is not being used to update the actual Online Diagnostic Subsystem files. Such a situation exists when the user either logs onto an account other than SYS and/or specifies a group other than DIAG in the RUN command "info" string when invoking the installer. The user is responsible for ensuring that the necessary diagnostic files are copied to an alternate group. Since the installer creates a number of working files, the integrity of non-diagnostic files in this alternate group cannot be guaranteed.

Defects and Enhancements

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

Minimum Configuration

The MPE XL Online Diagnostic Installer will run on any HP Precision Architecture Computer system using the MPE XL operating system. The MPE XL Online Diagnostic Subsystem must be supported.

Operating Instructions

The user runs the installer as a stand-alone utility and must have diagnostic level 0 capability (see the Online Diagnostics Overview). The Installer is not invoked from the DUI. It is a separate utility which interfaces with the diagnostic monitor but is run from the MPE XL operating system. The installer will prompt the user for the data it needs. All prompts will end with the ">" character.

Entry of the specific data will always terminate with a carriage return (Return). File names and file name qualification follow standard MPE XL rules. All characters are converted to upper case, making case selection unnecessary and impossible. Only as many characters as are necessary to differentiate one command from others need be entered. Normally this is two or three. If too few characters are entered, an "unrecognized command" error message is displayed.

Selection of tasks will be accomplished by entering the command name corresponding to a desired menu function. A null entry (Return) at program menu prompts will simply result in a reprompt for a valid selection. Other data items will have an explanatory prompt which will usually include a range of acceptable responses enclosed in parentheses "(" and ")" and a default value enclosed in brackets "[" and "]" to be assumed if the user simply enters a carriage return. Normally, this default will be the current value of the item as it presently resides in the directory. The installer makes each major change (program addition, deletion, or correction) permanent before allowing another change. This will normally involve reconstruction of the diagnostic system message catalog as well as the installer's work files.

Assistance on the use of this program is available by entering HELP at any prompt. Also available are:

- HELP GENERAL
- HELP COMMANDS
- HELP HELP
- HELP RECOVERY

The remainder of this manual explains the function corresponding to each main menu command. Explanations will include a brief overview of the purpose, an explanation of each step within the function, and an example of the expected output. The error messages listed in the last section of this manual are installer-specific. However, other error messages may also be printed.

RUN Command

To run the Installer, enter the following to the MPE XL prompt (The "info" string will be mandatory if the user is logged onto an account other than SYS):

```
:run diaginst.diag.sys;lib=g [;info="<group name>"]
```

6-2 MPE XL Online Diagnostic Installer

Test Execution

The following introductory message will be displayed:

```
*****
****
****      MPE XL ONLINE DIAGNOSTIC INSTALLER      ***
****
****      (C) Copyright Hewlett Packard Co. 1987   ***
****              All Rights Reserved.           ***
****              Version A.00.00                 ***
****
*****
```

Initializing... .

```
-----
Type "HELP HELP" at any prompt for instructions
concerning the Installer's help facility.
-----
```

After a short wait for initialization, the main menu will be displayed:

Available Commands:

```
ADD      - Add a new program to the subsystem
CORRECT  - Correct an existing program entry
EXIT     - Return control to the operating system
LIST     - Display info for all programs in the subsystem
REMOVE   - Remove a program from the subsystem
SHOWMSG  - Display messages from the subsystem catalog
SYSTEM   - Update diagnostic subsystem core files
XCHECK   - Cross-check subsystem files for synchronization
```

INSTALLATION TASK (select by command name) >

If the program being installed will auto-diagnose a device and/or decode status for it, one or both of the following messages will appear after the device name is entered. A YES answer makes the current program the designated autodiagnostic/decoder. A NO answer leaves the previous designation intact:

```
"(device name)" is already auto-diagnosed.
That service is currently provided by "(module name)"
OVERRIDE? (YES/NO) ☐ NO >
```

```
"(device name)" is already decoded.
That service is currently provided by "(module name)"
OVERRIDE? (YES/NO) ☐ NO >
```

Since catalog manipulations take a considerable amount of time, creation of a working copy of the catalog is put off until actually needed.

Generating working catalog (roughly ! min.)....

The installer will be capable of recovering from a variety of errors. This recovery will be indicated by:

Recovery in progress...

If the installer cannot recover the directory, the user will be given the opportunity to create a new directory from scratch (though in most cases will probably prefer to replace it from backup via NEWDIR):

```
*** WARNING - The Diagnostic Subsystem directory is missing ...
                                     (INSTWARN 05514)
Installer found a working directory left over from the last attempt
to install. Attempting to use that directory....

or

*** WARNING - The Diagnostic Subsystem directory is missing ...
                                     (INSTWARN 05514)
Do you wish to build a catalog from scratch (YES/NO) ☐ NO >
Building new directory from scratch ...
A new directory has been created ...
```

The installer must on occasion communicate with the diagnostic system monitor. When this communication process breaks down, one or more of the following messages will be issued. When a timeout has occurred, the installer will usually give the user the option to wait longer or to try again. This will normally occur when there is a process other than the installer (and system monitor) running in the diagnostic system. The user should ensure other users are logged off before retrying an installation.

