Concurrent Computer Corporation
Diagnostic Reference Manual

0855007-100
April 2005

The information contained in this document is believed to be correct at the time of publication. It is subject to change without notice. Concurrent Computer Corporation makes no warranties, expressed or implied, concerning the information contained in this document.

To report an error or comment on a specific portion of the manual, photocopy the page in question and mark the correction or comment on the copy. Mail the copy (and any additional comments) to Concurrent Computer Corporation, 2881 Gateway Drive, Pompano Beach, FL 33069. Mark the envelope **"Attention: Publications Department**." This publication may not be reproduced for any other reason in any form without written permission of the publisher.

Printed in U. S. A.

Issue No. :	**Date:**	**Description:**
000	October 2001	Diagnostic Release 9.7
100	April 2005	Diagnostic Release 9.8

Preface

Scope of Manual

This manual is intended for users responsible for the maintenance of the Concurrent Computer Corporation NightHawk Series 6000 and Power Hawk 600, 700, and 900 Series computer systems. This manual describes the system-level diagnostic tests that run under PowerMAX OS™ as well as stand alone diagnostics used to test the NightHawk Series 6000 computer systems.

Structure of Manual

This guide is made up of the following sections:

- Chapter 1, “Introduction and Installation”, introduces the diagnostics and provides installation and patch update instructions.
- Chapter 2, “Standalone Diagnostics,” describes the standalone diagnostics for the NightHawk Series 6000 systems.
- Chapter 3, “System Level Tests,” describes the system-level diagnostics for the Power Hawk 600, 700 and 900 Series systems and the NightHawk Series 6000 systems.
- Appendix A, “PowerIO Test Configuration File,” provides a sample PowerIO test configuration file.
- Appendix B, “Diagnostic Programs,” lists the diagnostic programs contained in each product tape in the order in which they should be executed.
- Appendix C, “Diagnostic Products,” provides a list of all the diagnostic products and their model numbers.
- The Index contains an alphabetical reference to key terms and concepts and the pages where they occur in the text.

Syntax Notation

The following notation is used throughout this guide:

italic	Books, reference cards, and items that the user must specify appear in *italic* type. Special terms may also appear in *italic*.
`list bold`	User input appears in **`list bold`** type and must be entered exactly as shown. Names of directories, files, commands, options and man page references also appear in **`list bold`** type.

`list`	Operating system and program output such as prompts and messages and listings of files and programs appears in `list` type.
`[]`	Brackets enclose command options and arguments that are optional. You do not type the brackets if you choose to specify such option or arguments

Related Publications

Title	Pubs No.
PowerMAX OS Users Guide	0890428
PowerMAX OS System Administration, Volume 1	0890429
PowerMAX OS System Administration, Volume 2	0890430
Character User Interface Programming	0890424
PowerMAX OS Programming Guide	0890423
STREAMS Modules and Drivers	0890426
Audit Trail Administration	0890431
Power Hawk Series 600 Console Reference Manual	0830050
Power Hawk Series 700 Console Reference Manual	0830059
Power Hawk Series 900 Console Reference Manual	0830060
HN6800 Console Reference Manual	0830045
HN6800 Architecture Reference Manual	0830046

Contents

Chapter 3 System Level Tests

Screens

Illustrations

Tables

1
Introduction and Installation

1
Introduction and Installation

Overview of Diagnostics

This manual describes the diagnostic programs that test the NightHawk Series 6000 and Power Hawk Series 600, 700, and 900 computer systems.

This chapter describes the distribution media for the diagnostic products and procedures for installing the diagnostic programs on the system.

Chapter 2 documents the diagnostics that execute in a standalone environment. Information includes instructions for loading and executing the diagnostics and man pages for each diagnostic.

Chapter 3 provides instructions for executing the system-level tests that run under the operating system.

Diagnostic Product Distribution

A Diagnostic Products Tape is available for each NightHawk Series 6000 system and Power Hawk Series 600 system. A Diagnostic Products CD is available for each Power Hawk Series system. A list of all the diagnostic products and their corresponding model numbers can be found in Appendix C. The tape or CD contains all the diagnostic programs necessary to test the system. A copy of all writable control store (wcs) files at the revision level supported by the diagnostics is also provided.

The diagnostic product tapes for NightHawk Series 6000 contain standalone diagnostics and system-level tests. The diagnostic product tapes and CDs for Power Hawk Series 600, 700, and 900 contain only system-level tests.

Standalone diagnostics provided on tape can be read by the console or loaded onto the system disk under the operating system. Diagnostics provided on the CD can be loaded onto the system disk under the operating system. The instructions for installing the diagnostics from tape and CD are given in the next section.

Installing Diagnostic Programs on Disk

A UNIX® Bourne shell script called **load_diags** facilitates the transfer of diagnostic products to the system disk. On the tape, the script occupies the first 8,192 bytes (eight 1KB records). On the CD, the script is the first 8,192 bytes (eight 1KB records) of the **dd** image contained on the CD.

The **load_diags** script displays a menu of diagnostic products to be loaded. For each product, the directory where the product will be loaded is shown (**/usr/d** or **/var/d**) and a warning message is displayed if that directory already exists. After the operator selects the products to be loaded, **load_diags** copies the products from the tape or **dd** image to the disk and then returns to the selection menu.

Installing Diagnostic Products from Tape

The following is the recommended procedure for installing diagnostic products onto the system disk from tape:

1. Login to the system as root.
2. Ensure that **/usr** and **/var** are accessible (mounted).
3. Load the diagnostic products tape on tape device 0 and put the tape drive online.
4. Enter the following commands:

```
# cd /
# dd if=/dev/rmt/0m of=load_diags count=8 bs=1k
# sh load_diags
```

   To load diagnostics from a tape unit other than **/dev/rmt/0m**, use the **-f** option as follows:

```
sh load_diags -f /dev/rmt/Xm
```

   The following menu will be displayed:

```
1) Load All Diagnostic Products
2) Exit
3) Load Standalone Diagnostic Products in /var/d
4) Load System-level Diagnostic Products in /usr/d/system
```

5. Enter menu selection(s) and <**Return**>.
6. Select the desired products. All selections must be entered on one line and a hyphen may be used to include a range of products. For example,

   1 <**Return**>
   3-4 <**Return**>

Installing Diagnostic Products from CD

The following is the recommended procedure for installing diagnostic products onto the system disk from the CD.

1. Login to the system as root.
2. Ensure that **/usr** and **/var** are accessible (mounted).
3. Load the diagnostics products CD onto the CD reader device.

In the following steps, <***mountpoint***> is a user-supplied mount point, and <***dd_image***> is **tape_img**{**5.1**|**6.1**|**6.2**}**_SYN**; for example **tape_img6.2_SYN**.

4. Select a mountpoint and mount the CD as follows:

```
# mkdir /<mountpoint>
# mount -F cdfs -r /dev/cd/0 /<mountpoint>
```

5. Enter the following commands:

```
# cd /
# dd if=/<mountpoint>/<dd_image> of=load_diags count=8 bs=1k
# sh load_diags -f /<mountpoint>/<dd_image>
```

The following menu will be displayed:

```
1) Load All Diagnostic Products
2) Exit
3) Load Standalone Diagnostic Products in /var/d
4) Load System-level Diagnostic Products in /usr/d/system
```

6. Enter menu selection(s) and <**Return**>.

7. Select the desired products. All selections must be entered on one line and a hyphen may be used to include a range of products. For example,

   `1` **<Return>**
   `3-4` **<Return>**

Patching Diagnostic Programs

If a diagnostic program requires a patch, the **`dpatch`** script is used to modify the program object on disk. This script should only be used as directed in Software Bulletins. The script will be installed in the same directory as standalone diagnostic products (either **`/var/d`** or **`/usr/d`**; see "Loading and Executing Standalone Diagnostic Programs" in Chapter 2). **`dpatch`** must be executed by the super-user under the operating system as follows:

`./dpatch` [**`-v`**] [**`-s`**] *program patch_file*

`dpatch` modifies the specified *program* according to the entries in the *patch_file*. The **`-v`** option causes verbose output of the patch process. For standalone programs, the *patch_file* contains a series of console 'write' commands to be applied to the memory image of the *program*. For online, system-level or console programs, the *patch_file* contains symbolic **`adb`** commands to be applied to *program* and the **`-s`** option must be used.

The original *program* image is saved in '*program*.before.*patch_file*.<time>' before the patch is applied. After the *program* is patched, the checksum change due to the installation of the patch is output for verification. A history of patches is maintained in the file **`patches`** under the standalone directory (either **`/var/d`** or **`/usr/d`**).

Patches to diagnostic programs are specified in Software Bulletins that are issued when an error has been found in a diagnostic program. The patch usually consists of a series of console commands that modify memory locations in the program. For example:

```
ww 1a05c 4e71 4e71 4e71
w  27024 1 2 4 8
```

These are the commands that are entered into *patch_file*. When running the diagnostic from tape, the patch can be entered manually or *patch_file* can be read by the console using the 'fs' command after *program* has been loaded into memory. If *program* is available on disk, the **`dpatch`** program can be used to permanently correct the disk image.

For example, to install the patch described in Software Bulletin #999 to the 'interact' diagnostic:

1. Login as super-user and change directory to the standalone directory (**`/var/d`** or **`/usr/d`**).

2. Create the file **`dib999`** containing the console 'w' commands.

3. Run **dpatch**:

```
./dpatch interact dib999
```

4. Verify that the checksum change printed by **dpatch** matches the value given in the Software Bulletin.

Note that the original version of the ‘interact’ diagnostic was copied to **interact.before.dib999.<time>** and a record of the patch was made in the *patches* file.

2
Standalone Diagnostics

2
Standalone Diagnostics

General

This chapter describes how to install and run the standalone diagnostic programs. Standalone diagnostics are only available on NightHawk Series 6000 computer systems.

Standalone diagnostics provide low-level functional tests of hardware subsystems in an offline environment. All standalone diagnostic programs include a copy of the Standalone Library which provides common system initialization, user interfaces, and error processing to the standalone CPU, memory subsystem, and I/O diagnostics.

Loading and Executing Standalone Diagnostic Programs

The standalone diagnostic programs can be run automatically using the "saint" program or run manually from the console. To properly isolate faults, the diagnostic suite must be run in a prescribed order (see Appendix B).

Before loading and executing a standalone diagnostic program, the operator must know where the standalone programs have been loaded on the disk. Standalone diagnostics will be loaded under **`/var/d`** unless **`/var/d`** is a symbolic link, in which case they will be loaded under **`/usr/d`**. The **`load_diags`** script displays the directory chosen for standalone diagnostics when the diagnostics are installed. The console 'fc' command may be used to list the directory contents of either **`/usr/d`** or **`/var/d`** , but it will not follow symbolic links.

NOTES

Standalone diagnostics can only be run if they have been loaded into a UFS file system type or from the Diagnostic Products Tape itself.

Some standalone diagnostic programs are dependent on the revision level of downloadable microcode files (e.g. **`/hsadisk`** depends on **`/hsawcs`** microcode). When an incompatibility between a released microcode file and diagnostic program arises, a Software Bulletin is issued to explain the problem and document a workaround. In the case of a disk controller microcode revision mismatch, the following procedure is used to load the correct version of the microcode before running the diagnostic.

The following is the procedure for loading a diagnostic copy of disk controller microcode. Note that this procedure only needs to be used when the latest disk controller microcode under the root directory is incompatible with the diagnostic program.

```
#>fl /var/d/hsawcs
#>v -l 400
device? dsk(x)                  x is the disk controller number
device? dsk(y)
(repeat for each disk controller)
device?.                        enter a dot to exit
#>
```

The revision level of the disk microcode that is compatible with the diagnostic is now loaded. After running the diagnostic, type **<Return>~b** before booting the operating system to cause the latest microcode to be reloaded.

Running Standalones Using the Standalone Interface (saint)

The **S**tand**A**lone **INT**erface (saint) is a tool that runs all necessary standalone diagnostics for a system automatically. Refer to the section “The Standalone Interface (saint)” on page 2-5 for details.

Running Standalones Using the Console

Standalone diagnostic programs can be loaded into CPU main memory by the console using one of the ‘f’ (file operation) commands. The ‘fr’ command loads the program into main memory and begins execution from there. The ‘fl’ command only loads the program into main memory. The ‘r’ command must then be entered to start execution. The file name must be supplied to the load command, including a device specification if the default device is not being used. The program may be loaded from system disk or from the diagnostic products tape.

The path name of diagnostic products on the diagnostic products tape is:

diag_load_name

The path name of diagnostic products on disk is one of the following:

/var/d/*diag_load_name*
/usr/d/*diag_load_name* (only if **/var** is a symbolic link)

NOTE

If the auto file mount feature is disabled (o+m), diagnostics on disk may be loaded using the following load path:

dsk(*controller,unit,partition,backplane***)/d/***diag_load_name*

where:

controller	is the disk controller number
unit	is the drive number on the disk controller which contains the diagnostics.
partition	is the partition of the drive where the diagnostics reside
backplane	is the backplane where the controller resides (0 = primary, 1 = secondary)

It is much quicker to load a diagnostic from the system disk. However, if the programs are not on the system disk or the disk is inaccessible, the tape serves as the alternate load medium.

Below are two examples of manually loading and executing standalone diagnostic programs.

Example 1 - Load and Execute from the System Disk:

#>**h.**	Halt the CPU.
#>**y.**	Initialize the CPU.
#>**fd dsk(*x*,*y*)**	Set the default device to the disk, where ***x*** is the disk controller number and ***y*** is the disk unit number.
#>**v.**	Reset and Initialize all boards.
#>**fr /var/d/memory**	Load and execute the ‘memory’ diagnostic.

Example 2 - Load and Execute from the Diagnostic Products Tape:

#>**h.**	Halt the CPU.
#>**y.**	Initialize the CPU.
#>**o+m**	Disable auto file mount option.
#>**fd mt(*x*,*y*)**	Set the default device to the tape drive, where ***x*** is the tape controller number and ***y*** is the tape unit number).
#>**fl /memory**	Load the ‘memory’ diagnostic from tape.
#>**r.**	Begin execution of the diagnostic.

Load names and the suggested order of execution of diagnostic programs are given in Appendix B.

Standalone Program Options

Standalone diagnostics have optional parameters which may be passed with the run (r) command or the file load and run command (fr). These parameters are interpreted by the programs and control such items as test selection, run options and pass count. The parameters for library-based programs are described in “Program Initiation Options” later in this chapter.

Standalone Menu Processing

The Standalone Interface (saint) and the Standalone Library share a common menu processing scheme, which are described in this section. The meanings of specific menu options are given in later sections.

As shown in Screen 2-1, a menu consists of three columns of keyword/value pairs.

```
                    NH6408 System Interaction Test Rev: 9.8
=====================  =====================  ==========================
    Program Options        Standard Options       Section/Test Selection
=====================  =====================  ==========================
minutes             10  passes                1   sect  1
multi               on  nohalt              off   s1    1
summary           0x07  silent              off
test_menu               r<un>
io_menu                 d<efault>
                        h<elp>
                        l<ist>
                        q<uit>

$>
```

Screen 2-1. Sample Standalone Library Options Menu

An entry of the form:

keyword value

will change the option *keyword* to *value*. If no *value* is specified, the option will assume its default value. The various types of values are shown in the following table. Note that numeric options may be specified in decimal or hexadecimal, but numeric ranges can only be specified in decimal. Hexadecimal entries must be preceded by the characters ‘0x’.

Type	Example
Boolean on/off	off
Boolean yes/no	no
Boolean true/false	true
Numeric hexadecimal	0x1ffff
Numeric decimal	10
Numeric range	1-3 5 7

Some keywords do not have values. For these keywords, an entry of the form

keyword

will perform the action specified by *keyword*.

When entering keywords, it is only necessary to enter enough characters of the keyword to make it unique. For example, in Screen 2-1, an entry of

p 5

is all that is necessary to change the ‘passes’ option to have value ‘45’, since it is the only keyword that begins with ‘p’. However, to change the ‘sect’ option, an entry of **se** *value* is required since other keywords begin with an ‘s’.

Some keywords are required to be entered in short form. These keywords are displayed with embedded ‘< >’ characters. In these cases, the keyword is only accepted if the letter(s) preceding the ‘<‘ are entered. The letters within the ‘< >’ characters are displayed to document the full name of the option. For example, in Screen 2-1 the h<elp> keyword will only be accepted when entered as an ‘h’. Note that the ‘h’ is distinguished from the ‘high’ keyword, because h<elp> is required to be entered in short form.

Additional information on all keywords is available by using the ‘help’ command. Entering

keyword

will provide a line of text to further describe the meaning of *keyword*.

The menu may be re-displayed (e.g. to show changes that were entered by entering a null line; i.e. a carriage return with no input).

The Standalone Interface (saint)

The **S**tand**A**lone **INT**erface (saint) examines the system configuration and automatically determines which diagnostics should be executed. An options menu allows the operator to change the default behavior of diagnostic program execution. Saint records the pass/fail status of all programs as they complete and provides a summary status report when all programs have completed.

NOTE

Saint cannot be used to run programs that require patches to be entered into memory before the program is executed.

Loading and Running

Saint is a console program which is loaded and executed using the ‘f’ (file operation) commands. Saint is *not* a CPU-based standalone program, therefore, it doesn’t load at the default load address in system memory. Once saint is loaded, it is executed by using the console ‘r’ (run) command with the **-p** option.

The following are examples of loading and running saint.

Example 1 - Load and Execute saint from the System Disk:

#>**y.**	Initialize the system.
#>**fd dsk(*x*,*y*)**	Set the default device to the disk, where ***x*** is the disk controller and ***y*** is the disk unit number. Note that the disk is usually the default device.
#>**fr /var/d/saint**	Load and execute saint. If **/var** is a symbolic link, saint will be under the **/usr/d** directory.

Example 2 - Load and Execute from the Diagnostic Products Tape:

#>**y.**	Initialize the system
#>**o+m**	Disable auto file mount option.
#>**fd mt(*x*,*y*)**	Set the default device to the tape drive, where ***x*** is the tape controller and ***y*** is the tape unit number. Note that the disk is usually the default device.
#>**fr /saint**	Load the saint program from tape.
#>**r -p**	Begin execution of saint.

Once saint has begun execution, it will initialize the system and display the options menu. Note that saint uses the system configuration that has been established by the operator using the console. Any operator changes to the configuration using the ‘v’, ‘y’, ‘td’, or ‘tu’ commands before saint is executed will be in effect throughout saint’s execution.

WARNING

Saint is loaded into system memory by the console. This reduces the number of physical memory pages available to the operating system. To remove saint from system memory, use the console command ‘v2’ before booting the operating system.

If the interact program is to be run, a scratch tape should be mounted on the default tape drive (i.e. drive 0 of first tape controller). Scratch tapes should be mounted in all tape drives that will be tested by tape controller diagnostics. If saint is being run from the default tape drive, saint will avoid testing the drive if the tape is write-protected.

Options Menu Processing

Once saint has begun execution, an options menu is displayed followed by the saint prompt (`saint>`). You can then modify the Commands/Options, CPU/Memory Diagnostics or I/O Diagnostics and Board Select. As shown in Screen 2-2, the menu consists of three columns of keyword/value pairs which are manipulated as described under “Standalone Menu Processing” on page 2-3.

```
                                SAINT Rev: 9.8
======================  ======================  ==========================
   Commands/Options        CPU/Memory Diags       I/O Diags & Board Select
======================  ======================  ==========================
passes                1  local               on   hsadisk    p6
dpasses               1  global              on   hsatape    p6
cpus                0-1  cache               on   ise        p0
iobuses             0-1  interrupt           on   interact   on
cpudiags                 timers              on
iodiags                  bench               on
fullrun             off
silent              off
nohalt              off
opt
dir
r<un>
d<efault>
h<elp>
l<ist>
s<tatus>
q<uit>

saint>
```

Screen 2-2. Sample saint Options Menu

Commands/Options

The commands and options available for saint are described in Table 2-1.

