Sun Enterprise 10000 SSP 3.3 Reference Manual

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part Number 806-2888-10
February 2000, Revision A

Copyright 2000 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303-4900 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, SunDocs, Java, the Java Coffee Cup logo, and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227–14(g)(2)(6/87) and FAR 52.227–19(6/87), or DFAR 252.227–7015(b)(6/95) and DFAR 227.7202–3(a).

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2000 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, Californie 94303-4900 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées du système Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, SunDocs, Java, le logo Java Coffee Cup, et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Sun Enterprise 10000 SSP Attributions:

This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim inany distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

IN NO EVENT SHALL THE AUTHORS OR DISTRIBUTORS BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE, ITS DOCUMENTATION, OR ANY DERIVATIVES THEREOF, EVEN IF THE AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHORS AND DISTRIBUTORS SPECIFICALLY DISCLAIM ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, AND THE AUTHORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

RESTRICTED RIGHTS: Use, duplication or disclosure by the government is subject to the restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software Clause as DFARS 252.227-7013 and FAR 52.227-19.

This is scotty, a simple tcl interpreter with some special commands to get information about TCP/IP networks. Copyright (c) 1993, 1994, 1995, J. Schoenwaelder, TU Braunschweig, Germany, Institute for Operating Systems and Computer Networks. Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that this copyright notice appears in all copies. The University of Braunschweig makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Contents

Intro(1M) 9
addboard(1M) 12
autoconfig(1M) 15
board_id(1M) 18
bringup(1M) 20
cb_prom(1M) 24
cb_reset(1M) 25
cbs(1M) 26
check_host(1M) 27
deleteboard(1M) 28
domain_create(1M) 31
domain_history(1M) 33
domain_link(1M) 34
domain_remove(1M) 36
domain_rename(1M) 38
domain_status(1M) 40
domain_switch(1M) 41
domain_unlink(1M) 42
edd(1M) 44

edd_cmd(1M) 46

fad(1M) 48

fan(1M) 49

hostinfo(1M) 51

hostint(1M) 52

hostreset(1M) 54

hostview(1M) 55

hpost(1M) 56

machine_server(1M) 63

moveboard(1M) 64

netcon(1M) 67

netcon_server(1M) 70

netcontool(1M) 71

obp_helper(1M) 73

power(1M) 75

redx(1M) 83

showusage(1M) 85

sigbcmd(1M) 88

snmpd(1M) 90

ssp_backup(1M) 92

ssp_config(1M) 93

ssp_downgrade(1M) 95

ssp_restore(1M) 96

ssp_startup(1M) 97

ssp_unconfig(1M) 98

straps(1M) 99

sys_clock(1M) 100

sys_id(1M) 102

sys_reset(1M) 105

thermcal(1M) 106

thermcal_config(1M) 107

Intro(4) 109

blacklist(4) 110

cb_config(4) 113

cb_port(4) 115

domain_config(4) 116

domain_history(4) 118

edd.emc(4) 119

edd.erc(4) 125

fad_files(4) 136

postrc(4) 137

redlist(4) 166

ssp_resource(4) 169

ssp_to_domain_hosts(4) 170

CHAPTER

Maintenance Commands

8

NAME

Intro – Sun Enterprise 10000 SSP 3.3 administration

DESCRIPTION

This section describes commands, scripts, and programs executed in the Sun Enterprise 10000 SSP environment. For a list of Dynamic Reconfiguration (DR) commands, see the `Intro-dr`(1M) command.

Note - Solaris 7 11/99 is the first Solaris release to support the InterDomain Networks (IDN) feature.

LIST OF FUNCTIONS

`autoconfig`**(1M)**	scantool/interconnect auto config
`addboard`**(1M)**	attach a specific board to a specific domain
`board_id`**(1M)**	read or write board ID values
`bringup`**(1M)**	configure and boot the domain
`cb_prom`**(1M)**	upgrade PROM
`cb_reset`**(1M)**	reset and reboot hung control board
`cbs`**(1M)**	control board server
`check_host`**(1M)**	determine whether the domain is up
`deleteboard`**(1M)**	detach a specific board from its current domain
`domain_create`**(1M)**	create new domain or recreate old one
`domain_history`**(1M)**	display domain history
`domain_link`**(1M)**	link domains to form or expand an InterDomain Network
`domain_remove`**(1M)**	remove an existing domain
`domain_rename`**(1M)**	rename an existing domain
`domain_status`**(1M)**	display domain status
`domain_switch`**(1M)**	change domain name in SUNW_HOSTNAME