The following advisories and prompts are issued along with explanatory error/warning messages:

```
There are currently ! other users on the diagnostic subsystem.
Wait another ! seconds?      (YES/NO) ☐ YES >
Do you wish to try again?    (YES/NO) ☐ YES >
```

When the installer detects an error in user-entered data, one of the following explanatory advisories will accompany an "invalid response" error message prior to reprompting for that item:

```
--Something other than a number was entered.
--Your choice must fall between ! and !.
--Only commands from the current menu are now valid.
To see the current menu, type "REFRESH".
```

One or more of the following messages will follow errors issued by GENCAT when attempting to update the subsystem catalog:

```
--The above error is often the result of a missing message
continuation symbol ("% or "&") before an unnumbered line.
--For more details, consult the Diagnostic Program Development Guide.
```

Command Summary

The following commands are available in the Online Diagnostic Installer:

ADD	Adds a program to the Online Diagnostic Subsystem.
CORRECT	Updates user supplied data.
EXIT	Terminates the installer.
LIST	Displays user correctable data for all diagnostic programs in the subsystem.
REMOVE	Purges a module from the diagnostic system.
SHOWMSG	Displays a particular subsystem message catalog.
SYSTEM	Replaces core message sets and restores entire subsystem directory, or message catalog while the system is up and running.
XCHECK	Displays the contents of the directory, subsystem catalog, and diagnostic account files against one another to check for inconsistencies and/or file corruption.

The Online Diagnostic Subsystem has reserved the following file names for its exclusive use. These names are therefore unavailable to the user for programming purposes.

CDUICAT	LAYOUTF	WORKCAT
CINSTCAT	LOGGERL	WORKDIR
DIAGINST	MEMLOGP	WORKGEN
DIAGLIST	NL	WORKLIST
DUI	TESTPROC	WORKNCAT
DUIDIR	WORKADD	WORKPRG
LAYMSG	WORKDEL	WORKSRC

ADD

This function will add a program to the Online Diagnostic Subsystem. A new entry will be added to the directory, a new message set added to the catalog, and a new program file copied into the diagnostic subsystem account. The following information will be needed for diagnostic module to be added:

- Name of file containing the executable code for the program
- Name of file containing the messages for the program
- Module name (<= 8 characters, first must be alphabetic)
- Module version number (normally in the form xx.xx)
- Module Type (diagnostic, exerciser, verifier, utility)
- Name/number of each product to be diagnosed by the program
- Whether program is an auto-diagnostic
- Whether program will decode device status
- Required User Mode (single or multi-user)
- Minimum security level required to run program

Output:

```
ADD A NEW DIAGNOSTIC ...

Program VERSION          (v.uu.ff) [?.??.??] >
Program NAME             [<cr> = abort] >
  Program Type Options:
    DIAGNOSTIC
    EXERCISER
    UTILITY
    VERIFIER
Program TYPE              [DIAGNOSTIC]>
SINGLE USER MODE required? (YES/NO) [NO] >
Program access SECURITY LEVEL (0..!) [ ! ] >
Does program always REQUIRE DEVICE (YES/NO) [NO] >
Is program AUTO-DIAGNOSTIC? (YES/NO) [NO] >
Does program DECODE device status? (YES/NO) [NO] >
Enter new DEVICES NOW SERVICED by this program:
    DEVICE NAME [<cr> = quit] >
    DEVICE NAME [<cr> = quit] >
    DEVICE NAME [<cr> = quit] >
    :
    :
    :

Need file containing executable PROGRAM code....
    FILE NAME      [<cr> = abort] >
Copying program file....
Need file containing program MESSAGES....
    FILE NAME      [<cr> = abort] >
Updating message catalog. (roughly ! min.)....
```

The following program module is about to be ADDED
to the Online Diagnostic Subsystem:

Name	Versn	Auto	Dcode	Securty	SUM	Funcn	Dev-Req'd	Devices
=====	=====	=====	=====	=====	=====	=====	=====	=====
xxxxxxxxx	x.xx.xx	xxx	xxx	x	xxx	xxxx	xxx	xxxxxxxxx
								xxxxxxxxx
								:
								:

Formatting message catalog (roughly ! min.)....
Updating main subsystem files....

Your last installation request completed successfully and is now finalized.
Generating working file(s)....

CORRECT

The correction function allows updates on any of the user-supplied data. Any or all of the following options may be selected in any order. Any subfunction may be selected more than once for correction of a correction (excepting, of course, the CANCEL option).