Table 2-1. saint Commands/Options

Option	Description
passes	The number of times to execute the selected group of diagnostics. An entry of '-1' will cause the group of diagnostics to be cycled forever. The group of diagnostics selected by saint is repeated for the specified number of passes.
dpasses	The number of passes that each individual diagnostic will perform every time it is run.
cpus	The range of CPUs on which the diagnostics should be executed. Note that diagnostics that execute multiprocessing tests are not restricted by this entry. Also note that the first selected processor runs the entire diagnostic, other processors run only those tests which do not duplicate testing of a global resource (e.g. global memory RAM is only tested by the first CPU). This restriction can be overridden by turning the 'fullrun' option on. I/O diagnostics are only run on the first selected processor.
iobuses	The I/O buses that are to be tested. The primary I/O bus is designated as '0' and the secondary I/O bus is designated as '1'.

Table 2-1. saint Commands/Options (Cont.)

Option	Description
cpudiags	The cpudiags command provides a convenient way to turn the entire set of CPU/Memory Diags on or off with a single command. The menu display is updated when this command is entered. Diagnostics can still be turned on or off individually.
iodiags	The iodiags command provides a convenient way to turn the entire set of I/O Diags on or off with a single command. The menu display is updated when this command is entered. Diagnostics can still be turned on or off individually.
fullrun	Fullrun on will override the saint default condition that prevents duplicate testing of global resources (e.g. global memory RAM) when the same diagnostic is run on multiple CPUs. Fullrun off prevents the duplicate testing.
silent	If on, section/test descriptions and program status messages will not be printed during program execution. Saint runtime output will also be inhibited.
nohalt	If off, saint will allow the operator to enter an Error Option when an error occurs. If on, saint will exit a failing program at the first error, then continue running the rest of the selected diagnostics.
opt	Sets/displays/clears options to be passed to the diagnostics. Refer to the following section in this chapter for details on the use of this command. Options for the individual diagnostics are described in the man pages at the end of this chapter.
dir	Sets/displays the directory where saint will search for diagnostic programs.
r<un>	Causes saint to begin executing the selected diagnostics.
d<efault>	Resets all options to their default values.
h<elp>	Provides a more detailed description of an option.
l<ist>	Lists the diagnostic program descriptions.
s<tatus>	Displays completion status report of the most recent saint run.
q<uit>	Halts saint and returns to console mode.

Using the ‘opt’ Command

The opt command provides a means of setting and displaying options that are passed to the individual diagnostics. To display all the options, enter:

`opt`

To display the options that will be passed to a diagnostic when it is run on a specific unit or units of hardware, enter:

`opt` *diagname unitslist*

If no options have been set up, these commands will not generate any output.

The command syntax used for setting up the options is identical to the command syntax used for displaying the options, except that the previous examples would be followed by a colon (:) and a list of options separated by colons.

For example:

```
opt hsadisk: drv0 off: drv1 on
opt local: section 1
```

In these two examples, the specified options will be set up for *all* runs of the named diagnostic, because *unitslists* were not provided.

In the following examples, *unitslists* are provided indicating that these options are only to be passed to the named diagnostics when the diagnostic is run on that unit of hardware.

```
opt hsadisk p2: drv0 off: drv1 on
opt local 0-2: section 1
```

In the hsadisk diagnostic example, options are passed to the diagnostic only when it is run on the HSA in slot 2 on the primary I/O bus (p2). In the local diagnostic example, the specified options are passed to the diagnostic only when it is run on cpus 0-2:

The following example clears all options pertaining to runs of the local memory diagnostic:

```
opt local:
```

The last example shows how to clear *all* options for *all* diagnostics:

```
opt:
```

CPU/Memory Diagnostics

This column of the saint options menu (Screen 2-2) lists the applicable CPU/Memory Diagnostics to be run.

Each diagnostic is described by a man page at the end of this chapter.

Note that diagnostic names ‘local’ and ‘global’ refer to executions of the memory diagnostic with type “local” and “global” respectively.

I/O Diagnostic and Board Select

This column of the saint options menu (Screen 2-2) lists the applicable I/O Diagnostics to be run along with their associated devices. Each diagnostic is described by a man page at the end of this chapter.

Instead of displaying ‘on’ when enabled (with the exception of interact), the I/O diagnostics display a list of the devices pertaining to that diagnostic. A device is identified by a letter for the I/O bus and a number for the slot/controller number. The letter ‘p’ stands for primary and the letter ‘s’ stands for secondary; therefore, the value ‘s2’ stands for “secondary I/O bus, slot/controller #2”.

Valid arguments to the commands are ‘on’, ‘off’ or a list of devices.

Runtime Output and Runtime Termination

During a program's execution, the normal output (e.g. section/test descriptions) is displayed. Also, each time saint loads or executes a program, a status message is displayed. These messages are inhibited if the silent option is 'on'.

To terminate a standalone diagnostic while it is running, enter **<Return><Return>**. If a diagnostic is aborted while running, saint considers it to have failed. Note that this will not work when saint is between runs of diagnostics.

Error Processing

If an error occurs while running a diagnostic, the error screen is displayed. If the nohalt option is 'off', the operator will be queried for an Error Option. If the operator decides to quit the diagnostic at this point it is considered a "nonstandard exit" and the operator is queried by saint. Refer to the following section entitled "Nonstandard Exits from Diagnostics" for complete details. If the nohalt option is 'on', the diagnostic will exit and saint will continue by executing the next diagnostic.

If multiple passes are being run, the program that failed will not be run on the failing CPU or I/O board in subsequent passes.

Nonstandard Exits from Diagnostics

A nonstandard exit from a diagnostic can be caused by the following:

- quitting the diagnostic through the Error Options
- runtime termination of the diagnostic
- a catastrophic condition (e.g. parity error)

If one of these conditions occurs and the nohalt option is 'off', you are prompted to select one of the three options listed in the following display:

```
*** nonstandard exit from diagnostic ***
enter:    'c' to continue with next diagnostic
          'm' to go to saint menu
          'x' to exit saint
>
```

Completion Status Report

At the end of each pass of saint, a completion status report is displayed which lists each of the selected diagnostics and its completion status on each CPU or I/O device. A completion status of 'passed' indicates that the diagnostic executed without error; 'FAILED' means that the program was abnormally terminated (e.g. a backplane parity error or other catastrophic condition). Program detected failures are reported by showing the section/test which detected the error.

For example, Screen 2-3 shows a pass of saint in a dual processor configuration. CPU 0 had a catastrophic error during the interrupt diagnostic, failed section 1 test 2 of the timers diagnostic and ran all other selected diagnostics without error. CPU 1 had no errors in the selected diagnostics. Also, the hsadisk diagnostic had a catastrophic error when it was run on the HSA in slot 6 of the primary I/O bus.

```
*** Pass 1 of SAINT Complete ***
diagnostic   | cpu 0     1
=============+========+=======+=======+=======+=======+=======+=======
local        | passed passed
global       | passed passed
cache        | passed passed
interrupt    | FAILED passed
timers       | 1/ 2   passed
=============+=============================================================
hsadisk      | p04:passed p06:FAILED
ise          | p00:passed
interact     | passed

enter <cr> to view SAINT menu...
```

Screen 2-3. Sample saint Completion Status Report

Standalone Library-Based Programs

Standalone library-based programs are loaded into CPU main memory by using one of the console ‘f’ (file operation) commands. Once a program is loaded into memory, it can be executed or re-executed by using the console ‘r’ (run) command. Refer to the section “Loading and Executing Standalone Diagnostic Programs” for more detail on loading and running diagnostic programs.

This section describes the library features which are common to all standalone diagnostics. Man pages for each standalone diagnostic program are appended to the end of this chapter in alphabetical order.

Program Initiation Options

The library interprets the value of the first parameter of the console ‘r’ or ‘fr’ commands to set certain run-time options. For example,

`fr /`*diag_name*`,`*lib_opts*

where *diag_name* is a standalone diagnostic load name (see Appendix B) and *lib_opts* is a combination of the bits given in Table 2-2. (Note that a space must precede the comma in the above command.)

Table 2-2. Standalone Library Initialization Options

Hex Option	Bit Number	Definition
00000001	B0	Print help information on *lib_opts* library options.
00010000	B16	Don't probe for I/O controllers and devices.
00020000	B17	Re-probe for I/O controllers and devices.
10000000	B28	Halt after loading the symbol table.
20000000	B29	Halt before enabling virtual memory.
40000000	B30	Halt after enabling virtual memory.
80000000	B31	Set saint mode – inhibits display of main menu and conditions error processing for use by saint.

Options Menu Processing

Once the program has begun execution, an options menu is displayed followed by the library prompt ($>). You can then modify the Program Options, Standard Options, or Section/Test Selection. As shown in Screen 2-4, the menu consists of three columns of keyword/value pairs which are manipulated as described in the section "Standalone Menu Processing" on page 2-3.

```
                    NH6408 Global Memory Diagnostic Rev: 9.8
======================  ======================  ==========================
    Program Options         Standard Options        Section/Test Selection
======================  ======================  ==========================
low          0x0003e6c4  passes                1   sect     1-3
high         0x00ff9fff  nohalt             off   s1       1-4
                         silent             off   s2       1-2
                         r<un>                    s3       1-3
                         d<efault>
                         h<elp>
                         l<ist>
                         q<uit>

$>
```

Screen 2-4. Sample Standalone Library Options Menu

Program Options

Program options are different for each diagnostic. See the MENU OPTIONS section of the program's man page at the end of this chapter for more information.

Standard Options

Standard options are the same for all programs that use the Standalone Library. The standard options are described in Table 2-3.

Table 2-3. Standalone Library Standard Options

Option	Description
passes	The number of times to execute the selected group of diagnostics. An entry of '-1' will cause diagnostic to execute forever.
nohalt	If 'off', causes the program to query for an Error Option when an error occurs. If 'on', causes the program to continue execution with the next test when an error occurs.
silent	If 'on', Section/Test descriptions and program status messages will not be printed during program execution.
r<un>	Causes the diagnostic to begin execution.
d<efault>	Resets all options to their default values.
h<elp>	Provides a more detailed description of an option.
l<ist>	Lists the Section/Test descriptions.
q<uit>	Halts the diagnostic and returns to console mode.

Section/Test Selection Options

Diagnostic programs are organized into *sections* of related *tests*. The number of sections and tests varies with each diagnostic. The Section/Test Selection options allow any group of tests within the diagnostic to be executed. The tests are selected as shown in Table 2-4.

Table 2-4. Standalone Library Section/Test Selection Options

Option	Description
sect	The range of sections to be executed
sn	The range of tests in section *n* to be executed. Note that section *n* must be enabled by the sect option.

Runtime Input/Output

During program execution, a description of each section and test is displayed before the test is executed. Other status information may also be displayed by the diagnostic. At the end of each pass of the program, a message indicating the pass count and number of errors in the pass is displayed. All of these outputs can be inhibited by the silent option.

While the program is running, the commands listed in Table 2-5 can be entered.

Table 2-5. Standalone Library Runtime Commands

Command	Action
m	Exit testing and return to the options menu.
q	Exit testing and return to console mode.

Note that these commands must be preceded and followed by a carriage return in order to be interpreted correctly. Also, the input is only read while output is occurring.

Error Processing

When a diagnostic program detects an error, a descriptive message is displayed along with a menu of options for continuation. The following screen depicts a typical error output.

```
NH6800 System Interaction Test Rev: 9.8
Error #1  Pass 1 CPU 0 Page 1
Section    1: Interaction Test
Test       1:

Pass 1 Elapsed Time = 0:49

*** Disk controller data compare error ***    offset: 40

Disk controller read from address: 0x20029308
Disk controller wrote to address:  0x2001a008
Expected data:  0x20029308
Received data:  0x20028308

Suspected fault(s):    1) CPU/MEM Slot 0: CPU 0 board
                       2) Controller # 5 bus 0: Disk Controller
                       3) CPU/MEM Slot 0: Local Memory address 0x2001a008


NONFATAL  1: Continue with test   4: Cycle on error    <cr> - Redisplay
page
ERROR     2: Go to next test      5: N/A
OPTIONS   3/m: Return to menu     6/q: Quit the diagnostic
err>
```

Screen 2-5. Sample Standalone Library Error Output

Error Messages

All error messages consist of three parts: a header, descriptive information, and a list of suspected faults.

Error Header

The first four lines of each error message presents the following information in a standardized format (see Screen 2-5).

- error number (incremented for each error occurrence)
- CPU number which detected the error current pass number
- error display page number
- Section, Test, and Case numbers and descriptions

Error Information

The descriptive error information which follows the error header varies depending on the program and the error. Typically, the information includes a short description of the test sequence followed by expected and received results. This information may exceed one screen; the next screen may be displayed by selecting the appropriate error option.

Suspected Fault(s)

At the end of each error message is a list of field-replaceable units that are suspected to have caused the problem. The list is presented in order so that the most likely suspect is on line 1, the second most likely suspect is on line 2, etc.

Error Options

There are two types of error option menus: FATAL and NONFATAL. The type is displayed in the lower left side of the error screen (see Screen 2-5). The error options are depicted in Table 2-6.

Table 2-6. Standalone Library Error Options

Entry	Name	Description
1	Continue with test*	Continue execution of the failing test after the point of error/
2	Go to next test	Continues execution with the next selected test.
3/m	Return to menu	Aborts testing and re-displays the options menu.
4	Cycle on error*	See the following sub-section.
5	Display event log**	Displays the unexpected event log.
6/q	Quit the diagnostic	Aborts testing and halts the system.
<cr>	Re-display page	Displays other page if message has two pages.
* N/A for FATAL errors ** N/A if no unexpected events have occurred		

Some error options have two designations. These entries are shown in the form *x*/*y* where either *x* or *y* may be entered.

Cycling on Error

The test sequence which generated a non-fatal error may be repeatedly executed in a tight loop by choosing error option 4: Cycle on error. When this option is chosen, the following prompt is output:

```
suppress error messages? (y/n)
cyc>
```

To eliminate the overhead of producing the screen output (e.g. during troubleshooting with an oscilloscope) or to determine how often an error is occurring, enter ‘y’. To view the error message each time an error occurs during the cycle loop, enter ‘n’. If ‘y’ is entered, the following prompt is output:

```
enter the number of cycles to perform between
status reports (0 = suppress status reports):
cyc>
```

The entry to this prompt determines how frequently the library will report the status of the cycle loop. The status report indicates how many times the loop had an error compared to the number of times through the loop. This feature allows a very tight ‘scope loop’ while keeping the operator informed as to whether the error is still occurring. If ‘0’ is entered, no status reports will be printed. Once this entry is made, the program will begin cycling on the failure.

If ‘n’ was entered to the suppress error messages question, the following prompt is displayed:

```
stop after error messages? (y/n)
cyc>
```

The response to this question determines whether the standalone library will stop to accept an error option input after subsequent errors. If ‘y’ is entered, each time an error is detected by the cycle loop, the error message will be output and a new error option will be requested. If ‘n’ is entered, an error message will be output each time an error is detected by the cycle loop but the program will continue to execute the cycle loop. After the response to this prompt is entered, the program will begin cycling.

Simultaneous Errors

In systems with multiple processors, it is possible for more than one processor to detect an error at the same time. When this occurs, one processor will perform normal error processing and request an error option. However, after the error option is entered, the processor will detect the simultaneous error and display the following message:

```
** catastrophic error condition: simultaneous errors **
valid inputs: 2, 3/m, 6/q
err>
```

One of the specified error options must then be entered to continue.

Standalone Diagnostic Program Man Pages

Each standalone diagnostic program is described in a man page provided at the end of this chapter. These are diagnostic man pages (denoted by section number **1D**) and are not available online.

Man Page Summary

The man pages that are provided in the following pages are listed in Table 2-7.

Table 2-7. Standalone Diagnostic Man Page Summary

Diagnostic	Description
bench(1D)	Standalone system performance benchmark
cache(1D)	Test primary and secondary caches
hsadisk(1D)	Test SCSI disks on HSA controller
hsatape(1D)	Test SCSI tapes on HSA controller
interact(1D)	Standalone system interaction test
interrupt(1D)	Test Series 6000 interrupt controller hardware
interrupt(1D)	Test Power MAXION interrupt controller hardware
ise(1D)	Test Integral SCSI/Ethernet/Graphics daughtercard and SCSI devices
memory(1D)	Test local or global memory
saint(1D)	Standalone Interface
timers(1D)	Test various system clocks and timers

How to Use Man Pages

The man pages in this section are independent entries that give specific information about individual standalone diagnostic programs. The name of the program is in the upper corner of each page along with the diagnostic manual section number **1D** in parentheses. The entries are presented in alphabetic order. Entries are based on a common format as follows:

The **NAME** specifies the name of the entry and briefly states its purpose.

The **SYNOPSIS** summarizes the program's use. The following conventions are used:

Boldface strings are literals and should be typed as they appear.
Square brackets around an argument indicate that the argument is optional.

The **DESCRIPTION** section discusses the features of the program.

The **PREREQUISITES** section indicates the steps that must be taken before the program is executed.

The **MENU OPTIONS** section describes the program-specific options that appear on the options menu.

The **SECTIONS/TESTS** section lists the titles of all Sections and Tests of the program.

The **NOTES** section gives any additional information that may be helpful when running the program.

The **SEE ALSO** section lists related man pages that can be consulted for further information.

This page intentionally left blank

NAME

bench - standalone system performance benchmark

SYNOPSIS

fr /bench , [lib_opts]; # from tape

fr /var/d/bench , [lib_opts]; # from disk

DESCRIPTION

Bench verifies that the system hardware is performing at expected levels. Processor, memory, and interrupt response time performance are checked.

PREREQUISITES

Memory, cache, interrupt, timers, ise, hsadisk and *hsatape* should be run before this program is executed.

MENU OPTIONS

procMHz	processor speed in MHz (100,150,200,etc.)
mem_size	memory test size in bytes

SECTIONS/TESTS

```
Section  1: Processor
   Test  1: PowerPC 604/604e
Section  2: Memory
   Test  1: local memory
   Test  2: global memory
Section  3: Interrupts
   Test  1: IPL read/write
   Test  2: Interrupt Response
```

SEE ALSO

saint(1D)

cache(1D) **Diagnostic Reference Manual** **cache(1D)**

NAME

cache - test primary and secondary caches

SYNOPSIS

fr /cache , [lib_opts];# from tape

fr /var/d/cache , [lib_opts];# from disk

DESCRIPTION

Cache tests the primary (L1) and secondary (L2) caches and associated cache coherency hardware for each specific processor. The program executes one pass in approximately 5 minutes. Execution time is less for systems without secondary cache and/or frontplane snoop filter logic.

PREREQUISITES

Memory should be run before this program is executed.

MENU OPTIONS

icache	enable primary instruction cache during data cache tests.
io_menu	menu of supported SCSI disk devices used for DMA testing.

SECTIONS/TESTS

```
Section  1: L1 Instruction Cache
   Test  1: read->XU->icbi->INV
   Test  2: read->XU->snoop hit icbi->INV
Section  2: L1 Data Cache
   Test  1: read->XU->kill->INV
   Test  2: rwitm->XM->kill->INV
   Test  3: rwitm->XM->clean->XU
   Test  4: rwitm->XM->flush->INV
   Test  5: dcbz->XM->clean->XU
Section  3: L2 Cache
   Test  1: read->XU->kill->INV
   Test  2: rwitm->XM->kill->INV
```

```
   Test  3: rwitm->XM->clean->XU
   Test  4: rwitm->XM->flush->INV
   Test  5: dcbz->XM->clean->XU
Section  4: Multiprocessing Cache Snooping
   Test  1: read->XU->snoop hit read->SU
   Test  2: rwitm->XM->snoop hit read->SU
   Test  3: read->XU->snoop hit rwitm->INV->read->SU
   Test  4: rwitm->XM->snoop hit rwitm->INV->read->SU
   Test  5: read->SU->snoop hit rwitm->INV->read->SU
   Test  6: read->SU->rwitm->XM->snoop hit read->SU
Section  5: DMA Cache Snooping
   Test  1: DMA Reads
   Test  2: DMA Writes
Section  6: Frontplane Snoop Filter
   Test  1: tag RAM
   Test  2: index and tag comparator
```

NOTES

The program tests unique cache hardware associated with each processor. Therefore, each processor must run the program.

Diagnostic cylinders are used on the disk, so user data is not destroyed except in cases of extreme malfunction.