domain_unlink**(1M)**	unlink domain(s) from an InterDomain Network
edd**(1M)**	event detector daemon
edd_cmd**(1M)**	send a command to edd(1M)
fad**(1M)**	file access daemon
fan**(1M)**	display or control fan power and speed
hostinfo**(1M)**	display system information
hostint**(1M)**	interrupt processor, dump kernel core
hostreset**(1M)**	reset a hung domain
hostview**(1M)**	system monitor GUI
hpost**(1M)**	control and sequence POST through JTAG
machine_server**(1M)**	multi-purpose server
moveboard**(1M)**	detach a board from its domain and attach it to a specific domain
netcon**(1M)**	network console
netcon_server**(1M)**	network console server daemon
netcontool**(1M)**	network console tool
obp_helper**(1M)**	download OpenBoot to system memory
power**(1M)**	control power
redx**(1M)**	remote emulation debugger
showusage**(1M)**	display DR and board resource information
sigbcmd**(1M)**	send commands from SSP to domain
snmpd**(1M)**	system SNMP proxy agent

`ssp_backup`**(1M)**	back up the SSP environment
`ssp_config`**(1M)**	configure SSP control boards
`ssp_downgrade`**(1M)**	prepares the SSP and Sun Enterprise 10000 for downgrading to a previous SSP release
`ssp_restore`**(1M)**	restore the SSP environment
`ssp_startup`**(1M)**	invoke SSP daemons
`ssp_unconfig`**(1M)**	deconfigure the SSP
`straps`**(1M)**	SNMP trap sink server
`sys_clock`**(1M)**	display / change / set system clock frequencies
`sys_id`**(1M)**	display or change machine ID in SSP IDPROM
`sys_reset`**(1M)**	reset the domain
`thermcal`**(1M)**	read or write thermistor calibration
`thermcal_config`**(1M)**	create SSP thermistor calibration data file

NAME

addboard – attach a specific board to a specific domain

SYNOPSIS

addboard –b *board_number* –d *domain* [–h] [–r *retry_count*] [–q] [–t *timeout*]

DESCRIPTION

`addboard` attempts to attach the board specified by *board_number* to the domain specified by *domain* (also referred to as the target domain). `addboard` performs several Dynamic Reconfiguration (DR) attachment operations that write progress messages to the platform log on the SSP and to standard output.

If `addboard` fails, it writes an error message to standard error, rolls back the state of the specified board to the present and powered on state, and returns a nonzero exit status. If the specified board is already in the specified domain, `addboard` returns an exit status of zero, indicating success.

The `–r` option specifies the maximum number of retry attempts, *retry_count*, in the event that an attach operation fails. The `–t` option specifies a time period in seconds, *timeout*, between retry attempts. If you specify the number of retries, but no timeout, there is no delay between retry attempts. If you specify a timeout, you must also specify the number of retries.

To be eligible for attachment, a board must be powered on and in one of the following states:

- Not in a domain
- In an intermediate attach state (after initial attachment but before complete attachment) in the target domain

If any board in the target domain is in an intermediate detach or unknown state, the `addboard` operation fails and returns an exit status of 11.

OPTIONS

The following options are supported:

`–b` *board_number*	System board number, 0 through 15, to be attached to a domain.
`–d` *domain*	The domain to which the board is to be attached.
`–h`	Write a usage message for `addboard` to standard error.
`–r` *retry_count*	The number of retry attempts to be performed in the event an attach fails.
`–t` *timeout*	Time in seconds before next retry attempt.
`–q`	Quiet mode, which means informational messages are not written to standard output.

EXAMPLES

The following example attaches system board 2 to a domain named `xf1-b4`. If the first attempt to attach the board fails, two retries will be made, if necessary, with a wait time of 10 minutes between retries.

```
addboard -b 2 -d xf1-b4 -r 2 -t 600
```

EXIT STATUS

The following exit values are returned:

`0`	Successfully attached the specified board to the specified domain.
`1`	Command syntax error.
`2`	Failed to acquire DR lock.
`3`	Failed to release DR lock.
`4`	Failed to initialize DR communications.
`11`	Board or target domain is not eligible for attachment.
`12`	A board in the target domain is in an intermediate attach state.
`13`	Failed in initial attachment stage.
`14`	Failed in complete attachment stage.
`15`	Failed to dynamically link with DR library.

ERROR MESSAGES

If `addboard` fails, it writes diagnostic messages to standard error.

FILES

The following files are used by `addboard`:

`/var/opt/SUNWssp/.ssp_private/domain_config`

Software domain configuration

`/var/opt/SUNWssp/adm/messages`

Platform log file

NOTES

`addboard` prevents more than one DR attach or detach command from executing concurrently.

If a previous DR command left the specified board in the intermediate attach state in the target domain, `addboard` continues the DR operation from that stage to complete the attach operation.