Example:

```
CORRECT AN EXISTING DIAGNOSTIC ...

Correction Subcommands:

CANCEL   - Countermand all pending corrections
CODE     - Replace program code
DEVICES  - Change diagnosed devices
DONE     - Done with corrections
DR       - Change device needed requirement
LIST     - Display program information
MESSAGES - Replace program messages
SECURITY - Change program access security level
SUM      - Change Single User Mode requirement
TYPE     - Change type of program

CORRECTION TASK >

CANCEL subcommand:

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

CODE subcommand:

    Need file containing executable PROGRAM code...
    FILE NAME      [<cr> = abort] >
    Copying program file...
    Program VERSION      (v.uu.ff) [?.?.??] >

DEVICES subcommand:

    Is program AUTO-DIAGNOSTIC?      (YES/NO) [ ! ] >
    Does program DECODE device status? (YES/NO) [ ! ] >
    Enter new DEVICES NOW SERVICED by this program:
    DEVICE NAME  [<cr> = quit] >
    DEVICE NAME  [<cr> = quit] >
```

```

:
Verifying remaining devices....
No devices are now currently listed for this program.

(or)

- device x
- device y
- :
- :

Enter DEVICES NO LONGER SERVICED by this program:
      DEVICE NAME [<cr> = quit] >
      DEVICE NAME [<cr> = quit] >
      :
      :

DONE subcommand:

The following CORRECTION is about to be made
to the Online Diagnostic Subsystem:

      Name   Versn  Auto Dcode  Securty  SUM  Functn  Dev-Req'd  Devices
=====
xxxxxxxxx x.xx.xx  xxx  xxx      x      xxx  xxxxx      xxx  xxxxxxxx
                                         xxxxxxxx
                                         :
                                         :

Formatting message catalog (roughly ! min.).... {if applicable}
Updating main subsystem files....

Your last installation request completed successfully and is now finalized.

DR subcommand:

Does program always REQUIRE DEVICE (YES/NO)[ ! ] >

LIST subcommand:

      Name   Versn  Auto Dcode  Securty  SUM  Functn  Dev-Req'd  Devices
=====
xxxxxxxxx xx.xx  xxx  xxx      x      xxx  xxxxx      xxx  xxxxxxxx
                                         xxxxxxxx

```

```
:
MESSAGES subcommand:

Need file containing program MESSAGES....
FILE NAME [<cr> = abort] >

Updating message catalog. Please be patient (roughly ! min.)....

SECURITY subcommand:

Program access SECURITY LEVEL      (0..!) [ ! ] >

SUM subcommand:

SINGLE USER MODE required?          (YES/NO) [ ! ] >

TYPE subcommand:

Program Type Options:
  DIAGNOSTIC
  EXERCISER
  UTILITY
  VERIFIER
Program TYPE                        [ ! ] >
```

EXIT

Selection of the EXIT command will cause the installer to terminate.

Output:

LEAVING ONLINE DIAGNOSTIC INSTALLER.... .

:



LIST

This function will display, on the terminal screen the user, correctable data for all diagnostic programs in the subsystem. This is slightly different from LIST in CORRECT mode, which lists only a single program module.

Output:

DIRECTORY CONTENTS:

Name	Versn	Auto	Dcode	Securty	SUM	Functn	Dev-Req'd	Devices
=====	=====	=====	=====	=====	=====	=====	=====	=====
xxxxxxx	x.xx.xx	xxx	xxx	x	xxx	xxxx	xxx	xxxxxxx
								xxxxxxx
								:
								:
yyyyyyyy	y.yy.yy	yyy	yyy	y	yyy	yyyy	yyy	yyyyyyyy
								yyyyyyyy
								:
								:
zzzzzzzz	...	...	...					
								:
								:
=====	=====	=====	=====	=====	=====	=====	=====	=====

REMOVE

This function will purge a module, including its directory entry, program file, and message set, from the diagnostic system. The user will be requested for verification just before the actual removal is done.

Output:

REMOVE AN EXISTING DIAGNOSTIC ...

Program NAME [<cr> = abort] >

The following program module is about to be REMOVED
from the Online Diagnostic Subsystem:

Name	Versn	Auto	Dcode	Securty	SUM	Functn	Dev-Req'd	Devices
=====	=====	=====	=====	=====	=====	=====	=====	=====
xxxxxxxx	x.xx.xx	xxx	xxx	x	xxx	xxxx	xxx	xxxxxxxx
								xxxxxxxx
								:
								:

OK to purge this program module? (YES/NO) [NO] >

Updating message catalog. Please be patient (roughly ! min.)....
Formatting message catalog (roughly ! min.)....
Updating main subsystem files....

Your last installation request completed successfully and is now finalized.

SHOWMSG

This function enables the user to verify the contents of the subsystem message catalog by displaying any message. The user is required to supply both a set number and the number of that message within that set. Since the installer automatically allocates set numbers to programs, a program name may be entered in place of the set number (this does not apply to the core system programs, i.e., message sets 1-6).