SEE ALSO

saint(1D)

NAME

hsadisk - test SCSI disks on HSA controller

SYNOPSIS

fr /hsadisk , [lib_opts];# from tape

fr /var/d/hsadisk , [lib_opts];# from disk

fr /usr/d/hsadisk , [lib_opts];# from disk if /var is symbolic link

DESCRIPTION

Hsadisk verifies the complete SCSI disk functionality of the HSA. From zero to seven disk drives may be connected to the controller during test. The program uses the internal turnaround features of the controller as well as the diagnostic cylinders on the SCSI disk drives to verify data transfers. The program takes approximately 5 minutes to complete one pass.

PREREQUISITES

Memory, cache, and *interrupt* should be run before this program is executed.

MENU OPTIONS

slot	the HSA controller (HVME slot) number (2-21).
bp	the I/O backplane where the HSA is located (0 = primary, 1 = secondary). This option is only displayed if a secondary HVME backplane is available.
*drv*X	selection flag for drive X (on/off).
oit	selection flag for operator intervention tests (on/off).

SECTIONS/TESTS

```
Section  1 - HSA Registers Verification
  Test 1 - ID Return Address Register Test
  Test 2 - Hard Reset Register Test
  Test 3 - Soft Reset Register Test
  Test 4 - Master MCB Address Register Test
Section  2 - Master MCB Verification
```

```
  Test 1 - MCB Completion Indicators Test
  Test 2 - Immediate Execution MCL Test
  Test 3 - Master MCB Operations Test
Section  3 - Control Store Verification
  Test 1 - Write/Read/Execute Control Store Test
Section  4 - MCB Queue Verification
  Test 1 - Queued Execution MCL Test
  Test 2 - MCB Queue Processing Test
Section  5 - Controller Write/Read Turnaround Tests
  Test 1 - Write/Read Diagnostic Area Commands Test
  Test 2 - Write/Read Buffer Area Commands Test
  Test 3 - Write/Read Data Chaining Test
Section  6 - Controller Error Generation & Detection Verification
  Test 1 - Invalid Command Code Error Test
Section  7 - Drive Information Commands Verification
  Test 1 - Read Drive Status Command Test
Section 8 - Disk Write/Read Commands Verification
  Test 1 - Write/Read Data Block Commands Test
Section 9 - Cylinder/Head/Sector Spills Tests
  Test 1 - Sector Spill Test
  Test 2 - Head Spill Test
  Test 3 - Cylinder Spill Test
Section 10 - Disk Error Detecting & Reporting Verification
  Test 1 - Drives Online/Offline Detection Test
  Test 2 - Drives Write Protect Detection Test
```

NOTES

The disk(s) must be formatted before running this program (see the system administration manuals). Diagnostic cylinders are used on the disk, so user data is not destroyed except in cases of extreme malfunction. Sections 1 thru 6 of this diagnostic do not require that a disk drive be present in order to execute properly. Sections 7 thru 10 require that at least one disk drive be present. Also, Section 10, Tests 1 and 2 are only executed if the *oit* option is enabled. Section 10, Test 1 requires that the disk(s) be powered down (this may not be possible on some systems). Section 10, Test 2 requires that the disk(s) have a write-protect switch.

Do not attempt to test a floppy (removal media) disk device with this diagnostic. Test failures will occur!

SEE ALSO

saint(1D)

NAME

hsatape - test SCSI tapes on HSA controller

SYNOPSIS

fr /hsatape , [lib_opts];# from tape

fr /var/d/hsatape , [lib_opts];# from disk

fr /usr/d/hsatape , [lib_opts];# from disk if /var is symbolic link

DESCRIPTION

Hsatape verifies the complete SCSI tape functionality of the HSA. The program supports up to seven tape devices (SCSI IDs 0 through 6). Steps are taken to assure that other types of devices (e.g. printers and disks) on the SCSI bus are not accessed. When auto configuration is enabled, Section 7 Test 1 is automatically selected and the appropriate tests are selected.

SCSI tape devices require cartridge/tape media for each tape device tested and media must be write enabled (unprotected).

The following tape devices are supported:

- 6140 cartridge tape
- 6141 cartridge tape
- 6150 8 mm tape
- 6750 9-track tape
- Archive 2150S cartridge tape
- Wangtek 5150 ES cartridge tape

This program takes approximately 20 minutes to complete one pass when executed with one tape device.

PREREQUISITES

Memory, cache, and *interrupt* should be run before this program is executed.

MENU OPTIONS

slot	the HSA controller (HVME slot) number (2-21).
bp	the I/O backplane where the HSA is located (0 = primary, 1 = secondary). This option is only displayed if a secondary HVME backplane is available.

auto	selects auto configuration or manual tape selection (on = auto, off = manual). When manual tape selection is enabled, tape devices for testing are selected with the *idN* options.
idN	for manual device selection (on = selected, off = not selected), where *N* is the tape device ID number. A manually selected device must be a tape device. If the selected device is determined to be a non-supported device, it is automatically removed from the run-time test configuration. The *idN* options are ignored when auto configuration is enabled.
eot_tst	selects the end-of-tape test (yes = select, no = not selected). By default, the end-of-tape test is not selected.

SECTIONS/TESTS

```
Section  1 - HSA Registers
  Test 1 - ID Return Address Register
  Test 2 - Hard Register Reset
  Test 3 - Soft Register Reset
  Test 4 - Master MCB Address Register
Section  2 - Master MCB
  Test 1 - MCB Completion Indicators
  Test 2 - Immediate Execution MCL
  Test 3 - Master MCB Operations
Section  3 - Control Store
  Test 1 - Write/Read/Execute Control Store
Section  4 - MCB Queue
  Test 1 - Queued Execution MCL
  Test 2 - MCB Queued Processing
Section  5 - Write/Read Turnaround
  Test 1 - Write/Read Diagnostic Area Commands
  Test 2 - Write/Read Buffer Area Commands
  Test 3 - Write/Read Data Chaining
Section  6 - Controller Error Generation and Detection
  Test 1 - Invalid Command Code Error
Section  7 - Auto Configuration
  Test 1 - Test-Unit-Ready and Inquiry
Section  8 - Tape Attributes
  Test 1 - Request Sense
  Test 2 - Read Block Limits
  Test 3 - Mode Sense and Mode Select
Section  9 - Tape Non-data Motion Command
  Test 1 - Rewind
  Test 2 - Erase
  Test 3 - Write Filemarks
Section  10 - Tape File Position
  Test 1 - Space Files
```

```
  Test 2 - Backward Space File and Crash BOT
Section  11 - Tape Write/Read Data
  Test 1 - Basic Write
  Test 2 - Basic Read
Section 12 - Tape Block Positioning
  Test 1 - Space Blocks
  Test 2 - Space Forward Filemark
  Test 3 - Forward Space Over Filemark
  Test 4 - Space Sequential Filemarks
  Test 5 - Read Over Filemark
  Test 6 - Backward Space Block and Crash BOT
  Test 7 - Backward Space Block Over Filemark
  Test 8 - Backward Space Filemark Over Filemark
Section 13 - Tape Drive Status and Error Detection
  Test 1 - SCSI Illegal Request Status
  Test 2 - End-of-Media Status
  Test 3 - Illegal Write
  Test 4 - Illegal Read
  Test 5 - Data Overwrite
```

NOTES

Successful execution of Section 7 Test 1 is required before any tape device testing can begin. Not all tests are applicable to every device.

When loaded by *saint*, this program will exit without error if no applicable tape devices are found.

This program will not configure a tape device that is the *saint* load device.

SEE ALSO

saint(1D)

NAME

interact - standalone system interaction test

SYNOPSIS

fr /interact , [lib_opts];# from tape

fr /var/d/interact , [lib_opts];# from disk

fr /usr/d/interact , [lib_opts];# from disk if /var is symbolic link

DESCRIPTION

Interact loads the system with integer and floating point operations, DMA and program I/O transfers, context switches, real-time clock interrupts, remote port UART activity, and page faults on all available CPUs. All available memory is used for code migration, stacks, and DMA buffers. The operator can easily degrade the test to isolate any failing interactions. By default, the test runs for approximately 10 minutes.

PREREQUISITES

Memory, cache, interrupt, timers, ise, hsadisk and *hsatape* should be run before this program is executed.

MENU OPTIONS

minutes	select minimum execution time per pass (1-6000)
multi	enable multiprocessing (on/off).
snoop	enable snooping of DMA accesses (on/off).
summary	end of pass summary bitmask. Bit 0 enables printing 60Hz clock, context switch, page fault, and realtime clock interrupt counts for each CPU. Bit 1 enables printing of memory pool usage by I/O devices. Bit 2 enables printing of DMA operation counts for each I/O device. Any combination of bits (0x0 to 0x7) can be selected.
test_menu	display test options menu (see Test Menu below).
io_menu	display I/O device menu (see I/O Menu below). The default for *minutes* is *10*. The default for *multi* is *on*.

The default for *snoop* is *on*.
The default for *summary* is *0x0F.*

SECTIONS/TESTS

```
Section  1: Interaction Test
   Test  1:
```

TEST MENU

Selecting the *test_menu* option causes *interact* to print a list of tests that can be enabled or disabled. A sample list is shown below:

```
Number  State      Test Description
   1     on     floating point operations
   2     on     integer matrix multiply
   3     on     integer base conversion
   4     on     basic memory operations
   5     on     multiprocessing cache snooping
   6     on     object code migration
   7     on     semaphore operations
   8     on     realtime clock interrupts
   9     on     interprocessor interrupts
  10     on     softclock interrupts
  11     on     interrupt priority (IPL) checks
  12     on     interval timer reads
  13     on     non-volatile RAM write/read
  14     on     program I/O write/read
  15     on     load/store string operations
  16     off    remote port RS-232 turnaround
```

All tests shown above are enabled by default, except for test number 16 (remote port RS-232 turnaround), which requires an RS-232 turnaround connector.

Entering a test number will toggle the test state from *on* to *off* and vice versa. Multiple tests can be enabled/disabled by entering several test numbers on one line separated by spaces. A range of tests can be enabled/disabled by entering the first and last test numbers of the range separated by a hyphen (-). Individual tests and ranges of tests can be entered on the same line. Typing a carriage return will cause the menu to be re-printed with any changes that may have been made by the operator.

I/O MENU

Selecting the *io_menu* option causes *interact* to print a summary of I/O controllers and devices found during program startup. A maximum of fourteen (14) devices are printed.

On Power MAXION systems with more than four (4) CPU boards, the DEC ethernet devices are not configured in the *io_menu*.

Following the I/O device summary is a list of commands which the operator may use to select/deselect a particular device for testing, change the default drive number for a given device, or quit and return to the main menu. Typing a carriage return will cause the menu to be re-printed with any changes that may have been made by the operator.

NOTES

Selecting the error menu option '3/m: Return to menu' will cause the program to halt. The program must be restarted using the console commands 'y' and 'r'.

When run from *saint*, none of the *test_menu* or *io_menu* options can be changed by using the *opt* command in the *saint* menu. If a menu option must be changed from its default value, run *interact* outside of *saint*.

Remote port UART testing requires an RS-232 turnaround connector on the remote port for each CPU board. The remote port test will not be run if the Remote Enable frontpanel switch is set to the enabled position.

Disk operations only access reserved diagnostic cylinders; therefore, user data will not be destroyed except in cases of extreme malfunction.

Currently, *interact* supports

- HSA SCSI disk, VIA SCSI disk, Integral SCSI disk,
- Embedded PCI SCSI disk, Embedded PCI ethernet,
- Interphase Eagle 4702 ethernet,
- Interphase Condor 4221 ethernet,
- Interphase Peregrine II 5211 FDDI,
- and Systech HPS controllers for testing DMA transfers.

Ethernet and FDDI controllers are used for DMA purposes only, therefore no data is transmitted or received from the network, and no network connection is necessary for testing.

HPS controllers are used for DMA purposes only, therefore no data is transmitted or received from the RS-232 ports, and no RS-232 turnaround connector is necessary for testing.

Interphase Peregrine 4211 FDDI controllers are not supported.

SEE ALSO

saint(1D)

NAME

interrupt - test Series 6000 interrupt controller hardware

SYNOPSIS

fr /interrupt , [lib_opts];# from tape

fr /var/d/interrupt , [lib_opts];# from disk

DESCRIPTION

Interrupt tests the interrupt hardware on the CPU. The diagnostic verifies proper operation of the interrupt hardware in that the CPU can receive and acknowledge interrupts according to the priority of the interrupt and any masking of interrupts that may be in effect. The diagnostic will also test the interrupt hardware in a multi-processing environment. This program executes one pass in about 5 seconds.

PREREQUISITES

Memory and *cache* should be run before this program is executed.

MENU OPTIONS

cmdlog displays interrupt command log.

This option will print a summary of the commands issued to the interrupt controller prior to detection of any failure.

SECTIONS/TESTS

```
Section  1: Register Integrity
   Test  1: arm register
   Test  2: enable register
   Test  3: request register
   Test  4: IPL register
Section  2: Emulated Interrupt Acknowledge
   Test  1: single requests
   Test  2: multiple requests
Section  3: Interrupt Acknowledge By CPU
   Test  1: single requests
   Test  2: IPL comparator: request > IPL
   Test  3: IPL comparator: request <= IPL
   Test  4: multiple active interrupts
```

```
   Test  5: multiple pending requests
   Test  6: 60 Hz clock
Section  4: Miscellaneous
   Test  1: HXI/RXI
   Test  2: interrupt level write of IPL
   Test  3: set request/write ipl interaction
   Test  4: interrupt level
Section  5: Multiple Processors
   Test  1: read/write IPL
   Test  2: multiple CPU interrupts
   Test  3: lost interrupts
   Test  4: foreign CPU interrupts
   Test  5: local CPU interrupt combinations
   Test  6: interrupts with IPL write/read interaction
```

NOTES

Sections 1 through 4 of the diagnostic only test the interrupt hardware for the processor that is executing the program. Therefore, sections 1 through 4 must be run on each processor. Section 5 may be run on any or all processors present on a given CPU board, but must be run on at least one processor on each board.

SEE ALSO

saint(1D)

NAME

interrupt - test Power MAXION interrupt controller hardware

SYNOPSIS

fr /interrupt , [lib_opts];# from tape

fr /var/d/interrupt , [lib_opts];# from disk

DESCRIPTION

Interrupt tests the interrupt hardware on a Power MAXION CPU. The diagnostic verifies proper operation of the interrupt hardware in that the CPU can receive and acknowledge interrupts according to the priority of the interrupt and any masking of interrupts that may be in effect. The diagnostic will also test the interrupt hardware in a multi-processing environment. This program executes one pass in about 5 seconds.

PREREQUISITES

Memory and *cache* should be run before this program is executed.

MENU OPTIONS

cmdlog displays interrupt command log.

This option will print a summary of the commands issued to the interrupt controller prior to detection of any failure.

SECTIONS/TESTS

```
Section  1: Register Integrity
   Test  1: arm register
   Test  2: enable register
   Test  3: request register
   Test  4: IPL register
   Test  5: IACK register
Section  2: Interrupt Acknowledge
   Test  1: single requests
   Test  2: IPL comparator: request > IPL
   Test  3: IPL comparator: request <= IPL
```

```
   Test  4: multiple active interrupts
   Test  5: multiple pending requests
Section  3: Miscellaneous
   Test  1: set request/write ipl interaction
   Test  2: hardclock
   Test  3: hardclock (disarmed)
Section  4: Multiple Processors
   Test  1: request register
   Test  2: interrupt acknowledge
```

NOTES

All sections of the diagnostic only test the interrupt hardware for the processor that is executing the program. Therefore, all sections must be run on each processor.

SEE ALSO

saint(1D)

NAME

ise - test Integral SCSI/Ethernet/Graphics daughtercard and SCSI devices

SYNOPSIS

fr /ise , [lib_opts];# from tape

fr /var/d/ise , [lib_opts];# from disk

fr /usr/d/ise , [lib_opts];# from disk if /var is symbolic link

DESCRIPTION

Ise tests the interface and functionality of the Integral SCSI/Ethernet (ISE) daughtercard. The ISE does not exist on systems that incorporate PCI devices in their architecture, therefore the *ise* diagnostic will not exist on these systems. The ethernet cable/transceiver interface is tested if present. SCSI device tests are also performed, but SCSI devices need not be present for this program to execute successfully. Program execution time is approximately 4 minutes with extended SCSI device tests disabled. Extended tape testing can significantly increase execution time.

PREREQUISITES

Memory, *cache*, *interrupt*, and *timers* should be run before this program is executed.

MENU OPTIONS

prom	enables verification of the network address PROM.
netw	enables testing of the network cable/transceiver interface.
drives	select SCSI drives to test (-1 = deselect device testing)
extended	enable extended SCSI device tests (sections 7 and 9)
devlist	list devices found on SCSI bus

The "prom" and "netw" options default to "no" values. The "drives" option defaults to 0-6. The "extended" option defaults to "off".

When selected, the "prom" option requires the operator to input the 6-byte physical ethernet address which has been programmed into the network address PROM. The first 3 bytes of this address are fixed for all ISE daughtercards: 0x00, 0x00, 0xc3. The last three bytes are unique to each ISE daughtercard and are obtained from the bar-code UID on the daughtercard.

The "netw" option should be selected if the ISE daughtercard is cabled to an ethernet transceiver. The operator is warned that *all external network traffic must be suppressed for these tests to execute properly,* and *data will be transmitted onto the ethernet cable.*

The "drives" option allows the operator to select and deselect the SCSI devices that will be used for testing. By default, the program will probe for all SCSI disk and tape devices and run the appropriate tests for each device.

The "extended" option enables tests that verify special SCSI peripheral operations.

The "devlist" option prints a summary of the SCSI devices found.

SECTIONS/TESTS

```
Section  1: ISE Basic Access
   Test  1: Daughterboard Registers
   Test  2: Network Processor Registers
   Test  3: Network Processor Address PROM Verification
   Test  4: SCSI Processor Registers
Section  2: Network Processor Tests
   Test  1: Self-Test
   Test  2: Channel Attention With Interrupt
   Test  3: CU Initialization
   Test  4: Network Processor Commands
   Test  5: Transmit Command, Internal Loopback
   Test  6: 8023A Transceiver Test
   Test  7: Time Domain Reflectometry Test
   Test  8: Transmit Command, External Loopback
Section  3: SCSI Processor Tests
   Test  1: SCSI Interrupt
   Test  2: SCSI FIFO Test
   Test  3: SCSI Bus Arbitration/Selection
   Test  4: SCSI Data And Control Lines
   Test  5: DNAD Register Increment
   Test  6: DBC Register Decrement
   Test  7: DMA FIFO Test
   Test  8: SCSI I/O Processor
   Test  9: DMA Writes
   Test 10: DMA Reads
   Test 11: SCSI Parity Error Detection
Section  4: SCSI/Ethernet Simultaneous Operations
   Test  1: Interaction Test
Section  5: Graphics Processor
   Test  1: GSP Interrupt Test
   Test  2: GSP/Ethernet Interrupt Interaction Test
Section  6: Tape Basic Tests
   Test  1: Request Sense
   Test  2: Read Block Limits
```

```
   Test  3: Mode Sense and Mode Select
   Test  4: Rewind
   Test  5: Basic Write
   Test  6: Basic Read
Section  7: Tape Extended Tests
   Test  1: Erase
   Test  2: Write Filemarks
   Test  3: Space Files
   Test  4: Backward Space File and Crash BOT
   Test  5: Space Blocks
   Test  6: Space Forward Filemark
   Test  7: Forward Space Over Filemark
   Test  8: Space Sequential Filemarks
   Test  9: Read Over Filemark
   Test 10: Backward Space Block and Crash BOT
   Test 11: Backward Space Block Over Filemark
   Test 12: Backward Space Filemark Over Filemark
   Test 13: SCSI Illegal Request Status
   Test 14: Illegal Write
   Test 15: Illegal Read
   Test 16: Data Overwrite
Section  8: Disk Basic Tests
   Test  1: Request Sense
   Test  2: Mode Sense and Mode Select
   Test  3: Write Extended
   Test  4: Read Extended
   Test  5: Write/Read Multiple Blocks
Section  9: Disk Extended Tests
   Test  1: Rezero
   Test  2: Seek Extended
   Test  3: Sector Spills
   Test  4: Write Large Data Transfers
   Test  5: Read Large Data Transfers
   Test  6: Illegal Write
   Test  7: Illegal Read
```

NOTES

This diagnostic must be run on all available ISE daughtercards.