Output:

```
Program NAME/SET #      [<cr> = quit] >
MESSAGE #               [<cr> = quit] >

Message #! from Set #!:
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

SYSTEM

Selection of this function allows replacement of core message sets or restoration of the entire subsystem directory and/or message catalog while the system is up and running.

Note

The installer's message catalog CANNOT be replaced in this way.



Output:

Available Commands:

```
DIAGMON - Change Diagnostic Monitor messages
DUI      - Change Diagnostic User Interface msgs
SERVICES - Change System Services messages
NEWCAT   - Replace entire message catalog from backup
NEWDIR   - Replace entire directory from backup
EXIT     - Return to main menu
```

MESSAGE SET selection>

DIAGMON, DUI, SERVICES subcommands:

Need file containing program MESSAGES....

FILE NAME [<cr> = abort] >

Updating message catalog. Please be patient (roughly ! min.)....

Formatting message catalog (roughly ! min.)....

Updating main subsystem files....

Your last installation request completed successfully and is now finalized.

Generating working file(s)....

NEWCAT subcommand:

Need file containing replacement(backup) SUBSYSTEM MESSAGE CATALOG ...

FILE NAME [<cr> = abort] >

Updating message catalog. Please be patient (roughly ! min.)....

Formatting message catalog (roughly ! min.)....

Updating main subsystem files....

Your last installation request completed successfully and is now finalized.

NEWDIR subcommand:

Need file containing replacement(backup) SUBSYSTEM DIRECTORY ...

FILE NAME [<cr> = abort] >

Updating main subsystem files....

Your last installation request completed successfully and is now finalized.

XCHECK

This function will enable the user to cross-check the contents of the directory, subsystem catalog, and diagnostic account files against one another to search for inconsistencies and/or file corruptions as a means of determining the functional state of the diagnostic subsystem. It is intended to be used in conjunction with the LIST and SHOWMSG commands. Suggestions on the use of these commands as well as interpretation of XCHECK results and recommended recovery procedures will be provided in the appropriate help message(s).

Output:

The status table is built according to the following criteria:

ok	-No problems were detected.
dir	-The name index refers to a directory module whose name does not match this program.
corrupt	The directory must be considered corrupted.
msgset	-The message set used by this program is also used by one or more of the previously listed programs.
conflict	This is probably a directory corruption.
missing msgset	-The diagnostic message catalog does not contain the message set for this program.
missing program	-A file matching the program name was not found in the diagnostic group/account.
not shippable	-The program name was not found in the list of diagnostic subsystem files.

```
      :           :           STATUS
PROGRAM : MESSAGE :-----
NAME   : SET    :      dir   msgset  missing  missing  not
      :       :      ok    corrupt  conflict msgset   program  shippable
=====;=====;=====
xxxxxx  xx      X
yyyyyy  yy              X              X
zzzzzz  zz              X
etc.    .
      .
      .
```

The following files reside in the diagnostic subsystem account but are not reserved files and are not known to the diagnostic directory:

aaaaaaa	bbbbbbb	ccccccc	ddddddd
eeeeeee	ffffff	.	.
....			
....			

The checking done above has the following limitations:

1. Message sets are only guaranteed to exist, not to contain applicable messages. Use SHOWMSG to verify that the message set is the correct one.
2. Program files are verified by name only.

There are several additional errors/warnings which are displayed to pinpoint other problems with the subsystem.

Warning Messages

The following is a list of warning messages that may appear when using INSTALL. Other messages may appear at any time. All warning messages without the (INSTWARN #) trailer are generated by the Online Diagnostic Subsystem or MPE XL. The "!" indicates that a parameter of some sort replaces the exclamation point when the message is displayed. Listed below each message are a probable cause and recommended action statement. The following warnings are issued for potential problems or for residual problems after other errors.

05501	WARNING - The subsystem message catalog may be missing, invalid or corrupted. (INSTWARN 05501)
CAUSE	
ACTION	
05502	WARNING - List of directory empty space has probably been corrupted. This may result in directory corruption and/or inefficient directory utilization. Notify a Diagnostic Subsystem maintenance engineer. (INSTWARN 05502)
CAUSE	
ACTION	
05503	WARNING - The diagnostic subsystem is in essentially good condition, but the installer was not able to cause the diagnostic monitor to recognize either the message catalog or the directory. (INSTWARN 05503)
CAUSE	The installer has functioned correctly, but the remainder of the diagnostic subsystem may fail soon.
ACTION	
05504	WARNING - Your request has been cancelled. (INSTWARN 05504)
CAUSE	
ACTION	
05505	WARNING - This product is already listed for this diagnostic. Replacing previous reference.... (INSTWARN 05505)
CAUSE	Whenever the user attempts to add a duplicate product to the list of products diagnosable by the program at hand, the previous reference will be replaced by the one new one.
ACTION	

05506	WARNING - "!" is not currently diagnosed by this program. (INSTWARN 05506)
CAUSE	This message occurs when the user has attempted to remove a device from a program's device list when that device was not originally listed.
ACTION	
05507	WARNING - "!" will no longer have a designated AUTO-DIAGNOSTIC or DECODING PROGRAM. (INSTWARN 05507)
CAUSE	Issued when removing a diagnostic or revising the decoding or autodiagnostic capabilities of a diagnostic.
ACTION	
05508	WARNING - "!" will no longer have a designated AUTO-DIAGNOSTIC. (INSTWARN 05508)
CAUSE	Issued when removing a diagnostic or revising the autodiagnostic capabilities of a diagnostic.
ACTION	
05509	WARNING - No change made. (INSTWARN 05509)
CAUSE	
ACTION	
05510	WARNING - "!" will no longer have a designated DECODING PROGRAM (INSTWARN 05510)
CAUSE	Issued when removing a diagnostic or revising the decoding capabilities of a diagnostic.
ACTION	
05511	WARNING - Changes made since the last successful update could not be finalized. (INSTWARN 05511)
CAUSE	All changes in this session prior to and including the last "finalized" request are still good.
ACTION	
05512	WARNING - File could not be restricted to the system disk. Therefore, this file cannot be considered a shippable part of the diagnostic subsystem, even if the current installation task is successful on this computer system. (INSTWARN 05512)
CAUSE	
ACTION	

05513	WARNING - Changes were made according to your request to the subsystem catalog and directory. However, the program file for your request remains unchanged. (INSTWARN 05513)
CAUSE	Issued when the installer is not able to recover from a partial installation.
ACTION	
05514	WARNING - The Diagnostic Subsystem directory is missing ... (INSTWARN 05514)
CAUSE	
ACTION	Upon issuing this message, the installer will attempt to recover the directory if a previous installation attempt was the cause of its disappearance. Otherwise, the user will be given the opportunity to create a new one from scratch.
05515	WARNING - All other Diagnostic Subsystem users must be logged off for any installation to ultimately succeed. (INSTWARN 05515)
CAUSE	This warning could be printed if an autodiagnostic is running, even though the user has taken care to log off all other users.
ACTION	Terminate any currently running autodiagnostic programs and make certain that all users are logged off of the system.
05516	WARNING - No corrections were requested, therefore no changes were made. (INSTWARN 05516)