SEE ALSO

saint(1D)

NAME

memory - test local or global memory

SYNOPSIS

fr /memory , [lib_opts];# from tape

fr /var/d/memory , [lib_opts];# from disk

DESCRIPTION

Memory tests the local memory on a CPU board or the global memory in the system. Each time the program is started, it asks for the type of memory (1=Local, 2=Global, ?=Generic) to be tested before displaying the options menu.

The program performs RAM and addressing tests on the memory array and forces errors and verifies that the CPU reports them correctly. The program executes for about one minute for each 16Mb of local or global memory.

PREREQUISITES

None.

MENU OPTIONS

low	the low address of the memory to be tested.
high	the high address of the memory to be tested.
io_menu	menu of supported SCSI disk devices used for DMA testing.

These options default to the memory range of the type of memory being tested.

SECTIONS/TESTS

```
Section  1: Basic Tests
   Test  1: 8-bit write/read
   Test  2: 16-bit write/read
   Test  3: 32-bit write/read
```

```
   Test  4: 64-bit write/read
   Test  5: Other size write/read
Section  2: Error Detection and Correction
   Test  1: Single Bit Errors
   Test  2: Multiple Bit Errors
Section  3: Memory Array Tests
   Test  1: 32-bit Fixed patterns test
   Test  2: 32-bit Fixed patterns test (parity/checkbits)
   Test  3: 32-bit Addressing patterns test
   Test  4: 32-bit Random patterns test
Section  4: DMA Tests
   Test  1: DMA reads/writes
   Test  2: Single Bit Errors during DMA reads
Section  5: Multiprocessing Memory Tests
   Test  1: 8-bit Memory Array Test
   Test  2: 16-bit Memory Array Test
   Test  3: 32-bit Memory Array Test
   Test  4: 64-bit Memory Array Test
   Test  5: 32-bit Memory Array Test with DMA
```

NOTES

The minimum memory that can be tested with this diagnostic is 256 longwords.

Multiprocessing tests may execute for long periods of time, especially on large capacity memories.

Error detection and correction (EDAC) tests are only executed on memories that contain EDAC hardware.

Diagnostic cylinders are used on the disk, so user data is not destroyed except in cases of extreme malfunction.

SEE ALSO

saint(1D)

NAME

saint - Standalone Interface

SYNOPSIS

fr /saint;# from tape

fr /var/d/saint;# from disk

fr /usr/d/saint;# from disk if /var is symbolic link

DESCRIPTION

Saint provides an interface to the standalone diagnostics. Any group of the standalone diagnostics may be run automatically with no operator intervention. This program is fully documented in Chapter 2 of the *Diagnostic Reference Manual* (Publication Number 0855007).

SEE ALSO

timers(1D) **Diagnostic Reference Manual** **timers(1D)**

NAME

timers - test various system clocks and timers

SYNOPSIS

fr /timers , [lib_opts];# from tape

fr /var/d/timers , [lib_opts];# from disk

fr /usr/d/timers , [lib_opts];# from disk if /var is symbolic link

DESCRIPTION

Timers tests various clock and timer hardware on the system. The diagnostic verifies the proper operation of interval timer, 60Hz clocks, real-time clocks, and the time of century clock. This program executes one pass in approximately 20 seconds.

PREREQUISITES

Memory, *cache*, and *interrupt* should be run before this program is executed.

MENU OPTIONS

debug	enables/disables printing of some test results while running.
wdog	enables/disables system watchdog timeout test.

SECTIONS/TESTS

```
Section  1: Interval Timer
   Test  1: timer read/write
   Test  2: timer carry
   Test  3: timer synchronization
   Test  4: timer accuracy
Section  2: 60Hz Clocks
   Test  1: clock accuracy
   Test  2: clock spacing
Section  3: Time of Century Clock
   Test  1: read date and time
Section  4: Am9513A Real-Time Clock - Register Operations
```

```
   Test  1: master reset
   Test  2: read/write integrity
   Test  3: register integrity
   Test  4: interrupts
   Test  5: comparators
Section  5: Am9513A Real-Time Clock - Counter Operations
   Test  1: load and save
   Test  2: arm and disarm
   Test  3: binary scaling
   Test  4: counter mode A
   Test  5: counter mode D
   Test  6: watchdog timeout
Section  6: NightHawk Real-Time Clock - Register Operations
   Test  1: read/write control registers
   Test  2: read/write count registers
   Test  3: counter resolution
   Test  4: counter interrupt
   Test  5: counter repeat disabled
   Test  6: counter repeat enabled
```

NOTES

Running *timers* on each processor on every CPU board will test all timer and clock hardware.

SEE ALSO

saint(1D)

3
System Level Tests

3
System Level Tests

General

This chapter describes diagnostic products which test the system at the operating system level on Power Hawk Series 600, 700 and 900 computer systems and NightHawk Series 6000 systems.

The system-level tests currently include two diagnostic products: PowerIO and PowerAT. These products execute test sets designed to insure that every system meets a minimum level of functionality and performance. They are run on every system prior to shipment from the factory and again when that system is first installed at the customer site.

All tests run under control of the operating system and use standard operating system services.

Conventions and Definitions

The following terms and documentation conventions are used as they relate to system-level diagnostics:

test	The term test, depending on the context in which it is found, may refer to an individual test module or to the set of test modules that make up either PowerAT or PowerIO. Individual test modules are user programs designed to tax the hardware's computational and processing abilities and, where possible, provide error information useful for isolating failures to a specific hardware subsystem. Testing is accomplished by executing functions provided by the system's operating system, language processor, and program library software. Verification is accomplished by comparing the results returned by these functions with known good results.
cycle	A cycle is defined to be the completed execution of one set of test modules scheduled by the system-level test scheduler (schedat or schedio).
pass	The term pass is used to describe the successful completion of a test.
fail	The term fail (a failure) is used to describe the presence of one or more test detected errors.
schedat	Schedat is the executable object name of the PowerAT test scheduler and monitor. Schedat is invoked by and executes concurrently with PowerAT.
schedio	Schedio is the executable object name of the PowerIO test scheduler and monitor. Schedio is invoked by and executes concurrently with PowerIO.
<key>	A key on the operator's terminal keyboard is indicated by a description of the key enclosed in angular brackets. For example: <Return>, <Esc>, and <Ctrl> represent the Return, Escape, and Control keys, respectively.

Running System-Level Tests

To initiate a system-level test, move to the system level diagnostic products directory, **`/usr/d/system`**, and enter either ‘powerat’ or ‘powerio’.

The operator must be logged in as “root” to run the tests.

Detailed information concerning the startup of these programs is discussed in the sections dealing with the respective system-level test: “PowerIO – Input/Output Subsystem Test” on page 3-30 and “PowerAT – System Reliability and Acceptance Test” on page 3-46.

Test modules are auto-configured during the first invocation after installation of the system-level tests. The test configuration may be modified, saved, or auto-configured at any time (see the “Configure Menu” section).

PowerAT and PowerIO are designed to run continuously (cycle) until one of several events occurs (see Table 3-3 for a list of reasons why test cycling will stop). There is virtually no constraint on the length of time or the number of cycles for which the testing may run.

The system-level test should be started from a terminal other than the Console Processor terminal. The Console terminal receives system error messages as they occur. Since the system-level test menu displays may overwrite these messages, it is useful to leave the Console terminal free. The Console terminal also serves as a “safety valve” in case the operator’s terminal should hang for any reason.

PowerAT or PowerIO may cause the operating system (kernel) to display warning messages to the console terminal. The warning messages occur when the kernel approaches or exceeds internal limits.

The system-level test software attempts to maximize the system hardware resource usage; therefore, depending on the system configuration, internal kernel resource usage may approach or exceed the aforementioned limits.

NOTE

Kernel warning messages appearing during system-level test execution do not necessarily indicate a problem with the system. For the purpose of verifying successful system-level test execution, kernel warnings may be ignored unless they coincide with an abnormal system behavior (e.g. a test failure or a system panic, crash or hang).

User Interface

PowerAT and PowerIO have nearly identical user interfaces. This section describes the menus of both system-level test interfaces. Any description given or examples shown apply to either system-level test unless explicitly specified otherwise.

The interface is primarily menu driven, with a direct command interface built in. You are prompted by the menu selection prompt `Enter Selection>`. If the selection chosen is one of the valid options displayed by the menu, the action corresponding to the option is taken. Make a selection by typing in the option letter or digit and then pressing the **<Return>** or **<Enter>** key.

Menu Structure

The menu structure along with the actions that can be selected under each of the menus is illustrated in Figure 3-1.

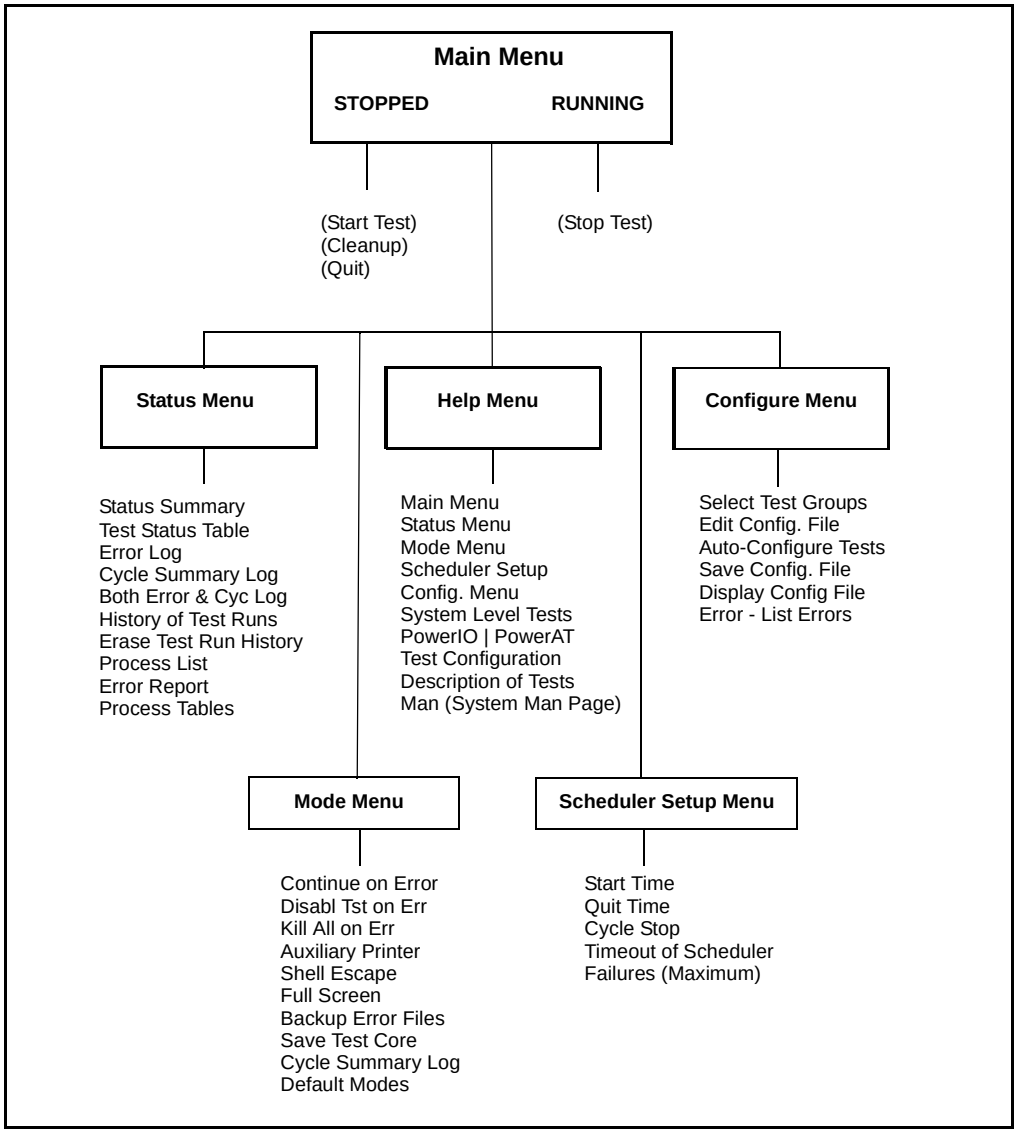

Figure 3-1. System Test - Menu Hierarchy

The following subsections describe functions of the interface common to both PowerAT and PowerIO including the menus, their options, and the options taken by the particular system-level test.

Command Interface

The system-level tests (PowerIO and PowerAT) provide a direct command interpreter that is available in addition to the menu driven interface. The commands available fall into four categories:

- Go to menu command: Go directly to any menu by entering the menu name at the prompt line.
- Menu option command: Any option can be selected from within a menu by entering the first word of the option description at the prompt line. Option commands may be used instead of the one-character menu selections.
- Combination "go to menu /menu option" commands: A menu option may be selected from another menu by entering the menu [name] command followed by *either* the option letter or option command on the prompt line. For example, if the user wants to display the Status Menu's Error Log option (3) from the Main Menu, the following commands at the prompt line would do so:

  `Enter Selection>` **`status 3`** **<Return>**

  or

  `Enter Selection>` **`status error`** **<Return>**

- Generic commands: These commands, given in Table 3-1, may be entered at the prompt line of *any* menu:

Table 3-1. Generic Prompt Line Commands

Command	Description
`!`	Run shell or command: ![<command>]
`?`	Get help: ?[<command>|<menu>]
`ps`	System **`ps(1)`** command
`man`	System **`man(1)`** command
`exit` or **<Esc>**	Return to previous menu
`run`	Run/Start tests
`stop`	Stop tests or query reason stopped
`cleanup`	Clean up before next run
`tty`	Enter 'tty' (non-full screen)display mode
`usrcheck`	Disable or enable PowerAT non-root user checking

NOTE

Commands are recognized as soon as they can be uniquely identified (i.e. the user need not enter all characters of a command).

Help On Menu

Entering **<H>** or **<?>** from the menu prompt line will give help on the current menu.

Exit Menu

As shown in Table 3-1 above, you can return to the previous menu by entering the ‘exit’ command or by typing the **<Esc>** key. As a reminder, the message “<Esc> to Exit Menu” is displayed on all lower level menu displays.

Menu Redraw

At any time during the display of a menu, the current menu can be redrawn. It is possible for the menu display to get “garbled” (e.g. from noise on a dial up line) or scroll off the screen if full screen mode is OFF (see the “Mode Menu” section on page 3-16). Typing **<Ctrl> <R>** (pressing the control key and the letter ‘R’ at the same time) or **<Ctrl> <L>** (pressing the control key and the letter ‘L’ at the same time) will redraw the current menu screen.

System Identification

To assist the operator running system-level tests on multiple systems, an identification string is embedded in the bottom border of each menu. The string consists of the network node name, the operating system type, the operating system release level, and the system architectural model (e.g. HN6800).

Configuration/Status/Result File Directories

All files and directories generated by system-level tests are written under the **`/var/d/system/I<inode>J<major>N<minor>`** directory; where, **`<inode>`** is the inode number of the directory where system-level tests are loaded (typically **`/usr/d/system`**), and **`<major>`** and **`<minor>`** are the major and minor device numbers of the file system under which the system-level tests are loaded. Table 3-2 summarizes the files and directories found under this directory.

Table 3-2. Configuration/Status/Result Files/Directories

Name	File/Directory	Contents
`.loadtime`	File	Time and date of system-level test installation (**`load_diags`**)
`.hdwcfg_history`	File	Binary data representing a history of hardware configurations detected since last system-level test installation
`run_log`	File	A history of system-level test runs
`HOME`	File	The path name of the system-level test executable directory; e.g.: **`/<mountpoint>`** | **`<installdir>/d /system`**

Table 3-2. Configuration/Status/Result Files/Directories (Cont.)

Name	File/Directory	Contents
`PowerIO.conf`	Directory	PowerIO test configuration files
`PowerAT.conf`	Directory	PowerAT test configuration files
`PowerIO.stat`	Directory	PowerIO system-level test status and error files
`PowerAT.stat`	Directory	PowerAT system-level test status and error files
`PowerIO.rslt`	Directory	PowerIO test module output files
`PowerAT.rslt`	Directory	PowerAT test module output files

Paginator

Most of the actions associated with the display of help or status use a function similar to the **`pg(1)`** filter. The paginator is a file perusal function which is used to display backwards and search forwards or backwards for a particular character pattern. At the end of each screenful of display, the paginator prompts (with a colon character) for the next command. Help on the possible commands to the paginator can be displayed by entering **<H>** (for Help) at the prompt.

Attention Messages

PowerAT and PowerIO display messages for any condition that may require attention. These messages are designed to alert the user to important conditions such as the system-level test stopping due to failures.

Upon startup, if the abnormal termination of a previous run is detected, an attention message is displayed:

```
!! ATTENTION: The last run of PowerIO exited abnormally (crashed?)!!
```

During initialization, if the last run was not “cleaned up” an attention message is displayed:

```
CLEANUP not done on last exit from PowerIO - last status files are intact.
```

An attention message is displayed if the test scheduler is STOPPED due to detected failure(s). For example:

```
TEST SCHEDULING STOPPED: ERROR LOG IS FULL. TOO MANY FAILURES - CHECK STATUS
TEST SCHEDULING STOPPED: ERROR DETECTED WITH “Kill All On Error” MODE SET
TEST SCHEDULING STOPPED: ALL TESTS FAILED WITH “Disable Tst On Err” MODE SET
TEST SCHEDULING STOPPED: TEST(S) FAILED WITH “Continue On Error” MODE OFF
TEST SCHEDULER TIMEOUT: A TEST (OR TESTS) DIDN'T COMPLETE IN TIME (HUNG?).
```

Main Menu

The Main Menu offers different options depending on the run state. One set of options is available while testing is STOPPED, and another while RUNNING. Both sets include options for accessing the other menus. Menu options which depend on the run state are discussed below.

Screen 3-1 shows an example of the Main Menu in the STOPPED state. If testing is STOPPED, the Main Menu options allow the user to select a menu, start testing (Run), cleanup a previous run, or quit (return to the shell from which the system-level test was invoked).

```
================================================================================
                    PowerIO - Input/Output Subsystem Test
                                Release 9.8

  Main Menu          1 > Run - Start Test         6 > Configure Menu
                     2 > Status Menu
                     3 > Mode Menu
                     4 > Help Menu                C > Cleanup
                     5 > Scheduler Setup          Q > Quit - Leave Test

  Enter Selection> []
 +------------------------------------------------------------------------+
 | Status Summary        Mon Mar 15 11:05:41 2004     PowerIO  is  STOPPED  |
 |                                                                        |
 | Cycles:       0 Complete                Elapsed Run  Time:      0:00:00 |
 | Tests:        0 Total                   Cycle Times: Current    0:00:00 |
 |               0 Queued     0 Disabled                Previous   0:00:00 |
 |               0 Active     0 Complete                Average    0:00:00 |
 | Failures:     0 Total      0 Current     Start Time:                     |
 +------------------------------------------------------------------------+

====================abbu: 6.1-20040127 6.1 Synergy mpc7455====================
```

Screen 3-1. Example Main Menu (PowerIO/STOPPED)

If a system-level test is RUNNING, the Cleanup and Quit - Leave Test options are not available and option 1, Run - Start Test, becomes Stop - Stop Test. Screen 3-2 shows an example of the Main Menu in the RUNNING state.

```
================================================================================
               PowerAT - System Reliability and Acceptance Test
                                Release 9.8

  Main Menu          1 > Stop - Stop Test         6 > Configure Menu
                     2 > Status Menu
                     3 > Mode Menu
                     4 > Help Menu
                     5 > Scheduler Setup

  Enter Selection> []
 +------------------------------------------------------------------------+
 | Status Summary        Mon Mar 15 11:05:41 2004     PowerAT  is  RUNNING  |
 |                                                                        |
 | Cycles:       0 Complete                Elapsed Run  Time:      0:00:00 |
 | Tests:        0 Total                   Cycle Times: Current    0:00:00 |
 |               0 Queued     0 Disabled                Previous   0:00:00 |
 |               0 Active     0 Complete                Average    0:00:00 |
 | Failures:     0 Total      0 Current     Start Time:                     |
 +------------------------------------------------------------------------+

====================abbu: 6.1-20040127 6.1 Synergy mpc7455====================
```

Screen 3-2. Example Main Menu (PowerAT/RUNNING)

Option 1 > Run - Start Test

Selection of this option from the Main Menu starts execution of the system-level test. The Run option is only displayed if testing is STOPPED. PowerAT and PowerIO differ somewhat in their startup sequence.