CAUSE	
ACTION	
05518	WARNING - The installer was unable to recover the original catalog. You now have new catalog but an old directory. After the above problem has been corrected, your system can be resynchronized by redoing your last installation task. (INSTWARN 05518)
CAUSE	Issued when the installer is not able to recover from a partial installation.
ACTION	Redo the last installation task.
05519	WARNING - This procedure causes a wholesale replacement of the diagnostic subsystem directory or message catalog. It is intended primarily for restoring these files from backup in the event of subsystem corruption. (INSTWARN 05519)
CAUSE	Issued upon any request to restore files from backup.
ACTION	

05520	WARNING - If the previous error is an attempt to read past the end file, the diagnostic directory is probably corrupted. (INSTWARN 05520)
CAUSE	Issued upon failure to obtain an expected record from the directory. Any error other than an EOF problem probably indicates a file system problem and not a diagnostic subsystem problem.
ACTION	Verify correct file system operation.
05521	WARNING - Message set ! is in the subsystem catalog, but unknown to the subsystem directory. (INSTWARN 05521)
CAUSE	Issued when no program file in the subsystem directory references a message set found in the subsystem catalog.
ACTION	Correct discrepancy between subsystem directory and catalog.
05522	WARNING - To insure correct operation of the diagnostic subsystem, ! should be purged. (INSTWARN 05522)
CAUSE	Issued when the installer has finalized changes in the directory and catalog, but is unable to update the program file.
ACTION	Purge the indicated program file.
05523	WARNING - To insure correct operation of the diagnostic subsystem, fix the source of the above problem and then reinstall the code for this program using the correct mode. NOTE - Copying the program file into the system. (INSTWARN 05523)
CAUSE	
ACTION	

Error Messages

The following is a list of error messages that may appear when using INSTALL. Other error messages may appear at any time. All error messages without the (INSTERR #) trailer are generated by the Online Diagnostic subsystem or MPE XL. The "!" indicates that a parameter of some sort replaces the exclamation point when the message is displayed. Listed below each error message are a probable cause and recommended action statement. There are several errors resulting from unsuccessful attempts to extract messages from a message catalog. Out of necessity, these are hardcoded into the program and for NLS reasons are included in this section even though they are not accessed through the catalog.

SYSTEM SERVICE "OUTPUT_ERROR" RETURNED STATUS OF xxx (INSTERR 05010)

INSTALLER COULD NOT CAUSE DISPLAY OF SYSTEM SERVICE ERROR MESSAGES.
(INSTERR 05051)

INSTALLER COULD NOT RETRIEVE A MESSAGE FROM ITS OWN CATALOG

Set # : xxx
Message # : yyy (INSTERR 05057)

INSTALLER UNABLE TO OPEN ITS OWN MESSAGE CATALOG. (INSTERR 05025)
The installer's message catalog is probably missing.

INSTALLER UNABLE TO OPEN THE DIAGNOSTIC SUBSYSTEM CATALOG. (INSTERR 05026)
The installer's message catalog is probably missing.

INSTALLER ABORTING DUE TO FATAL ERROR.... (INSTERR 05008)

Other errors are as follows:

05001	USER HAS INSUFFICIENT SECURITY CLEARANCE TO RUN INSTALLER. (INSTERR 05001)
CAUSE	The ability to run the diagnostic installer requires system manager level (0) security.
ACTION	User must obtain the specified security clearance.
05002	THE NEW MESSAGE SET HAS EXTRA OR MISPLACED SET IDENTIFIERS. (INSTERR 05002)
CAUSE	This condition occurs when the installer finds a set declaration error in the users message set file. The installer allows only one message set per program.
ACTION	The user should verify that only one (or none) \$SET declaration is made, that any set declaration is placed before the first message in the file, and that no \$DELSET declarations are made.

05003	INSTALLER WAS UNABLE TO FIND OR CREATE MORE SPACE IN THE DIAGNOSTIC DIRECTORY. (INSTERR 05003)
CAUSE	Either the size limit for the directory has been reached or an internal program error has corrupted the list of available space.
ACTION	This will normally require correction by a Diagnostic Subsystem maintenance engineer.
05004	INSTALLER IS UNABLE TO RECOVER FROM ERROR. (INSTERR 05004)
CAUSE	
ACTION	
05005	INSTALLER COULD NOT COPY FILE ! TO FILE ! (INSTERR 05005)
CAUSE	
ACTION	
05006	THE CURRENT CAPACITY OF THE DIAGNOSTIC SUBSYSTEM HAS BEEN EXCEEDED. NOTIFY SUPPORT ENGINEER. (INSTERR 05006)
CAUSE	The size limit for the directory has been reached. This should normally occur only when the number of diagnostic programs or number of diagnosed devices exceeds 200.
ACTION	This probably requires correction by a Diagnostic Subsystem maintenance engineer.
05007	INSTALLER WAS UNABLE TO PURGE !. (INSTERR 05007)
CAUSE	
ACTION	
05008	INSTALLER ABORTING DUE TO FATAL ERROR... (INSTERR 05008)