PowerAT immediately invokes the test scheduler which in turn schedules execution of the configured test suite. The message “Initializing Test Scheduler...” appears on the menu prompt line until schedat actually starts invoking tests.

NOTE

PowerAT normally will not allow test startup if there are users logged in on other terminals. The **-u** option during PowerAT invocation time or the ‘usrcheck’ generic command may be used to defeat this restriction.

PowerIO will display a summary of the scheduled tests before invoking the test scheduler.

The system-level test scheduler, when invoked, will start concurrent execution of as many scheduled tests as the system can sustain. The number of tests which may be run concurrently depends on the size of each test, available system swap space, and the maximum number of processes allowed.

NOTE

Response time may be severely impacted during execution (especially the first few minutes of each test cycle). All key strokes are read as they are entered but input echoing and response to input may take up to several minutes during certain stages of a test cycle.

Run has been implemented as a generic command (see the section “Command Interface”). A ‘run’ command can be issued from any menu.

Option 1 > Stop - Stop Test

The Stop option from the Main Menu halts execution of system-level test scheduling. Currently executing test modules are killed (no status is obtained from these) and the test scheduler exits without scheduling any further tests.

NOTE

After issuing the kill, the test scheduler exits without waiting on the exiting tests (child processes).

This option is only displayed if the system-level test is RUNNING.

Stop has been implemented as a generic command (see the section “Command Interface”). A ‘stop’ command can be issued from any menu and, if running, testing will be stopped.

While stopped, the stop command can be issued to query the reason testing was stopped. The ‘stop’ command will display one of the messages in Table 3-3 on the option selection prompt line:

Table 3-3. Stop Command - Query Reason Stopped

`Operator stopped testing` Testing was stopped manually via the user selecting Main Menu option 1, Stop Test, or by entering the ‘stop’ command.
`Test failed and Continue On Error mode was OFF` The Mode Menu option 1, Continue On Error, was OFF at the completion of a cycle in which one or more test modules failed. (See “Mode Menu” on page 3-16 for more information.)
`All tests failed with Disable Tst On Err mode ON` The Mode Menu option 2, Disable Tst On Err, was ON when all scheduled tests failed (been disabled), leaving no enabled tests to schedule. (See “Mode Menu” on page 3-16 for more information.)
`A test failed and Kill All On Error mode was ON` The Mode Menu option 3, Kill All On Error, was ON and one of the scheduled tests failed. (See “Mode Menu” on page 3-16 for more information.)
`Maximum failures (100) detected` The maximum total failures, specified by option 5 in the Scheduler Setup menu, was reached. (See “Scheduler Setup Menu” on page 3-24.)
`Scheduler Timeout Exceeded (0 hr 20 min)` Testing was stopped because the test scheduler/monitor test time out value, specified in the Scheduler Setup menu, was exceeded. The test scheduler times out if a test or tests do not exit within the time specified since the last detected test completion (exit). (See “Scheduler Setup Menu” on page 3-24.)
`Programmed stop after specified cycle` Option 3, Cycle Stop, of the Scheduler Setup menu was set and testing ran for the number of cycles specified. **Note**: The Cycle Stop option is reset to MANUAL when testing is stopped. (See “Scheduler Setup Menu” on page 3-24.)
`Programmed stop after specified time` Option 2, Quit Time, of the Scheduler Setup menu was set and testing ran until the time specified. The time is checked at the end of each cycle, thus guaranteeing completion of a full cycle. **Note**: The Quit Time option is reset to MANUAL when testing is stopped. (See “Scheduler Setup Menu” on page 3-24.)
`schedio exited abnormally, PowerIO aborted` The test scheduler exited for unknown reasons causing PowerIO (or PowerAT) to abort and exit.
`Already stopped` The ‘stop’ command is issued after a cleanup of the previous test run.

Option 2 > Status Menu

The Status Menu option from the Main Menu displays the Status Menu. The Status Menu provides a display of test and system status as well as an interface to failures that occur during execution of system-level tests. See “Status Menu” on page 3-11 for more information.

Option 3 > Mode Menu

User and test interface modes are set/reset when the Mode Menu is selected from the Main Menu. See “Mode Menu” on page 3-16 for more information.

Option 4 > Help Menu

The Help Menu option from the Main Menu gives access to help about the system-level test, as well as access to any available online system man page. See “Help Menu” on page 3-22 for more information.

Option 5 > Scheduler Setup

The Scheduler Setup option from the Main Menu provides user programmable parameters for test scheduling. Included are start and stop times, maximum cycle count, test scheduler time-out value, and maximum failure count. See “Scheduler Setup Menu” on page 3-24 for more information.

Option 6 > Configure Menu

Selection of this option from the Main Menu displays the Configure Menu where you can view, modify, save or reset the configuration of tests to be run. See “Configure Menu” on page 3-27 for more information.

Option C > Cleanup

The Cleanup option from the Main Menu initializes the system-level test in preparation for the next run. All status information and result files are initialized. If a previous run was STOPPED but no cleanup was done, cleanup *must* be selected before starting the next run. This option is only displayed if the testing is STOPPED.

Cleanup has been implemented as a generic command (see “Command Interface”). A ‘cleanup’ command can be issued from any menu, but the command is invalid while testing is RUNNING.

Option Q > Quit - Leave Test

This option from the Main Menu is selected when testing is complete. Quit leaves the system-level test environment and returns control to the shell.

NOTE

System-level testing can not be exited while RUNNING.

Status Menu

The Status Menu is entered by selecting Main Menu option 2 or by entering the ‘status’ command at the menu prompt. The Status Menu options provide actions associated with the system-level test status, system process status, and system error status.

NOTE

Some errors may occur and not be detected directly by the system-level tests. The operating system may detect and report errors to the system error logging daemon, or to the console terminal, or to the user’s terminal. Also, some tests, such as the printer test (prnt) in PowerIO, require visual inspection of the output to verify correct completion.

Screen 3-3 illustrates the Status Menu. Except for the title, the display is identical for PowerIO.

```
================================================================================
               PowerAT - System Reliability and Acceptance Test
                           Release 9.8 - Preliminary

   Status Menu        1 > Status Summary           6 > History of Test Runs
                      2 > Test Status Table        7 > Erase Test Run History
   <Esc> to           3 > Error Log                P > ps (Process List)
   Exit Menu          4 > Cycle Summary Log
                      5 > Both Error & Cycle Log

   Enter Selection> []
 +------------------------------------------------------------------------------+
 | Status Summary        Mon Mar 15 11:05:41 2004     PowerAT  is  STOPPED  |
 |                                                                              |
 | Cycles:       0 Complete                 Elapsed Run  Time:      0:00:00 |
 | Tests:        0 Total                    Cycle Times: Current    0:00:00 |
 |               0 Queued     0 Disabled                Previous   0:00:00 |
 |               0 Active     0 Complete                Average    0:00:00 |
 | Failures:     0 Total      0 Current     Start Time:                     |
 +------------------------------------------------------------------------------+

====================abbu: 6.1-20040127 6.1 Synergy mpc7455====================
```

Screen 3-3. Example Status Menu

Option 1 > Status Summary

Option 1 from the Status Menu is the Status Summary. The Status Summary option is only meaningful if the full screen mode is off or the auxiliary print mode is on (see “Mode Menu” on page 3-16). This is because the status summary is always displayed on the user’s terminal if it has full screen cursor addressing capabilities and the full screen mode is on.

The status summary gives a “snapshot” of the following information:

- The current date and time.
- The current state: STOPPED or RUNNING.
- The number of fully completed cycles.
- A summary of test activity during a cycle:
  - The total number of tests scheduled to execute. This number remains constant while the system-level test is running. The test total is always the sum of the next four categories.
  - Queued. The number of tests left to be scheduled in the current cycle.
  - Disabled. The number of tests which have failed cycles with Mode Menu option 2, Disable Tst On Err, ON (see “Option 2 > Disable Tst On Err” on page 3-18).

NOTE

This category does not reflect tests disabled manually from the PowerIO Select Test Groups menu.

  - Active. The number of tests running in the current cycle.
  - Complete. The number of completed tests in the current cycle.
- A summary of test detected failures:
  - The total number of failures detected.

NOTE

Testing will stop when the maximum number of failures have been detected (default is 100, see “Option 5 > Failures (Maximum)”on page 3-27). This is a safety measure to ensure the error log file will not grow unbounded.

  - The number of failures occurring in the current cycle.
- The elapsed time of execution of the current (or last) run.
- A summary of cycle run times:
  - The current cycle’s elapsed time. This time is equal to zero after cleanup and is zeroed at the end of each completed cycle.
  - The previous cycle’s elapsed time. This time is equal to zero after cleanup and during the first cycle.
  - The average cycle completion time. This time is equal to zero after cleanup and during the first cycle.
- The time at which the current (or last) run started.

NOTE

The values for the last five bullet items above are “frozen” at the time a run is stopped (automatically or by the operator) and remain available until the next “Cleanup” is selected.

Option 2 > Test Status Table

Selection of option 2 or entering the option command ‘test’ from the Status Menu displays the Test Status Table. Each row of the table represents the pass/fail status of a test module.

The columns of this table contain the following information about each test:

- The test name and a unit identification (e.g. device name/number, cpu bias mask, etc.).
- The number of times the test has been scheduled.
- The number of times the test has completed.
- The number of times the test detected a failure.
- The number of times the test passed.
- The last cycle the test detected a failure.
- If failures have been detected by the test, the date and time of the last detected failure.

The Test Status Table is updated by the test scheduler each time a test is scheduled and when completion of a test is detected.

Option 3 > Error Log

Option 3 or the option command ‘error’ from the Status Menu displays a chronological list of test detected failure descriptions. A failure description contains the following information:

- The keyword “ERROR” (may be used to search for the start of the next error log entry).
- The test name.
- The process identification number (pid) of the test.
- The cycle number in which the failure was detected.
- The date and time at which the failure was detected.
- The number of failures previously detected by the test.
- The last cycle in which the test detected a failure.
- The exit status of the test. The exit status is a non-zero number (e.g. 1) for a test which detected a failure.
- A nine bit map (displayed in hexadecimal) of the Mode Menu modes, interpreted as shown in Figure 3-2 below.

Bit	8	7	6	5	4	3	2	1	0
Mode	C	D	K	A	E	F	B	S	L

Mode	Description*	Default	Value
C	Continue on Error	OFF	0
D	Disable Test On Error	OFF	0
K	Kill All On Error	OFF	0
A	Auxiliary Printer	OFF	0
E	Shell Escape	OFF	0
F	Full Screen	ON	0
B	Backup Error Files	OFF	0
S	Save Test Core	OFF	0
L	Cycle Summary Log	OFF	0

*See “Mode Menu” on page 3-16 for detail on modes.

Figure 3-2. Mode Bits in Error Log

- If the operating system terminated the test, a description of what “signal” caused the test to exit.
- If generated, the name of the core image file (see the Mode Menu subsection “Option 8 > Save Test Core”).
- The first 2 KB of the test’s diagnostic output describing the failure. Test result information greater than 2 KB is truncated (not copied to the error log). See the Mode Menu subsection “Option 7 - Backup Error Files” for a method of preserving result information in its entirety.

The following is an example of an error log entry:

```
*** ERROR
*** Test 'disk.rdsk 2010' (pid 183) failed cycle 1 on Sat Nov 4 11:00:19
*** Exit status was 0, mode bits: 0x117.
*** Test 'disk.rdsk 2010' has failed 0 times previously.
*** NOTE: A bus error signal (signal number 10) terminated the test.
*** Core image written to: PowerIO.rslt/core.183.001
*** Description of failure:
<Failure description written by test>
```

The test error log is updated by the test scheduler each time a failure is detected. The test error log is capable of storing a variable number of test detected failure descriptions (default is 100 - see the Scheduler Setup subsection “Option 5 > Failures (Maximum)”). If the maximum number of detected

failures is reached, test scheduling is stopped immediately, all currently active tests are aborted, and the following attention message is displayed:

```
TEST SCHEDULING STOPPED: ERROR LOG IS FULL. TOO MANY FAILURES - CHECK STATUS
```

Option 4 > Cycle Summary Log

This option from the Status Menu is only meaningful if option 9, Log Cycle Summary, in the Mode Menu is turned ON when running. The following information is displayed for cycles completed:

- Cycle number
- Duration of cycle
- Average cycle duration
- Failures that cycle
- Total failures
- Trace (debug) level – only for use in debugging the test software
- Mode bits (see Figure 3-2)
- Cycle start time
- Cycle completion time

The following is an example display:

```
**
** Cycle Summary. Cycle 1     Failures: 0 Total: 0
**                 Trace level 0, Mode bits: 0x115
** Cycle Time: 00:00:50 Started:   Tue Jan 24 14:57:05
** Avg. Time: 00:00:50 Completed: Tue Jan 24 14:57:55
**
**
** Cycle Summary. Cycle 2     Failures: 0 Total: 0
**                 Trace level 0, Mode bits: 0x010
** Cycle Time: 00:00:50 Started:   Tue Jan 24 14:57:55
** Avg. Time: 00:00:50 Completed: Tue Jan 24 14:58:45
**
```

WARNING

The cycle summary log grows, currently by 52 bytes, with each completed cycle. It is possible to eventually exhaust available disk space running the system-level tests with the Log Cycle Summary mode ON. Also, display of the cycle summary log causes creation of a temporary display file approximately 3.6 times the size of the log file.

Option 5 > Both Error & Cycle Log

This option from the Status Menu displays a chronological log of the information described in options 3 and 4.

Option 6 > History of Test Runs

This option from the Status Menu gives the operator access to a chronological log file of system-level test runs. Entries in this file include:

- The date and time of the system-level test installation or when the history log was last erased.
- System-level test startup information consisting of the system-level test name (PowerAT or PowerIO), start time, a system identification, and a test configuration summary.
- Error summary for each error consisting of the system-level test name, the time of error, the cycle the error occurred, the 16 bit exit status and the name of the test module that failed.
- System-level test stop information consisting of the system-level test name, the stop time, the number of cycles started, the total failures detected during the test run and a comment describing why testing was stopped.

Option 7 > Erase Test Run History

This option from the Status Menu is used to purge the contents of the Run History log. The operator is prompted to enter ‘yes’ to confirm.

Option P > ps (Process List)

This Status Menu option will display the Process Status Table. The operator is prompted to enter any or none of the ‘ps’ command options. The information displayed is the output of the **`ps(1)`** command (see the **`ps(1)`** man page).

Ps has been implemented as a generic command (see the section “Command Interface”). The user may issue a ‘ps’ command from any menu. If arguments are used, no prompting is done.

Mode Menu

The Mode Menu is accessed through selection of Main Menu option 3 or by entering the ‘mode’ command at the menu prompt. The Mode Menu provides options to configure the input, output and error modes of the operator and test interface. In the Mode Menu, the operator may set the action to be taken by the test scheduler upon test failures:

- Continue to cycle test scheduling regardless of test detected failure.
- Disable execution of tests detecting failures and continue to cycle.
- Kill tests and stop immediately if any test detects a failure.
- Back up test error output.
- Save test core dump files.

Also in this menu, you can enable or disable (toggle) full screen mode, auxiliary printer mode, shell escape mode, and cycle summary logging. **Changes made to the modes are permanently saved, automatically.**

NOTE

Modes which affect test execution parameters (options 1 through 3 and 7 through 9 shown below) take effect in subsequent cycles provided they are set prior to the start of the next cycle.

Screen 3-4 illustrates the Mode Menu. Except for the title, the display is identical for PowerIO.

```
================================================================================
                PowerAT - System Reliability and Acceptance Test
                             Release 9.8 - Preliminary

   Mode Menu          1 > Continue On Error..OFF   6 > Full Screen.........ON
                      2 > Disable Tst On Err.OFF   7 > Backup Error Files.OFF
   <Esc> to           3 > Kill All On Error..OFF   8 > Save Test Core.....OFF
   Exit Menu          4 > Auxiliary Printer..OFF   9 > Cycle Summary Log..OFF
                      5 > Shell Escape.......OFF   D > Default Modes

   Enter Selection>
  +------------------------------------------------------------------------+
  | Status Summary        Mon Mar 15 13:21:01 2004    PowerAT  is  STOPPED   |
  |                                                                        |
  | Cycles:       0 Complete                Elapsed Run  Time:      0:00:19 |
  | Tests:      242 Total                   Cycle Times: Current    0:00:19 |
  |              15 Queued    0 Disabled                 Previous   0:00:00 |
  |             161 Active   66 Complete                 Average    0:00:00 |
  | Failures:     0 Total     0 Current     Start Time: Mon Mar 15 11:09:00 |
  +------------------------------------------------------------------------+

=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ===================
```

Screen 3-4. Example Mode Menu

The meanings and actions associated with each of the modes are described in more detail in the following subsections.

Option 1 > Continue On Error

By default, any test module failure will cause the system-level test scheduling to stop at the end of the current cycle (i.e. after all test modules have exited). This mode must be turned ON to continue scheduling test modules in subsequent cycles regardless of previous test module failures.

If OFF and any test fails, testing is STOPPED at the end of the cycle and the following attention message is displayed:

```
TEST SCHEDULING STOPPED: TEST(S) FAILED WITH "Continue On Error" MODE OFF
```

The default for this mode is OFF.

Option 2 > Disable Tst On Err

This mode enables/disables scheduling of failing test modules in subsequent cycles. If this mode is toggled ON before the start of any cycle, failing tests will be disabled (not scheduled) in subsequent cycles. The number of disabled tests appears in the status summary (see the Status Menu subsection “Option 1 > Status Summary” on page 3-11).

If this mode is toggled OFF before the start of any cycle, tests that have been previously disabled are re-enabled (scheduled) in subsequent cycles. The status summary is updated to reflect zero (0) disabled tests. While this mode is OFF, disabled test modules are re-enabled and failing tests are not disabled.

NOTE

Because turning Disable Tst On Err mode ON implies continuous cycling after an error is detected, Continue On Error mode is also toggled ON, if not already ON. Turning Disable Tst On Err mode OFF does not affect the state of the Continue On Error mode.

If all tests fail (are disabled) while this mode is ON, the scheduler is STOPPED and an attention message is displayed:

```
TEST SCHEDULING STOPPED: ALL TESTS FAILED WITH “Disable Tst On Err” MODE SET
```

The default for this mode is OFF.

Option 3 > Kill All On Error

If this mode is set (ON), test scheduling is stopped completely after the first test module detected failure and an attention message is displayed:

```
TEST SCHEDULING STOPPED: ERROR DETECTED WITH “Kill All On Error” MODE SET
```

Test modules which have not completed in the current cycle are terminated. No exit status is checked or recorded for tests terminated by the scheduler. The Continue On Error and Disable Tst On Err modes are automatically turned OFF when Kill All On Error is toggled ON.

The default value for this mode is OFF.

Option 4 > Auxiliary Printer

This option from the Mode Menu toggles the activation of an auxiliary printer to be used to capture a hardcopy of all input and output to the user screen. This mode is OFF at startup. When selected in the OFF state, this option will first prompt for the device name of the port to use.

Example:

```
Enter TTY Device File Name (e.g. /dev/ttyx_yy)> /dev/tty0_16<Return>
```

If the device name entered is a valid and available tty port, baud rate is prompted for (default is 9600).

Example:

```
Enter TTY Device Baud Rate (<return> = 9600)> <Return>
```

If the auxiliary printer mode is ON, selecting this menu option will display the prompt:

```
Turn OFF print mode on "/dev/ttyX_YY" (enter "yes" or "no")? >
```

Entering anything but "no" will turn OFF this mode.

The **`ttymon(1M)`** process (the process that waits for user logins) must be disabled on the async port selected for the auxiliary printer prior to enabling the printer from the menu. To do so, issue the appropriate **`pmadm(1M)`** commands. If the auxiliary printer port has a ttymon running on it, the following attention message is displayed:

```
ABORT: NOT_A_TTY or TTYMON_RUNNING.
```

The default value for this mode is OFF.

Option 5 > Shell Escape

This option from the Mode Menu toggles a mode which provides a means to execute shell commands from the menu selection prompt. The shell escape mode is initially turned OFF. The test scheduler (schedio or schedat) must be STOPPED to enable this mode. Once enabled, testing can be started. With shell escape mode ON, the user can execute any shell command at the menu selection prompt by preceding it with an exclamation mark ('!'). An exclamation mark alone puts the user into a sub-shell. Once in a sub-shell, the 'exit' command is used to return to the system-level test.

The following example would cause suspension of menu selection then execution of the 'date' command. The operator is prompted to type the **<Return>** key to continue (return to menu selection).

Example:

```
Enter Selection>!date <Return>
Working...
Thu Aug 21 11:02:46 EDT 1986
Type <return> to continue>
```

The default value for this mode is OFF.

Option 6 > Full Screen

The Full Screen option from the Mode Menu enables the user to switch from full screen mode to scroll mode (tty mode or hardcopy mode) and visa versa. The full screen mode is automatically set or reset upon startup. If the TERM environment variable is set and the defined terminal type has cursor addressing capability, full screen mode is set ON.

In full screen mode, the current menu display is maintained on the upper half of the screen while the status summary is being updated every second on the lower half of the screen. Operator attention messages are displayed on line 22 of the screen.

While full screen mode is OFF, the menu display is scrolled a line at a time to the terminal. Only the menu selection prompt is re-displayed after each input. If the menu display scrolls off the screen,

<**Ctrl**><**R**> will redraw the menu. The full screen OFF mode is useful to connect a printer to the auxiliary port of the terminal in order to capture a hardcopy of the system-level test interface input and output (e.g. the Test Error Log, etc.).

If the full screen mode is OFF, select the Full Screen option to turn it ON. You are prompted to enter the terminal type or hit the **<Return>** key to use the last known terminal type.

Example:

```
Enter Terminal Type (<return> = "h8665")> <Return>
```

If the terminal type specified has full screen capabilities, full screen mode is turned ON and the Mode Menu and Status Summary are displayed in full screen mode.

This option can be selected if full screen mode is ON and the user wishes to turn it OFF. The generic 'tty' command can also be used to turn full screen mode OFF.

Option 7 > Backup Error Files

When scheduling the execution of a test, the scheduler/monitor (schedat or schedio) creates the test's result file for the current cycle. The file is named:

Power*xx*.rslt/<name>.<unit>.<pid>.<cycle>

where:

xx	is 'AT' for PowerAT, and 'IO' for PowerIO
<name>	is the test name (e.g. disk, term, fppdbl, fppsgl, etc.)
<unit>	is the unit under test (i.e. device number or cpu bias)
<pid>	is the process ID of the test
<cycle>	is the current test cycle number

If this option from the Mode Menu is disabled (OFF), the test's result file from the previous cycle is removed. If this option is enabled (ON) and the test failed in the last cycle, the test result file is not removed. The default for this option is OFF.

NOTE

After a test detected error, the first 2 KB of the result file is always copied to the error log (see the Status Menu subsection "Option 3 > Error Log" on page 3-13). This mode may be helpful in the event the result file contains more than 2 KB of data.

Option 8 > Save Test Core

This option from the Mode Menu, when enabled (ON), will cause the test scheduler to save (rename) test core files to:

`Powerxx.rslt/core.<pid>.<cycle>`

where:

xx	is 'AT' for PowerAT, and 'IO' for PowerIO
`<pid>`	is the process ID of the test
`<cycle>`	is the current test cycle number

A core image file named 'core' is created if a test is failing due to receipt of one of the following UNIX signals (see the **`signal(2)`** man page):

Signal Mnemonic	Signal Number	Signal Description
SIGQUIT	3	quit
SIGILL	4	illegal instruction
SIGTRAP	5	trace trap
SIGABRT	6	ABORT instruction
SIGEMT	7	EMT instruction
SIGFPE	8	floating point exception
SIGBUS	10	bus error
SIGSEGV	11	segmentation violation
SIGSYS	12	bad argument to system call

The default value of the Save Test Core mode is OFF.

Option 9 > Cycle Summary Log

If this option from the Mode Menu is enabled (ON), the following information is logged upon completion of each cycle:

- Cycle number
- Failures that cycle
- Total failures
- Trace (debug) level – only for use in debugging system-level test software
- Mode bits (see Figure 3-2)
- Duration of cycle
- Average cycle duration
- Cycle start time
- Cycle completion time

This information can be displayed from the Status Menu.

WARNING

The cycle summary log grows, currently by 52 bytes, with each completed cycle. It is possible to eventually exhaust available disk space running the system-level tests with this mode ON. Also, display of the cycle summary log (via Status Menu option) causes creation of a temporary display file approximately 3.6 times the size of the log file.

The default value for this mode is OFF.

Option D > Default Modes

When this option is selected from the Mode Menu, all the Mode Menu options are reset to their default values.

Help Menu

The Help Menu gives the user online access to this Diagnostic Reference Manual as well as any installed man pages.

The following illustrates the Help Menu. Except for the references to “PowerAT”, the display is identical for PowerIO. The options under the help menu dealing with PowerAT or PowerIO Specific Help will display this manual starting at the section relative to the option selected.

```
================================================================================
                PowerAT - System Reliability and Acceptance Test
                            Release 9.8 - Preliminary

    Help Menu          1 > Main Menu Help           6 > System Level Tests
                       2 > Status Menu Help         7 > PowerAT Specific Help
    <Esc> to           3 > Mode Menu Help           8 > Test Configuration
    Exit Menu          4 > Scheduler Setup Help     9 > Description of Tests
                       5 > Config. Menu Help        M > man (System Man Page)

    Enter Selection>
  +------------------------------------------------------------------------+
  | Status Summary        Mon Mar 15 14:00:47 2004    PowerAT  is  STOPPED   |
  |                                                                        |
  | Cycles:       0 Complete                Elapsed Run  Time:      0:00:19 |
  | Tests:      242 Total                   Cycle Times: Current    0:00:19 |
  |              15 Queued    0 Disabled                 Previous   0:00:00 |
  |             161 Active   66 Complete                 Average    0:00:00 |
  | Failures:     0 Total     0 Current     Start Time: Mon Mar 15 11:09:00 |
  +------------------------------------------------------------------------+


=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ===================
```

Screen 3-5. Example Help Menu

Option 1 > Main Menu Help

The Main Menu Help option displays this manual starting at the Main Menu section.

Option 2 > Status Menu Help

The Status Menu Help option displays this manual starting at the Status Menu section.

Option 3 > Mode Menu Help

The Mode Menu Help option displays this manual starting at the Mode Menu section.

Option 4 > Scheduler Setup Help

The Scheduler Setup Help option displays this manual starting at the Scheduler Setup Menu section.

Option 5 > Config. Menu Help

The Config. Menu Help option displays this manual starting at the Configuration Menu section.

Option 6 > System Level Tests

This option begins display of this manual at the first page of the System Level Tests chapter.

Option 7 > {PowerIO|PowerAT} Specific Help

Help option 7 will display this manual starting at the appropriate (either PowerIO or PowerAT) section.

Option 8 > Test Configuration

If the Test Configuration option is selected, the section of the manual describing the configuration of tests under PowerIO or PowerAT is displayed (e.g. depending on which system-level test is active).

Option 9 > Description of Tests

The Description of Tests option displays this manual starting at the appropriate (PowerIO or PowerAT) Test Descriptions section.

Option M > man (System Man Page)

This option will display any man page available on the system. The information displayed is the output of the **`man(1)`** command (see the **`man(1)`** man page). When selected, you are prompted to enter any or none of the arguments to the **`man`** command. In most cases, the argument supplied will be the name of a man page. If no arguments are supplied, the introduction man pages of Section 2 (**`Intro(2)`**) are displayed. **`Intro(2)`** gives a list of system error numbers and their descriptions.

Man has been implemented as a generic command (see Command Interface section above). A ‘man’ command can be issued from any menu. If no arguments are given, you are prompted.

Scheduler Setup Menu

The Scheduler Setup Menu is designed to give the user some flexibility scheduling test runs.

Screen 3-6 illustrates the Scheduler Setup Menu. Except for the title, the display is identical for PowerIO.

```
================================================================================
                PowerAT - System Reliability and Acceptance Test
                            Release 9.8 - Preliminary

    Scheduler Setup    1 > Start Time......MANUAL
                       2 > Quit Time.......MANUAL
    <Esc> to           3 > Cycle Stop......MANUAL
    Exit Menu          4 > Timeout of Scheduler :   0  Hours, 20 Minutes
                       5 > Failures (Maximum)   :   100

    Enter Selection>
  +------------------------------------------------------------------------+
  | Status Summary        Mon Mar 15 14:10:34 2004    PowerAT  is  STOPPED   |
  |                                                                        |
  | Cycles:       0 Complete                Elapsed Run  Time:      0:00:19 |
  | Tests:      242 Total                   Cycle Times: Current    0:00:19 |
  |              15 Queued    0 Disabled                 Previous   0:00:00 |
  |             161 Active   66 Complete                 Average    0:00:00 |
  | Failures:     0 Total     0 Current     Start Time: Mon Mar 15 11:09:00 |
  +------------------------------------------------------------------------+


=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ===================
```

Screen 3-6. Scheduler Setup Menu - Example 1

Option 1 > Start Time

This option from the Scheduler Setup Menu is used to set the date and time when system-level testing should begin. The menu displays “Start Time.....MANUAL” as shown in Screen 3-6 above when the Start Time is not set. If selected and Start Time is MANUAL, you are prompted to enter the desired start time. If selected and the start time is already set, the start time is reset to MANUAL (i.e. auto startup is disabled).

The time can be entered in one of three formats:

1. A date/time format similar to **`date(1)`**: MMDDHHMM[YY],
   where MM is a two digit value representing the month
   DD is a two digit value representing the day
   HH is a two digit value representing hours
   MM is a two digit value representing minutes
   YY is an optional two digit value representing the year

   This format must specify a date/time greater than the current date/time (e.g., if the current time is "Thu Apr 15 15:32:08 2004", the start time using this format would need to be 04151533 or better). **Note:** If you will be crossing a year boundary, utilize methods 2 or 3.

2. HH:MM, where: HH is a two digit value representing hours
   MM is a two digit value representing minutes

   This format expects you to add the hours and minutes to the current time. For example, if the current time is "Thu Apr 15 15:32:08 2004" and you wish to start at 5 PM, use 01:28 to change the start time to "Thu Apr 15 17:00:00 2004".

3. [+]<value> [hours | min] (defaults to min)

   This is an optional '+' followed by the time value, followed optionally by the keyword 'hours' or 'min'. If 'hours' is specified, <value> is taken to be the number of hours.

   Use this format to quickly add a start time in hours or minutes from the current time.

NOTE

All values are decimal.

The time specified in format 1 above is interpreted as the absolute date/time to start testing. The time specified in formats 2 and 3 are relative to the current time (i.e. the start time is set to the current time plus the value given in formats 2 or 3).

Once set, the menu is updated to reflect the date/time testing is to begin (see Screen 3-7 below). The menu option is reset to MANUAL when testing starts.

NOTE

The previous test run is **automatically "cleaned up"** when testing is started via this option.

Option 2 > Quit Time

This option from the Scheduler Setup Menu is used to set the date and time when system-level testing should end. Testing stops **after the completion of the current cycle.** The menu displays "Quit Time.....MANUAL", as shown in Screen 3-6 above, when the start time is not set. If selected and Quit Time is MANUAL, you are prompted to enter the desired time to stop testing. If selected and the Quit Time is already set, the Quit Time is reset to MANUAL (i.e. auto stop is disabled).

The time can be entered in one of three formats as shown above.

Once set, the menu is updated to reflect the date/time testing is to end (see Screen 3-7 below). The menu option is reset to MANUAL when testing stops.

Option 3 > Cycle Stop

This Scheduler Setup Menu option is used to set a limit on the number of cycles of the system-level test to run. The menu displays “Cycle Stop.....MANUAL” as shown in Screen 3-6 above when the cycle number is not set. If selected and Cycle Stop is MANUAL, you are prompted to enter the desired cycle number. If selected and Cycle Stop is already set, the Cycle Stop is reset to MANUAL (i.e. auto stop is disabled).

Once set, the menu is updated to reflect the cycle after which testing is to stop (see Screen 3-7 below).

The menu option is reset to MANUAL when testing stops.

When Cycle Stop and Quit Time are both set, testing is stopped when either event occurs.

```
================================================================================
               PowerAT - System Reliability and Acceptance Test
                           Release 9.8 - Preliminary

   Scheduler Setup    1 > Start Time~~~~~~~~SET:   Tue Aug 13 12:00:00 1991
                      2 > Quit Time~~~~~~~~~SET:   Wed Aug 14 12:00:00 1991
   <Esc> to           3 > Cycle Stop~~~~~~~~SET:   Stop After Cycle 720
   Exit Menu          4 > Timeout of Scheduler :   0  Hours, 20 Minutes
                      5 > Failures (Maximum)   :   100

   Enter Selection> []
 +------------------------------------------------------------------------------+
 | Status Summary        Tue Aug 13 09:16:58 1991    PowerAT  is  STOPPED     |
 |                                                                              |
 | Cycles:       0 Complete                Elapsed Run  Time:      0:00:00    |
 | Tests:        0 Total                   Cycle Times: Current    0:00:00    |
 |               0 Queued    0 Disabled                 Previous   0:00:00    |
 |               0 Active    0 Complete                 Average    0:00:00    |
 | Failures:     0 Total     0 Current     Start Time:                        |
 +------------------------------------------------------------------------------+

=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ====================
```

Screen 3-7. Scheduler Setup Menu - Example 2

Option 4 > Timeout of Scheduler

Option 4 from the Scheduler Setup Menu allows the operator to change the length of time that the system-level test scheduler/monitor will wait for any particular test to complete. The scheduler (schedat or schedio) will time out, stopping test cycling, if a test does not complete in the amount of time specified (20 minutes default) **since the completion of a previous test** (i.e. the time out counter is reset whenever any test completes).

When selected, this option prompts the user to enter the new time out value. Only ‘infinite’ and time formats 2 and 3 described above are valid inputs for setting the scheduler time-out value. If set to ‘infinite’ the scheduler will not stop due to a time out.

Option 5 > Failures (Maximum)

Option 5 from the Scheduler Setup Menu provides the means for setting the total test detected failures to allow before testing is stopped. When selected, this option prompts for the maximum failures to allow before stopping. The number of failures to allow or the keyword ‘infinite’ are valid inputs. If set to ‘infinite’ the scheduler will not stop due to the number of test detected failures.

Configure Menu

The Configure Menu provides options for displaying and changing the system-level test configuration. Screen 3-8 illustrates the Configure Menu. Except for the title, the display is identical for PowerIO.

```
================================================================================
               PowerAT - System Reliability and Acceptance Test
                           Release 9.8 - Preliminary

  Configure Menu     1 > Select Test Groups       6 > Error - List Errors
                     2 > Edit Config. File
  <Esc> to           3 > Auto-Configure Tests
  Exit Menu          4 > Save Config. File
                     5 > Display Config. File

  Enter Selection> []
 +------------------------------------------------------------------------+
 | Status Summary        Tue Aug 13 09:25:51 1991     PowerIO  IS  STOPPED  |
 |                                                                        |
 | Cycles:       0 Complete                Elapsed Run  Time:      0:00:00 |
 | Tests:        0 Total                   Cycle Times: Current    0:00:00 |
 |               0 Queued    0 Disabled                 Previous   0:00:00 |
 |               0 Active    0 Complete                 Average    0:00:00 |
 | Failures:     0 Total     0 Current     Start Time:                      |
 +------------------------------------------------------------------------+


=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ===================
```

Screen 3-8. Example Configure Menu

The following subsections describe the options in the Configure Menu and the actions taken when those options are selected.

Option 1 > Select Test Groups

This option from the Configure Menu is currently available under PowerIO. In PowerIO, this option displays a menu of available tests and their current state, enabled or disabled, to run the next time testing is started (see the “Select Test Groups Menu” section under “PowerIO – Input/Output Subsystem Test”).

PowerAT does not currently support the Select Test Groups option of the Configure Menu. PowerAT will respond with the following prompt line message if this option is selected:

```
Sorry, “Select Test Groups” not yet supported for PowerAT.
```

Option 2 > Edit Config. File

This option from the Configure Menu provides a means for selecting an editor and modifying the test configuration file. The modified file is a only "working" copy and is not saved once the user exits (quits) the current invocation of the system-level test (PowerAT or PowerIO). "Option 4 > Save Config. File", described below, can be used to permanently store modifications.

The PowerIO configuration file usually needs modification to customize it for a particular system configuration or device testing application (refer to "Configuring PowerIO Tests" on page 3-30). A sample PowerIO configuration file is provided in Appendix A. In most cases, the PowerAT configuration file does not need to be edited.

This option prompts the user to enter the name of the editor to use (the default editor is taken from the environment variable EDITOR if it exists; otherwise, **`/bin/vi`** is used):

```
Please enter editor [default=vi]>
```

After exiting the editor, the configuration file is validated. If errors were detected, the following attention message is displayed:

```
ERROR IN CONFIGURATION FILE DETECTED - ERRORS MUST BE CORRECTED TO RUN
```

NOTE

PowerAT currently doesn't validate its configuration file after returning from the editor.

The errors detected may be listed by selecting option 6 of the Configure Menu. The test configuration file may be listed by selecting option 5 of the Configure Menu. Help on supported tests and device models/types may be selected by issuing the command: `help test`

Option 3 > Auto-Configure Tests

As stated before, PowerAT and PowerIO will auto-configure a test configuration the first time it is run after installation. Selecte this option from the Configure Menu (Screen 3-8) if you wish to have the system-level test auto-configure at any time after the initial installation.

NOTE

For PowerAT to properly test a system, the Auto-Configure Tests option must be run after a system CPU, memory, or disk subsystem configuration change has been made. Upon invocation or attempted test run, the following attention message is displayed if PowerAT detects a difference in the system hardware configuration:

```
NEW CPU/MEM/DISK CONFIGURATION DETECTED. AUTO-CONFIGURE TESTS BEFORE RUNNING.
```

If a system hardware configuration change is detected, PowerAT will not allow running the test until the Auto-Configure Tests option is selected.

When the Auto-Configure Tests option is selected, you are prompted to enter ‘yes’ to confirm the re-configuration:

```
Auto-Configure Tests   - Are you sure? (enter "yes" to confirm) >
```

If the response is ‘yes’, the system-level test configuration is auto-configured and the existing configuration and working configuration files are overwritten. If a user specified configuration file was being used, it is closed without modification.

Option 4 > Save Config. File

All modifications done to a system-level test configuration file from the Configure Menu (i.e. via “Option 2 > Edit Config. File”) are stored in a temporary work file. The configuration in the work file is only kept while the system-level test is active (Quit has not been selected from the Main Menu). To save a particular configuration for future runs, select this option. You are prompted to enter ‘yes’ to confirm the save:

```
Save Config. File   - Are you sure? (enter "yes" to confirm) >
```

Entering ‘yes’ causes the working configuration to be saved to a permanent configuration file, or to the user specified configuration file if one was given on the invocation.

Option 5 > Display Config. File

If selected, this option of the Configure Menu displays the current working configuration file.

Option 6 > Error - List Errors

If selected, this option of the Configure Menu displays any errors detected in the configuration file.

NOTE

PowerAT does not currently support error checking of the test configuration file. PowerAT will always report “No Errors Found In Config. File: PowerAT.conf/Work_file”, when this option is selected.

PowerIO – Input/Output Subsystem Test

The I/O Subsystem test verifies the system I/O configuration and functionality of I/O controllers and attached peripherals. The PowerIO test environment is configurable, allowing testing of one or more devices/controllers on the primary and/or secondary I/O subsystems. Configuring a single test module provides an online test of a particular device/controller. Concurrent execution of test modules provides verification of I/O backplane arbitration. PowerIO is designed to run independent of (i.e. not concurrent with) PowerAT.