CAUSE	A non-recoverable error as occurred. This message will often be accompanied by others indicating what must be fixed before attempting to re-run the installer.
ACTION	After all the other errors are fixed, re-run the installer.
05009	FILE "!" NOT FOUND. (INSTERR 05009)
CAUSE	This could result from an internal system error, but will more likely result from the user entering the name of a non-existent file. This, of course, may be due to misspelling or incomplete qualification of the file name.
ACTION	Verify correct data entry.
05010	SYSTEM SERVICE "OUTPUT_ERROR" RETURNED STATUS OF !. (INSTERR 05010)
CAUSE	
ACTION	
05011	INSTALLER IS UNABLE TO INITIALIZE ITSELF. (INSTERR 05011)
CAUSE	
ACTION	

05012	PERMISSION TO INSTALL YOUR MOST RECENT CHANGES HAS BEEN DENIED BY THE DIAGNOSTIC MONITOR. (INSTERR 05012)
CAUSE	This condition should most frequently result from other diagnostic system users being logged on or from the system running an autodiagnostic at the time the request is made. The user will be given the chance to try again.
ACTION	Terminate any currently running autodiagnostic programs and make certain that all users are logged off the system.
05013	INTERNAL INSTALLER PROGRAM ERROR OR INTERNAL DIAGNOSTIC SYSTEM ERROR. NOTIFY THE INSTALLER'S MAINTENANCE ENGINEER. (INSTERR 05013)
CAUSE	
ACTION	This requires correction by a Diagnostic Subsystem maintenance engineer.
05014	INSTALLER RECEIVED EITHER AN INVALID IPC MESSAGE OR ONE INTENDED FOR ANOTHER PROCESS. (INSTERR 05014)
CAUSE	This is indicative of a failure in the interprocess communications routines and is not user fixable.
ACTION	This requires correction by a Diagnostic Subsystem maintenance engineer.
05015	INSTALLER COULD NOT RETRIEVE RECORD #! FROM THE DIAGNOSTIC DIRECTORY. (INSTERR 05015)
CAUSE	This is indicative of IO system failure or of a corrupted directory.
ACTION	If the latter, the directory may have to be restored.
05016	INSTALLER COULD NOT MODIFY ITS WORKING COPY OF THE DIAGNOSTIC DIRECTORY. (INSTERR 05016)
CAUSE	The installer was not able to write information into its working directory. This may be indicative of IO system failure.
ACTION	
05017	INSTALLER WAS UNABLE TO ERASE THE CONTENTS OF ! (INSTERR 05017)
CAUSE	
ACTION	
05018	INSTALLER COULD NOT LOCATE DIAGNOSTIC MONITOR'S IPC PORT. (INSTERR 05018)
CAUSE	This is indicative of either the diagnostic system monitor not running or of a failure in the interprocess communications routines.
ACTION	Start-up procedures for the diagnostic monitor should be consulted.



05019	INSTALLER FAILED TO RECEIVE A VALID MESSAGE FROM THE DIAGNOSTIC MONITOR. (INSTERR 05019)
CAUSE	The monitor may have been overloaded and therefore failed to return a message in time. This is not expected to occur if the installer is the only process logged onto the diagnostic system (which should of course be the case). It may also represent a failure in the interprocess communications routines.
ACTION	The user will usually be given the option to wait longer.
05020	INSTALLER COULD NOT ESTABLISH AN IPC PORT. (INSTERR 05020)
CAUSE	This is indicative of a failure in the interprocess communications routines and is not user fixable.
ACTION	This requires correction by a Diagnostic Subsystem maintenance engineer.
05021	INSTALLER WAS UNABLE TO MODIFY CONTENTS OF THE DIAGNOSTIC MESSAGE CATALOG. (INSTERR 05021)
CAUSE	This error will normally occur as the result of a malformed message set file. The installer will point out set declaration (\$SET) problems. Otherwise, it will display GENCAT errors (GCERR) returned to it. The installer does not know where in the file the error occurred.
ACTION	The Diagnostic Program Development Guide should be consulted for correct catalog formation.
05022	INSTALLER WAS UNABLE TO UPDATE THE DIAGNOSTIC DIRECTORY'S INDEX TO PROGRAM NAMES. (INSTERR 05022)
CAUSE	
ACTION	
05023	INSTALLER IS UNABLE TO SALVAGE THE SUBSYSTEM MESSAGE CATALOG. YOU MUST RESTORE IT FROM BACKUP. (INSTERR 05023)
CAUSE	
ACTION	
05024	A BRAND NEW DIAGNOSTIC DIRECTORY COULD NOT BE STARTED. (INSTERR 05024)
CAUSE	
ACTION	
05025	INSTALLER IS UNABLE TO ACCESS ITS OWN MESSAGE CATALOG. (INSTERR 05025)
CAUSE	The installer's message catalog is probably missing.
ACTION	

05042	INSTALLER IS UNABLE TO GET ESSENTIAL MESSAGE THROUGH TO SUBSYSTEM MONITOR. (INSTERR 05042)
CAUSE	This message results either when interprocess communication procedures fail or when the monitor is so busy it is unable to receive messages.
ACTION	If the latter case is possible, the user is offered the opportunity to retry, though this should not normally be necessary since the installer should be run after all other Diagnostic Subsystem users have logged off. In most cases, a Diagnostic Subsystem maintenance engineer should be consulted.
05043	EXPECTED ESSENTIAL MESSAGE FROM SYSTEM MONITOR BY NOW. (INSTERR 05043)
CAUSE	This should normally occur only when the system is very busy, something which should not normally happen when the installer is being run.