PowerIO Startup

The PowerIO system-level I/O subsystem test environment is found in the **/usr/d/system** directory if the diagnostic products tape has been installed. To start PowerIO, change directory (**cd**) to **/usr/d/system** and enter 'powerio'. The command format is:

```
powerio [-h] [-f <user specified configuration file>]
```

The **-h** option will print a help message. The optional **<user specified configuration file>** argument may be used to override the use of the default PowerIO test configuration file. See the next section for information about the PowerIO test configuration file format.

Configuring PowerIO Tests

The first step in getting PowerIO ready to run on a system is to configure the tests to be run. The PowerIO test configuration is a file containing information about the tests to be scheduled for execution. The Configure Menu provides the interface for displaying and changing the I/O subsystem test configuration.

A skeleton test configuration file is created by PowerIO on the first invocation after installation or by selecting the Auto-Configure Tests option (3) of the Configure Menu. The auto-configured file lists tests for each supported device found by PowerIO on the system (e.g. disk drives, tape drives, and I/O ports). See Appendix A for a sample PowerIO test configuration file.

Only the disk tests are placed in the configuration file un-commented. Port turnaround and tape tests are placed in the configuration file but are commented out. Terminal tests are configured for each asynchronous port with a **ttymon(1M)** running on it (also commented out). Tests for any asynchronous ports which have printers attached must be added by the user – normally by editing the commented port test configuration file line for the respective port number. Tests for other supported devices found are placed, commented out, in the configuration file.

Edit the configuration file to customize it for a particular system device configuration (See "Option 2 > Edit Config. File" on the Configure Menu).

The configuration file consists of one entry per line (excluding blank and commented lines) for each I/O test to be configured. Each entry is made up of at least three blank (space) separated fields.

The format for each tape and port entry is as follows:

```
<name> <type> <device logical unit> [<optional>...]
```

The format for each disk test is as follows:

```
Bdisk <type> <unit> <mounted... >
```

<name> is a mnemonic for the I/O device test to be run. The test name is currently one of the following (described in the "PowerIO Test Descriptions" section of this manual):

- disk - Disk test
- port - RS-232 port test
- cart - 1/4" cartridge, 4mm DAT, 8mm Exabyte tape test
- tape - 1/2 inch (9-track) tape test
- term - CRT terminal test
- prnt - Printer test
- rtck - Real Time Clock test
- edge - Edge Triggered Interrupt test
- lan - Local Area Network test
- hsde - HSDE controller loop back test
- dr11 - DR11W controller loop back test

<type> is a model or vendor name designation which further qualifies the device under test. In some cases the model/type field may be a device type designation such as "132" for a generic (unknown model number) 132 column printer.

The **<name>** and **<type>** fields of a configuration file entry are used to form the executable test module name using the following convention:

<name>.<type>

For example, the PowerIO 1/2 inch tape device test module name is 'tape.rmt' (type 'rmt' – **r**aw **m**agnetic **t**ape device). Executable PowerIO test modules are kept in the **/usr/d/system/test** directory. The currently supported model/types are described in the "PowerIO Test Descriptions" section of this manual.

The specification and range of **<device logical unit>** depends on the device to be tested. It can be the device or controller number or the device special file name. For example:

- The first argument for a disk test is the disk device minor number. This argument is not used by the test, but is used by the scheduler to build unique test result filenames.
- The first argument to a test for a device connected to an RS-232 asynchronous port is the tty device name; for example, "/dev/tty0_01".

For disk tests the **<mounted... >** argument corresponds to file systems which have been mounted from the disk under test. Examples are "/dev/root", "/dev/usr" and "/dev/var". Multiple mount arguments may be entered.

Up to nine additional **<optional>** parameters are test dependent. For example, an optional baud rate parameter for term, port and prnt tests; or the optional speed and density parameters for the tape test. Optional parameters must be placed on the configuration file line *after* the device logical unit specification of the test.

NOTE

For documentation purposes, an optional descriptive word followed by an equal sign (=) may immediately precede the argument(s).

An example of a PowerIO configuration is shown in Appendix A.

Select Test Groups Menu

Option 1 of the Configure Menu selects the Select Test Groups menu. This menu can be used to select or de-select the execution of a group of tests given by the test name as defined above. A menu of possible test names is displayed with their current state, Enabled or Disabled. Screen 3-9 is an example of the Select Test Groups menu.

```
================================================================================
                     PowerIO - Input/Output Subsystem Test
                           Release 9.8 - Preliminary

   Select Test Groups 1 > Disk..........Enabled   6 > Prnt..........Disabled
                      2 > Term.........Disabled   7 > Rtck..........Disabled
   <Esc> to           3 > Port.........Disabled   8 > Edge..........Disabled
   Exit Menu          4 > Tape.........Disabled   N > Next Screen
                      5 > Cart.........Disabled   L > Last Screen

   Enter Selection>
  +------------------------------------------------------------------------+
  | Status Summary        Mon Mar 15 15:09:53 2004    PowerIO  is  STOPPED   |
  |                                                                        |
  | Cycles:       0 Complete                 Elapsed Run  Time:      0:00:00 |
  | Tests:        0 Total                    Cycle Times: Current    0:00:00 |
  |               0 Queued    0 Disabled                 Previous   0:00:00 |
  |               0 Active    0 Complete                 Average    0:00:00 |
  | Failures:     0 Total     0 Current      Start Time:                     |
  +------------------------------------------------------------------------+


=================== abbu: 6.1-20040127 6.1 Synergy mpc7455 ===================
```

Screen 3-9. Example Select Test Groups Menu

Tests are enabled as they are added to the configuration file (see the Configure Menu subsection “Option 2 > Edit Config. File” on page 3-28). Selecting the corresponding option number toggles the state between enabled and disabled.

An enabled test may be disabled by selecting the option number corresponding to the test. Disabling a test will cause all of the configured tests with that name not to be scheduled the next time the Run - Start Test option (1) is selected in the Main Menu.

A disabled test may be enabled only if there is at least one device test of that name in the configuration file.

This menu provides the means of quickly disabling all tests on a certain type of device (e.g. all port tests). The states selected in this menu are *not* saved once PowerIO is exited.

The Next Screen and Last Screen options display additional tests which are not shown.

Running PowerIO

The PowerIO tests are started either by selecting the Run/Start Test option (1) of the Main Menu, use of the ‘run’ command, or by automatic startup programmed in the Scheduler Setup menu.

When started other than by auto-startup, PowerIO displays a table of configured tests. For each test in the configuration file, PowerIO lists the model number, the unit number and a short comment. If the comment is “Test Scheduled”, that test will be scheduled and run by schedio. Any other comment is surrounded by double asterisks (**) and is a short explanation why the particular test is not able to be scheduled. Following this table is a summary of tests scheduled, followed by a prompt. Screen 3-10 is an example of the table and prompt.

```
=========================================================================
                    Test Schedule Summary
test  model  unit  comment          | test  model  unit  comment
===== ===== =====  ================ | ===== ===== =====  ================
disk  rdsk  12800  Test Scheduled   |
=========================================================================

                     Test Schedule Summary

               Total I/O Device Tests Scheduled = 1

                 Test Name  Configured  Scheduled
                 =========  ==========  =========
                 disk            1           1

=========================================================================
Enter "go"    to  start  testing or
      "abort" to cancel the startup > []
```

Screen 3-10. Example PowerIO Startup Test Schedule Summary

If ‘go’ is entered, the PowerIO test scheduler (schedio) is started. The message “Initializing Test Scheduler...” appears until schedio is actually ready to start invoking tests. PowerIO then returns to the test RUNNING Main Menu (unless no tests in the configuration file were scheduled). If none of the tests configured are able to be scheduled, the test run is canceled and PowerIO returns to the test STOPPED Main Menu and the following attention message is displayed:

```
NO TESTS WERE SCHEDULED - CHECK THE SCHEDULED RUN LIST IN CONFIGURE MENU!
```

If ‘abort’ is entered, the test run is canceled and PowerIO returns to the test STOPPED Main Menu and the following attention message is displayed:

```
NO TESTS WERE SCHEDULED........TEST START UP ABORTED BY OPERATOR!
```

Any other input at this prompt causes PowerIO to re-display the scheduled test table.

Schedio, when invoked, will start up concurrent execution of as many scheduled tests as the operating system will allow. The number of tests which may be run concurrently depends on the size of each test, available system swap space, and the maximum number of processes allowed.

PowerIO requires that all non-root users logoff before any terminal tests are scheduled.

NOTE

Response time (including user input echoing) may be impacted during test execution. The extent of the impact will depend on the number of tests configured and scheduled to run.

When the on screen clock is ticking every second, response time to input will probably be more favorable.

NOTE

PowerIO issues **`pmadm(1M)`** commands in order to stop ttymon and login processes. Prior to issuing these commands, a restoration script is built and copied to **`/etc/restore_ttymons`**. If the system crashes while PowerIO is RUNNING and terminals under test do not display login prompts after rebooting, this file may be executed if it exists (it should exist if PowerIO was running when the system crashed). PowerIO does not attempt to stop ttymon processes initiated from **`/etc/inittab`**, as this can cause failures of the PowerIO terminal and port tests.

PowerIO Test Descriptions

The following subsections describe the PowerIO tests.

Disk - Disk Test

The disk test verifies I/O to disk devices in the following manner: For each file system passed to the test, create a directory and write/read a file (disks only). Perform incremental seek/reads across each (raw) partition.

Models Supported

All disks and disk controllers supported by the operating system are also supported in PowerIO. The model/type name used is ‘rdsk’. The operating system gives the test access to necessary drive specific information.

Device Logical Unit

The first argument for a disk test is the disk device minor number. This argument is not used by the test, but is used by the scheduler to build unique test result filenames.

Mount Argument

The ‘mount’ argument corresponds to file systems which have been mounted from the disk under test. Examples are “/dev/root”, “/dev/usr” and “/dev/var”.

Cart - Cartridge Tape Test

This section describes the cartridge tape device test. Cartridge tape devices are tested by:

1. Writing/reading/comparing multiple tape records.
2. Verifying file and record positioning commands including:
   - Write End Of File, forward and backspace file
   - Forward and backspace record
   - Rewind and re-tension

NOTE

The cartridge tape test ‘cart’ may continue execution after PowerIO testing is stopped by the user. This will cause cartridge tape test errors in subsequent runs of PowerIO, provided the cartridge tape test process is active from a previous run. To avoid errors of this nature, ensure that the ‘cart’ tape test process has exited before running the test again. Use the **`ps(1)`** command to look for an active cartridge tape test in the process table.

Models Supported

All cartridge tape devices and tape controllers supported by the operating system are also supported in PowerIO. The model name used for the cart test is ‘6140’. The cart test will also work if the model/type specification is ‘rmt’.

Device Logical Unit

Valid device logical units for the tape test are the decimal integers zero (0) through thirty-one (31). The tape drives under test must be defined in the node file for the generic tape driver, **`/etc/conf/node.d/gt`**.

PowerIO verifies the logical units against the node number in the node file. PowerIO expects to see lines in this file such as:

Name	Node	Type	Adapter	Adapter#	Ctlr	Dev
gt	mt/<unit >	T	hsa	0	5	0
gt	mt/<unit>	TS	hsa	0	4	0

where <unit> is the logical unit number (0-31) that PowerIO expects to match.

Optional Parameters

One optional parameter may be specified for the cart test. The optional parameter is specified in the configuration file *after* the device logical unit:

Option	Usage	Description
Re-tension	re-tension	If this option is present, the cart test will re-tension the cartridge tape on the first cycle and every tenth cycle afterwards (cycle 11, 21, etc.). This option is supplied in the default configuration file.

Async Device Tests (term, port, and prnt)

The PowerIO tests which test async devices are as follows:

- term – CRT Terminal Test. The term test verifies terminal devices as follows:
  - Invoke CRT terminal device self-test if one exists.
  - Write/read/compare of a rotating ("barber pole") pattern to the tty port of the terminal under test.

NOTE

The read/compare portion of the test depends on the terminal device's capability to send, on command, data written to it.

  - Write **BEGIN** and *END DATA PATTERN* messages that indicate the tty line number of the terminal under test.
- port – Async Port Turnaround Test. The port test performs a write/read/compare of 1 through 256 bytes of data through a turnaround plug, part number 720-1830189-901. The port test requires a turnaround plug installed on the port(s) under test.

NOTE

Installation of turnaround plugs will cause active ttymons to respond to their own login prompts as user login attempts. **`ttymon(1M)`** processes (the process which waits for user logins) must be stopped on the async I/O ports prior to installing turnaround plugs. To do so, issue the appropriate **`pmadm(1M)`** commands.

- prnt – Printer Test. The printer test writes an 80 or 132 character rotating ("barber pole") pattern to the tty port of printer under test. The prnt (printer) test requires visual inspection of the printer output for full confirmation of successful test completion.

  The prnt test begins by writing the following message to the printer:

```
BEGIN TEST DATA PATTERN (TTY LINE xx)
```

  where *xx* is the port/line number to which the printer is attached. The prnt test continues by

writing multiple lines of a rotating printable ASCII character pattern to the printer, then concludes by writing the following message:

```
END OF TEST DATA PATTERN (TTY LINE xx)
```

If the user stops PowerIO while the prnt test is active, the following message is written to the printer:

```
TEST TERMINATED BY OPERATOR (TTY LINE xx)
```

The device logical unit and optional parameter specifications are the same for each of these tests.

PowerIO makes two assumptions regarding port monitor configuration. Failures can occur during the asynchronous device tests if either of these assumptions are incorrect:

1. PowerIO assumes the path name of the asynchronous port is of the form "/dev/ttyX_XX".
2. PowerIO requires that hardware controls for the port specify "Hang-up on last close". This is typically done via the "-l <ttylabel>" option on the **`ttyadm(1M)`** command. For example, "-l 9600H" specifies 9600 baud with hang-up on close.

Use the **`pmadm -l`** command to see if your system has its port monitors set up properly for PowerIO to auto-configure and run terminal and port tests. A sample "pmadm -l" output is as follows:

```
PMTAG           PMTYPE       SVCTAG       FLGS    ID      SCHEME   <PMSPECIFIC>
ttymon1         ttymon       02           u       -       login /dev/tty0_02
bhr 0 /usr/bin/shserv 60 9600H ldterm tty0_02: - l       break - #
ttymon1         ttymon       00           u       -       login /dev/tty0_00
bhr 0 /usr/bin/shserv 60 9600H ldterm guardian: - l break - #
ttymon1         ttymon       01           u       -       login /dev/tty0_01
bhr 0 /usr/bin/shserv 60 9600H ldterm colossus: - l break - #
```

The example above shows three asynchronous ports, with the correct naming convention PowerIO requires (/dev/ttyX_XX) with 9600H for "Hang-up on close".

See the man page for **`pmadm(1M)`** and **`ttyadm(1M)`** for a complete description of how to set up asynchronous port monitors.

PowerIO requires that all non-root users log off before any terminal tests are scheduled.

Models Supported

Async device tests support devices connected to the High Performance Serial communications controller (HPS). The device model/type names for each are shown in Table 3-4.

Table 3-4. Supported Models – Async Device Test

Test	Model/Type	Description
term	8665	ASCII Alphanumeric CRT (wyse 50)
	8691	ANSI Mono Graphics CRT (wyse95)
	tty	Generic ASCII Terminal (See Note 1)
port	232	RS232 Port Turnaround Test (See Note 2)
prnt	8760	55 CPS Letter Quality Printer.
	8761	80 CPS Letter Quality Printer.
	4420	240 CPS Serial Matrix Printer
	4327	300 LPM Band Printer
	4337	600 LPM Band Printer
	1230	12-PPM laser Printer
	2655	26-PPM Laser Printer (Diablo 639 Mode)
	4508	8 LPM Laser Printer
	4430	300 LPM Matrix Line Printer
	4440	600 LPM Matrix Line Printer
	80	Generic 80 Column Printer (See Note 3)
	132	Generic 132 Column Printer (See Note 3)

Notes:

1. The term test type ‘tty’ assumes a dumb terminal and merely writes the data pattern. Visual inspection of the data is required for verification of correct operation.
2. Port turnaround tests require the installation of a turnaround plug.
3. Prnt test types ‘80’ and ‘132’ may be used to force printing of 80 column or 132 column data pattern, respectively.

Device Logical Unit

The async tests use the operating system general terminal interface. The device logical unit is a four character tty line number (i.e. the last four characters of the /dev/ttyX_XX special file name for the port) as outlined below:

The async device test’s device logical unit is two characters in the form **B_PP**, where **B** represents a hexadecimal value for the logical I/O Board (controller) number:

Controller (HPS) 0 through 15

and **PP** represents the port number (in decimal):

00-16 Port 0 through 15, port 16 is the printer port

For example **1_15** specifies logical I/O controller 1, port 15.

Optional Parameters

One optional parameter may be specified for the async tests. The optional parameter is specified in the configuration file *after* the device logical unit.

Option	Usage	Description	Default
Baud Rate	Baud=*xxxx*	Select port baud rate where *xxxx* is a supported baud rate for the async port. Valid values for baud are defined in the **`stty(1)`** man page.	Baud=9600

Rtck - Real-Time Clock Test

The Real-Time Clock test may be run on each configured real-time clock on the system. These tests assure the clocks are operational, as well as provide additional I/O interrupt interaction. See the **`rtc(7)`** man pages for more detail on real-time clocks in general.

If no optional parameters are used, this test does the following:

- Sets the clock resolution to 10 milliseconds (MSEC10).
- Sets the clock count value to 10,000 (0x2710).
- Sets the clock mode to count down repetitively (RTC_DEFAULT | RTC_REPEAT).
- Starts the clock counting.
- Waits 1,000 times for the clock to reach zero (1,000 “ticks”).
- Stops the clock and exits the test.

NOTE

The test waits until the defined number of clock interrupts are detected. Due to programming overhead and system load, some clock interrupts may be missed.

WARNING

Using the optional parameters described below, it is possible to set the clock’s resolution and count so that the majority of system time is spent handling the excessive number of clock interrupts.

Models Supported

All real time clocks supported by the operating system are also supported in PowerIO. The model/type name used is ‘rrtc’ for NightHawk and Power MAXION systems and ‘mrtc’ for Power Hawk systems. Clock specific information is ascertained from the operating system.

Device Logical Unit

On NightHawk and Power MAXION systems, the real-time clocks are integral to the CPU boards. All processors on a board may be interrupted by the real-time clocks on that board.

The rtc device unit has the form ***c#**. The character * is the controller number which may be 0, 1, 2, or 3 and corresponds to the CPU board on which the clock resides. The # character is the minor number and refers to a given rtc, which may be 0–4.

Optional Parameters

Option	Usage	Description	Default
resolution	-r<res>	The resolution per clock count: 0) MSEC10, 1) USEC, 2) USEC10, 3) USEC100, 4) MSEC	PowerIO cycle modulus 5
count	-n<count>	Clock count value	60,000/resolution (in usecs)
ticks	-t<ticks>	The number of times the test will detect the clock reaching zero.	1000
trace	-d<trace>	debug trace level (0-9)	0
count up	-u	Set the mode to count up instead of down.	count down

Edge - Edge Triggered Interrupt Test

One Edge Triggered Interrupt (ETI) can be tested with each edge test. Each ETI is tested by generating software requested interrupts on the ETI with the ETI enabled and disabled.

See the **`eti(7)`** man page for general information about edge triggered interrupt devices.

Models Supported

The only model/type name used for this test is ‘reti’.

Device Logical Unit

The decimal unit number specified corresponds to the ETI number under test.

The device logical unit range depends on the number of interrupts supported on the system. See the **`eti(7)`** man page for this number.

Optional Parameters

Option	Usage	Description	Default
loop count	-l<loop count>	Loop through test <loop count> times. This option increases the number of times the edge triggered interrupt is generated, thus increasing the per cycle interrupt activity.	2

Hsde - HSD(E) to HSD(E) Loop Back Test

The High Speed Data (HSD) Enhanced (E) channel interface test is designed to verify communication between two HSD(E) boards as well as verify (H)VME data transfers.

Each board, in turn, is put into either master or slave mode and data is transferred in byte (8-bit), word (16-bit), and long word (32-bit) sizes. The slave verifies data sent by the master before sending it back to the master. The master then verifies the data sent by the slave.

The test requires the following setup:

- Two HSD(E) boards, strapped for proper operation.
- Two 50-pin flat ribbon cables (part number 720-1820031-901).