ACTION	The user will have the option to wait longer.
05044	INSTALLER'S MESSAGE PORT IS UNEXPECTEDLY OVERLOADED. UNABLE TO RECEIVE A MESSAGE. (INSTERR 05044)
CAUSE	This condition is never expected to arise.
ACTION	You must contact a Diagnostic Subsystem maintenance engineer.
05045	INSTALLER FOUND THE DIRECTORY'S LIST OF MESSAGE SET NUMBERS IN A CORRUPTED STATE. THIS IS NOT FATAL TO THE OPERATION OF THE DIAGNOSTIC SUBSYSTEM, BUT THE DIRECTORY SHOULD BE RESTORED FROM BACKUP PRIOR TO INSTALLING ANY MORE CHANGES TO THE SYSTEM. NOTIFY A DIAGNOSTIC SUBSYSTEM SUPPORT ENGINEER. (INSTERR 05045)
CAUSE	This condition is never expected to arise.
ACTION	You must contact a Diagnostic Subsystem maintenance engineer.
05046	THE NAME YOU HAVE CHOSEN IS RESERVED FOR EXCLUSIVE USE OF THE DIAGNOSTIC SUBSYSTEM. PLEASE SELECT ANOTHER. TYPE "HELP" FOR A LIST OF RESERVED NAMES. (INSTERR 05046)
CAUSE	
ACTION	
05047	PROGRAM COULD NOT BE RENAMED AS REQUESTED. (INSTERR 05047)
CAUSE	
ACTION	
05048	INSTALLER IS UNABLE TO GENERATE NEEDED WORKING FILES. (INSTERR 05048)
CAUSE	The installer sets up several working files, primarily safety copies of the diagnostic system directory and message catalog. Installation cannot take place without these files.
ACTION	This will normally require correction by a Diagnostic Subsystem maintenance engineer.

05050	INVALID NAME. TYPE "HELP" FOR FURTHER ASSISTANCE. (INSTERR 05050)
CAUSE	
ACTION	
05051	INSTALLER COULD NOT CAUSE DISPLAY OF DIAGNOSTIC SUBSYSTEM SERVICE ERROR MESSAGES. (INSTERR 05051)
CAUSE	
ACTION	
05052	INVALID RESPONSE. EXPECTED A COMMAND OR COMMAND QUALIFIER. CHECK SPELLING OR TYPE "HELP", "HELP COMMANDS", OR "REFRESH". (INSTERR 05052)
CAUSE	
ACTION	
05053	INSTALLER FOUND A DEVICE RELATED LIST(S) IN A CORRUPTED STATE. DIAGNOSTIC DIRECTORY MAY NEED TO BE RESTORED FROM BACKUP OR REBUILT FROM SCRATCH. NOTIFY A DIAGNOSTIC SUBSYSTEM SUPPORT ENGINEER. (INSTERR 05053)
CAUSE	
ACTION	
05054	INSTALLER COULD NOT RETRIEVE A MESSAGE FROM THE DIAGNOSTIC CATALOG (INSTERR 05054) SET #: ! MESSAGE #: !
CAUSE	
ACTION	
05055	INSTALLER COULD NOT RETRIEVE A MESSAGE FROM ITS OWN CATALOG (INSTERR 05055) SET #: ! MESSAGE #: !
CAUSE	
ACTION	
05056	CONTINUATION OF EXTRA MESSAGE SET LISTING NOT POSSIBLE OR IMPRACTICAL. (INSTERR 05056) The error(s) immediately previous to this one have negated the need to go on.
CAUSE	
ACTION	
05057	CONTINUATION OF CROSS CHECK NOT POSSIBLE OR IMPRACTICAL. (INSTERR 05057) The error(s) immediately previous to this one have negated the need to go on.
CAUSE	
ACTION	

05058	THE INSTALLER IS UNABLE TO VERIFY THE EXISTENCE OF MESSAGE SET !. (INSTERR 05058)
CAUSE	
ACTION	
05059	INSTALLER FOUND THE DIAGNOSTIC MESSAGE CATALOG IN A CORRUPTED STATE. (INSTERR 05059)
CAUSE	
ACTION	This will normally require restoring the catalog from scratch using the installer's SYSTEM command, NEWCAT option.
05060	INSTALLER WAS UNABLE TO OBTAIN A LIST OF FILES IN THE DIAGNOSTIC SUBSYSTEM ACCOUNT. (INSTERR 05060)
CAUSE	
ACTION	
05061	THERE IS ANOTHER DIAGNOSTIC INSTALLER ALREADY RUNNING. PERMISSION TO START THIS ONE IS THEREFORE DENIED. (INSTERR 05061)
CAUSE	For obvious reasons two users cannot be allowed to modify the same files at the same time.
ACTION	
05062	REQUESTED PROGRAM IS NOT INSTALLED IN THE DIAGNOSTIC SUBSYSTEM. (INSTERR 05062).
CAUSE	
ACTION	
05063	INSTALLER ATTEMPTED TO CHECK THE VALUE OF A JOB CONTROL WORD WHICH DOES NOT EXIST OR IS INVALID. THIS COULD IN TURN GENERATE FALSE ERRORS BELOW. NOTIFY A DIAGNOSTIC SUBSYSTEM SUPPORT ENGINEER. (INSTERR 05063)
CAUSE	
ACTION	
05064	NONSPECIFIC FILE SYSTEM ERROR RETURNED FROM GENCAT. (INSTERR 05064)
CAUSE	
ACTION	

05065	UNABLE TO EXPAND CATALOG ! INTO TEXT READABLE FORM. (INSTERR 05065)
CAUSE	
ACTION	
05066	GENCAT TERMINATED ABNORMALLY. GENCAT'S MESSAGE CATALOG, GCMGSCAT.PUB.SYS IS PROBABLY MISSING. (INSTERR 05066)
CAUSE	
ACTION	
05067	THE INSTALLER WAS ENVOCKED FROM AN ACCOUNT OTHER THAN "SYS" BUT A GROUP NAME WAS NOT SPECIFIED IN AN INFO STRING. (INSTERR 05067)
CAUSE	
ACTION	
05068	INSTALLER WAS UNABLE TO UPDATE THE LIST OF DIAGNOSTIC SUBSYSTEM FILES. (INTERR 05068)
CAUSE	
ACTION	
05069	INSTALLER WAS UNABLE TO CHECK THE LIST OF DIAGNOSTIC SUBSYSTEM FILES FOR THE POSITION AND OR PRESENCE OF A PROGRAM FILE. (INSTERR 05069)
CAUSE	
ACTION	
05070	ALTHOUGH THIS LAST INSTALLATION TASK WAS SUCCESSFUL FOR THIS SYSTEM, THE DIAGNOSTIC SUBSYSTEM MAY NOT NOW BE SHIPPABLE SINCE CHANGES TO THE LIST OF DIAGNOSTIC SUBSYSTEM FILES COULD NOT BE FINALIZED. (INSTERR 05070)
CAUSE	
ACTION	
05071	INVALID FILENAME FOUND IN THE LIST OF DIAGNOSTIC SUBSYSTEM FILES. (INSTERR 005071)
CAUSE	
ACTION	
05072	NAME LENGTH MAY NOT EXCEED ! CHARACTERS ON THIS SYSTEM. (INSTERR 05072)
CAUSE	
ACTION	
05099	THERE IS NO HELP AVAILABLE FOR THAT SPECIFIC TOPIC. FOR A LIST OF TOPICS, TYPE "HELP HELP". (INSTERR 05099)
CAUSE	Issued when an unrecognized "HELP" option is specified.
ACTION	Type "HELP HELP".



HP Part Number
09740-90021
Edition 4, U0690

Printed in U.S.A. June 1990