On HSDE-HSDE board pairs, attach the cables with connector JP1 cabled to JP1, and connector JP2 cabled to JP2.

Models Supported

The HSDE channel interface from Concurrent Computer Corporation is the only model supported.

The model/type name used for this test is ‘chnl’.

Device Logical Unit

The logical device numbers of the HSDE pair under test are given in decimal separated by a colon.

The first device specified will operate as master first and slave last. The second device specified will operate as slave first and master last.

Optional Parameters

The following optional parameters can be specified for the hsde test.

Option	Count	Description	Default
byte count	-b<bytes>	Channel interface transfer byte count. This value must be divisible by 4 and less than 262140.	65536
timeout	-w<secs>	Watchdog time-out value in seconds.	15
loop count	-l<count>	Test loop count. A large loop count may cause PowerIO test scheduler to time-out.	1

Dr11 - DR11W Emulator Loop Back Test

This test verifies “programmed” I/O and DMA features of the DR11W emulator board by “looping back” the two connectors with a 40-pin ribbon cable. The test requires one 40-pin ribbon cable per board (part number 720-1820030-901) connected to DR11-W board connector J1 to J2.

Models Supported

The DR11-W emulator from Ikon Corporation is supported.

The model/type name used for this test is ‘chnl’.

Device Logical Unit

This is a decimal integer in the range 0 through 7 (i.e. up to 8 DR11-W boards may be configured and tested at one time).

Optional Parameters

None.

LAN - Local Area Network Test

The Local Area Network test may be run on each configured network controller on the system. The test verifies that the network controllers are operational and provides additional I/O interaction. The LAN tests are based on the IEEE 802.3 Link-Level communications TEST packet, which when received by an IEEE 802.3 compliant interface, is required to be returned to the sender. Link-Level communications are supported by the Ethernet/FDDI Data Link Provider Interface (DLPI).

If no optional parameters are used, this test sends and receives 1500 iterations of ten different packet sizes to the remote host and checks the packets that are returned.

NOTE

The test waits until the defined number of packets have been sent and received before exiting. High network and system load can cause the test to run longer than allowed by the default settings. For this reason, it is recommended that the lan test be run only on low traffic networks.

Models Supported

All Local Area Network controllers supported by the Ethernet/FDDI Data Link Provider Interface are supported by the test.

Device Logical Unit

The device logical unit names are of the form *xxn*, where *xx* is the controller name which is displayed when the system is initializing, and *n* is the controller number, 0 for the first, 1 for the second, etc. Some controllers support multiple network interfaces (ports), and are therefore designated by *xxnm* where *m* is the port number on controller *n*. Standard interface names are as follows:

- sym – Symbios Embedded Ethernet
- gte – Galileo/Marvell Discovery Embedded Ethernet
- mve – Marvell Discovery II/III Embedded Ethernet
- ie – Integral Ethernet
- dec – DEC embedded Ethernet
- egl – Interphase Eagle Ethernet
- cnd – Interphase Condor Ethernet
- pg – Interphase Peregrine FDDI

Remote Host Specification

The remote host must be specified in the configuration file entry for the lan test following the device specification (i.e. egl0, pg0).

The remote host must be specified in one of the following ways:

Option	Usage	Description
hostname	<hostname>	The hostname of the network device to use as the remote host for the duration of the test. The hostname must exist as a valid entry in the system hosts file. The hostname can only be specified as the *last* parameter in the configuration file entry.
internet address	<inet addr>	The internet address of the network device to use as the remote host for the duration of the test. The internet address must be specified in dot format, i.e. 129.134.1.1. The internet address can only be specified as the *last* parameter in the configuration file entry.
physical address	-a <phys addr>	The 6-byte physical network address of the network device to use as the remote host for the duration of the test. The physical address must be specified as a series of six hexadecimal values from 0 to ff, separated by whitespace, i.e. -a 00 00 c3 01 02 03

Optional Parameters

Optional parameters may be specified for the lan test. If an optional parameter requires an argument, the option name and argument must be separated by white space. All options must follow the device specification (i.e. egl0, pg0).

Option	Usage	Description	Default
buffer checking	-c	Disables buffer checking for corrupted data. Each test command packet sent contains an information buffer 100 to 1000 bytes in size. The test command response packet is checked to verify that its information buffer is identical to the original packet sent.	enabled
iterations	-i <iterations>	The number of iterations the test will perform. Each iteration will write ten test command packets of varying sizes to the remote host, and read each test command packet response returned by the remote host. The default value will cause 15,000 packets to be sent and received by the local host. On a busy network, the number of iterations may need to be reduced to shorten the run time of the test.	1500
retry threshold	-r <retries>	The number of retries allowed per 10,000 packets sent. On a busy network, the number of retries allowed may need to be increased to allow for a greater frequency of packet collisions.	50
read timeout	-t <timeout>	The time in seconds before a read of the remote host's response will time-out, causing the local host to retry the write of the test command packet. Large numbers of retries will cause many read time-outs and increased test run time due to the overhead of the timeouts.	5

PowerIO Generic Test Interface

This section describes the method of adding new test modules to PowerIO. The requirements and restrictions for creating and installing test software to be run under PowerIO are described in the following subsections.

Naming Convention

Test module names under PowerIO use the following convention:

<name>.<type>

<name> is any name except as restricted (see "Restrictions" below) – normally a mnemonic describing the device or controller type under test. **<type>** is a model or vendor name designation which further qualifies the device under test. For example, the PowerIO tape device test module name is 'tape.rmt' (type 'rmt' - **r**aw **m**agnetic **t**ape device).

A maximum of 16 characters, including the period, may be used in constructing the test module name.

NOTE

The **<name>** and **<type>** fields of a configuration file entry are used to form the executable test module name.

Executable PowerIO test modules are kept in the **/usr/d/system/test** directory. The currently supported model/types are described in the "PowerIO Test Descriptions" section of this chapter.

Test Arguments

PowerIO requires that the first argument passed to a test module designates the device unit under test. The first argument corresponds to the **<device logical unit>** field of the PowerIO test configuration file described below. This may be the device or controller number or the device special file name.

For example:

- The first argument for a disk test is the disk device minor number. This argument is not used by the test. It is used by the scheduler to build unique test result file names.
- The first argument to a test for a device connected to an RS-232 asynchronous port is the tty device name, for example, "/dev/tty0_01".
- The Console remote port is given as "/dev/remote".

PowerIO supports up to 10 command line arguments, including the required device specification.

The 9 additional command line arguments are optional. These correspond to the optional arguments provided for in the PowerIO configuration file.

Test Input

Input to the test module must be from **stdin** (file descriptor number 0). If a test input file exists, PowerIO will open it as **stdin** on behalf of the test module. The test module input file must be named as follows:

<name>.<type>.d

where **<name>** and **<type>** are as defined above under "Naming Convention". The Diagnostic Output of a test must go to **stdout** (file descriptor number 1) and/or **stderr** (file descriptor number 2). **stdout** and **stderr** are both re-directed to a PowerIO result directory file (e.g. **PowerIO.rslt/tape.6140.0.err**). Typically, output is restricted to failure description but optional test execution trace output is often useful. Upon failure, the result file is copied into a chronological error log and used as the test failure description.

Test Module Error Detection

PowerIO test modules are designed to conform with UNIX command diagnostic conventions (i.e. they accept command line arguments and return conventional exit status – see the **intro(1)** man page).

Test modules must use the **exit(2)** system call to terminate execution. Upon successful test completion, exit is called with zero (0) as the argument. If a failure is detected by the test, exit is called with a non-zero value between one (1) and 255.

PowerIO recognizes a test module detected error if the test module exits with a non-zero status. The output written to **stdout** and **stderr** is saved to the PowerIO error log when a test exits due to failure.

Configuration

Configuring a PowerIO test module is described above in the "Configuring PowerIO Tests" section of this chapter.

Test Module Installation

The PowerIO system-level diagnostic is included in the diagnostic product tape and CD. The system-level diagnostics are installed under the **/usr/d/system** directory.

Test module executable files must be installed in the **/usr/d/system/test** directory.

Test module input data files must be installed in the **/usr/d/system/data** directory.

To install additional PowerIO test modules, a **cpio(1)** format tape must be provided so that the user would, after installing the diagnostic products tape, move to the **/usr/d/system** directory, and issue the **cpio(1)** command:

```
cpio -icBvmud < /dev/rmt/0ms
```

This implies that the additional test tape contains the test module file on tape as follows:

```
test/<name>.<type>
```

Restrictions

The test module name prefix (e.g. **<name>** as defined above under “Naming Convention”) must not match those already defined in PowerIO. The following is a list of the test name prefixes currently reserved by PowerIO:

Name	Description	Name	Description
disk	disk drive test	lan	Local Area Network test
cart	cartridge tape test	term	CRT terminal test
port	Asynchronous I/O port test	prnt	Printer test
rtck	Real Time Clock test	edge	Edge Triggered Interrupt test
hsde	HSDE Interface test	dr11	DR11-W Emulator test

CAVEAT

Although the test modules are installed in the **/usr/d/system/test** directory, PowerIO runs the test module from the **/usr/d/system** directory. Therefore, file accesses from within the test module are relative to **/usr/d/system**.

PowerAT – System Reliability and Acceptance Test

PowerAT is the system reliability and acceptance test found under the **/usr/d/system** directory if the diagnostic product has been installed.

Verification of system processor hardware functions is the emphasis of PowerAT. The system hardware that is tested includes:

- all CPUs
- Memory subsystem(s)
- I/O interface(s). I/O contention for the memory backplane is generated by scheduling concurrent execution of a disk test module with CPU and memory intensive test modules.

Testing is accomplished through concurrent execution of CPU, floating point co-processor, memory, and disk I/O test modules. Test module execution is governed by the PowerAT test environment scheduler (schedat) based on available process space and virtual memory.

PowerAT is intended to be run on a system with no other users logged in. PowerAT should be started from a terminal other than the Console Processor terminal. The Console terminal receives system error messages as they occur. It is useful to leave the Console terminal available to view these messages. The Console terminal also serves as a “safety valve” in case the operator’s terminal should hang for any reason. Logins on all terminal ports except the operator’s terminal and the Console terminal are disabled when PowerAT is scheduling tests (RUNNING).

PowerAT Startup

The operator must be logged in as “root.” To start PowerAT, move to the diagnostic directory, **/usr/d/system**, and enter ‘powerat’. The command format is:

```
powerat [-h] [-u] [-f <user specified test configuration file>]
```

Testing may not be started while users are logged in unless the **-u** option is given on the PowerAT invocation, or the ‘usrcheck’ generic command is used to defeat this restriction.

The **-f** option is used to specify a non-default test configuration. This option is *not* recommended for normal use.

Configuring PowerAT Tests

PowerAT automatically configures tests to be run the first time it is invoked after installation. The configuration of tests depends on the system hardware configuration:

- One set of tests is run on each CPU.
- A disk interaction test is scheduled for each disk device found on the system.
- Memory intensive tests are scheduled based on the system memory configuration.

NOTE

For PowerAT to properly test a system, the Auto-Configure Tests option of the Configure Menu must be run after a system CPU, FPP, memory, or disk subsystem configuration change has been made.

The configuration file consists of one line per test in the following format:

```
<test name> <test type> <unit spec> <mount ...>
```

<test name> One of the names listed in the “PowerAT Test Descriptions” section.

<test type> Either ‘cpu’, ‘fpp’, ‘dsk’, or ‘mem’ indicating a copy of the test is run per CPU, Floating Point Processor, Disk Device, or each 4 MB of physical memory, respectively.

<unit spec> For test types ‘cpu’, ‘fpp’, or ‘mem’ the decimal representation of the CPU bias mask indicating on which CPUs the test is allowed to execute.

For test type ‘dsk’, number ranges between 0–65535 and is derived from the device major and minor number. This argument is not used by the scheduler to build unique test result file names.

<mount> For disk tests only, this argument corresponds to file systems which have been mounted from the disk under test. Examples are “/dev/root”, “/dev/usr” and “/dev/var”. Multiple mount arguments may be specified.

As stated in the last section, PowerAT provides the **`-f`** option to allow the user to specify a non-default test configuration (NOT recommended for normal use). This option could be useful as a troubleshooting tool:

1. Copy the standard PowerAT configuration file to a temporary configuration file:

   # **`cp PowerAT.conf/Config_file my_config`**

2. Edit the temporary configuration file adding, deleting or rearranging the list of configured tests.

3. Run PowerAT using the modified test configuration:

   # **`powerat -f my_config`**

Running PowerAT

The PowerAT tests are started either by selecting the Run–Start Test option (1) of the Main Menu, using the ‘run’ command, or by automatic startup programmed in the Scheduler Setup menu.

PowerAT will not allow test startup if there are users logged in on other terminals (unless invoked with the **`-u`** option, or the ‘usrcheck’ command is given).

Schedat, when invoked, will start up concurrent execution of as many tests as the operating system will allow. The number of tests that can be run concurrently depends on the size of each test, available system swap space, and the maximum number of processes allowed.

NOTE

Response time is severely impacted during the first few minutes of each test cycle. All key strokes are read as they are entered but PowerAT response to input may take 30-60 seconds during certain stages of a test cycle. Input echoing is also impacted. When the on screen clock is ticking every second, response time to input will probably be more favorable.

On systems with processor “local” memories, **PowerAT automatically runs all test processes with “local” memory binding.**

PowerAT Test Descriptions

This section gives a list of PowerAT test module names and their titles.

- Standard (CPU) Tests – These tests are configured to run on each active CPU on the system.
  - cpu1 and cpu2 – Integer arithmetic and data type tests
  - page1 and page2 – Demand paging tests

  - uenv1 and uenv2 – User environment tests: file and string manipulation; integer, character and string data type formatting
  - file – Multiple file manipulation test
  - cvtbas – Integer base conversion test

- Disk subsystem test – One disk test is configured for each disk device
  - disk – Disk subsystem tests
- Memory intensive tests – The number of vm tests configured depends on the amount of available global and processor resident (local) physical memory
  - vm2 – Two megabyte virtual memory allocation test
  - vm4 – Four megabyte virtual memory allocation test
- Floating Point Tests – These tests are configured to run on each CPU
  - arcomb – Arithmetic combinations tests
  - except – Floating point exception test
  - fncast – Function casting test
  - mathop – Mathematical operator test
  - mixmod – Mixed mode arithmetic test
  - oprcom – Combinations of operator test
  - precic – Floating point data type precision (in ‘C’)
  - precif – Floating point data type precision (in ‘f77’)
  - ranges – Valid floating data type range test
  - resolu – Resolution of data type test
  - trigfn – Trigonometric function test
  - opratr – Operator test
  - costst – Cosine test
  - logtst – Natural logarithm test
  - tantst – Tangent test
  - divsqr – Divide and square root test
  - matrix – Matrix multiplication test
  - logten – Common logarithm test
  - trgcom – Trigonometric function combinations test
  - square – Square root test
  - fppsgl – Single precision floating point arithmetic test
  - fppdbl – Double precision floating point arithmetic test

A
PowerIO Test Configuration File

A sample PowerIO Test Configuration file is given below. Refer to "Configuring PowerIO Tests" on page 3-30 for instructions on editing this file to reflect a particular system device configuration.

```
#
#          PowerIO Test Configuration File
#
# The pound character '#' denotes a comment.
# All text from '#' to the end of line is ignored.
#

#
# CPU Board Tests
#
# Uncomment only those devices that exist on your system.
#
#port 232 /dev/tty1 4800        # embedded UART

#/dev/rrtc/0c0 configured as HRT callout queue RTC
#rtck mrtc /dev/rrtc/0c1        # MPIC Timer 1
#rtck mrtc /dev/rrtc/0c2        # MPIC Timer 2

#
# Test(s) for NCR885 SCSI 0
#
# Note: change entries for 6750 tape drives to:
#       tape rmt <unit> density=6250 speed=slow
#
disk rdsk 4096  /dev/dsk/0s4
disk rdsk 4352  /dev/root /dev/var /dev/usr

#
# Test(s) for Local Area Network
#
# NOTE: A local network host must be specified as follows:
# lan ether <interface> [options] <host>   where <host> is:
#  1) a 6-byte physical address, i.e. -a 00 00 c3 01 02 03
#  2) an Internet address in dot format, i.e. 129.134.78.1
#  3) a system name in the hosts file, i.e. systemX
#
#lan ether <interface> [options] <host> # Example Line
#lan ether /dev/sym0                    # Symbios Ethernet 0

#
# Test(s) for Realtime Clocks and Interrupts Module (RCIM)
#
```

```
#rtck rrtc /dev/rrtc/2c0        # 32-bit RCIM Timer 0
#rtck rrtc /dev/rrtc/2c1        # 32-bit RCIM Timer 1
#rtck rrtc /dev/rrtc/2c2        # 32-bit RCIM Timer 2
#rtck rrtc /dev/rrtc/2c3        # 32-bit RCIM Timer 3
#
#rcim reti /dev/reti/eti00      # Edge Triggered Interrupt 0
#rcim reti /dev/reti/eti01      # Edge Triggered Interrupt 1
#rcim reti /dev/reti/eti02      # Edge Triggered Interrupt 2
#rcim reti /dev/reti/eti03      # Edge Triggered Interrupt 3
#
#rcim pig /dev/reti/eti00       # Programmable Interrupt Generator 0
#rcim pig /dev/reti/eti01       # Programmable Interrupt Generator 1
#rcim pig /dev/reti/eti02       # Programmable Interrupt Generator 2
#rcim pig /dev/reti/eti03       # Programmable Interrupt Generator 3
#

#
# Test(s) For Miscellaneous Controllers
#
# NOTE: The following tests are NOT auto-configured.
# The templates below can be used to configure these
# tests if the device(s) are known to exist.
#
#hsde chnl hsde0:hsde1          # HSD Board 0 to Board 1
#dr11 chnl 0                    # DR11W Board 0
```

B
Diagnostic Programs

The following table lists the diagnostic programs contained on the diagnostic product media in the order in which they should be executed.

Note that to form file names, the program names must be prefixed by a directory name.

Table B-1. Diagnostic Programs

Load Name	Title
Standalone Operator Assistance Programs	
saint	Standalone Interface
Standalone CPU/Memory Subsystem Diagnostics	
memory	Local/Global Memory Diagnostic
cache	Cache Diagnostic
interrupt	Interrupt Controller Diagnostic
timers	Timers Diagnostic
Standalone I/O Controller Diagnostics	
ise	Integral SCSI/Ethernet Diagnostic
hsadisk	VME/SCSI Disk Diagnostic
hsatape	VME/SCSI Tape Diagnostic
Standalone System Interaction	
interact	System Interaction Diagnostic
System-level Tests	
PowerAT	System Reliability and Acceptance Test
PowerIO	Input/Output Subsystem Test

C
Diagnostic Products

The following table lists all available diagnostic products and their corresponding product numbers.

Table C-1. Diagnostic Products

Model Number	Media	System	Image Names			
			4.3	5.1	6.1	6.2
730-9815121	1/4"	PowerMAXION 6400	tape_img4.3_NH5	tape_img5.1_NH5	Not Supported	
730-9815122	DAT	PowerMAXION 6400	tape_img4.3_NH5	tape_img5.1_NH5		
730-9815126	1/4"	PowerMAXION 6408	tape_img4.3_NH7	tape_img5.1_NH7		
730-9815127	DAT	PowerMAXION 6408	tape_img4.3_NH7	tape_img5.1_NH7		
730-9815091	1/4"	NightHawk 6200	tape_img4.3_NH4	tape_img5.1_NH4		
730-9815092	DAT	NightHawk 6200	tape_img4.3_NH4	tape_img5.1_NH4		
730-9815095	1/4"	NightHawk 6800	tape_img4.3_NH4	tape_img5.1_NH4		
730-9815096	DAT	NightHawk 6800	tape_img4.3_NH4	tape_img5.1_NH4		
730-9815132	DAT	NightHawk 6800T	tape_img4.3_NH6	tape_img5.1_NH6		
730-9815112	DAT	Power Hawk Series 600	tape_img4.3_MOTO	tape_img5.1_MOTO		
WT9200-JM	CD	Power Hawk Series 700	Not Supported	tape_img5.1_SYN	tape_img6.1_SYN	tape_img6.2_SYN

Index

S

T