SEE ALSO

`complete_attach`(1M), `dr`(1M), and `init_attach`(1M) in the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual*

`deleteboard`(1M), `moveboard`(1M), and `showusage`(1M) in the *Sun Enterprise 10000 Reference Manual*

Sun Enterprise 10000 Dynamic Reconfiguration User Guide

Sun Enterprise 10000 SSP User Guide

NAME

autoconfig – scantool/interconnect auto config

SYNOPSIS

autoconfig [*board_name*]

CAUTION

Only authorized service providers can execute this command. You must be user ssp to run this command.

DESCRIPTION

`autoconfig` checks, in turn, the chip revisions of all configured system boards and the two centerplane halves in the Sun Enterprise 10000 system. It then looks at the `sysboard/chip.ids` file for system boards or the `centerplane/chip.ids` file for centerplane halves in `$SSPVAR/data/Ultra-Enterprise-10000/common/board` to determine the signature for the selected board. `autoconfig` then looks at the file `board.sigs` in the same directory to determine if and how the selected system board or centerplane half should be updated in the Scantool database.

`Warning:` Do not run this command on system boards that are running the operating system, or on the centerplane if any domain is running the operating system.

`autoconfig` executes the following steps:

Step 1 Reads the chip IDs from all rings on the selected board to determine the ID value and the number of chips on a ring.

Step 2 Determines that all chip IDs are valid and that the module type for variable-length rings is recognized.

Step 3 Builds ring signatures for each ring on the selected board.

Step 4 Determines from the `board.sigs` file which rings need updating in the Scantool database, and adds new ring signatures to the `board.sigs` file.

Step 5 Selects the correct ring templates for all variable-length rings that need updating in the Scantool database.

Step 6 Updates the Scantool database if necessary by creating a new revision for the board. `autoconfig` then adds new revision signature to the `board.sigs` file.

Step 7 Updates the *platform_name*`.config` file in `$SSPVAR/data/Ultra-Enterprise-10000/config` with the new revision for the board.

If `autoconfig` detects an error on a ring, it leaves the ring unchanged and continues. If it detects errors when accessing

the `chip.ids` or `board.sigs` files, or in attempting to edit the Scantool database, `autoconfig` restores the files for the selected board, then continues.

On completion `autoconfig` prompts you to reboot the SSP to start the new Scantool database. Do this while logged in as user ssp so that the shell environment variables are set correctly.

OPTIONS

The following options are supported:

board_name Configure the specified board. Valid board names are **`sysboard`***n*, where *n* is an integer 0 to 15, inclusive; or **`centerplane`***n*, where *n* is either 0 or 1. Use this option only when a system board is moved from one slot to another, a new system board is added to the system, or the centerplane is replaced.

FILES

The following files in `$SSPVAR/data/Ultra-Enterprise/` are supported:

`common/board/sysboard/chip.ids`

Contains the IDs for all the chips on a system board.

`common/board/centerplane/chip.ids`

Contains the IDs for all the chips on a centerplane.

`common/board/sysboard/board.sigs`

Contains all the ring signatures and board revision signatures built so far for the system boards.

`common/board/centerplane/board.sigs`

Contains all the ring signatures and board revision signatures built so far for the centerplane.

`config/`*platform_name*`.config`

Contains the current revision given to each system board and centerplane.

`common/board/sysboard/rev`*n*`/`*ringname*`.chips`

`rev`*n* is a copy of the **`rev1`** directory. `autoconfig` edits the chips files in this copied directory as necessary.

`common/board/sysboard/rev`*n*`/`*template*`.chips`

rev*n* is a copy of the **rev1** directory. Variable-length rings require a template containing the correct number of chips, which can be copied over the *ringname*`.chips` file.

`common/board/centerplane/rev`*n*`/`*ringname*`.chips`

rev*n* is a copy of the **rev1** directory. `autoconfig` edits the *ringname*`.chips` files for each effected ring.

NAME

board_id – read or write board ID values

SYNOPSIS

board_id [–r] –b *board_type* –n *board_number*

board_id –w –b *board_type* –n *board_number* –p *part_number* –s *serial_number*

CAUTION

This command is for use by Sun Manufacturing personnel only.

DESCRIPTION

`board_id` enables the Sun Microsystems engineer to display (read) or assign (write) board ID values in the EEPROMs on the system board, centerplane, centerplane support board, control board, memory module, or I/O module.

If neither –r nor –w is specified, –r is the default.

OPTIONS

The following options are supported:

–w — (Write) Assign the specified part number and serial number to the specified board.

–r — (Read) Display information about the specified board.

–b *board_type* — Read or write to the specified board type, where *board_type* is one of the following:

`sb` - System board

`cp` - Centerplane

`cb` - Control board

`csb` - Centerplane support board

`mem` - Memory module

`io` - I/O module

–n *board_number* — Read or write to the specified board number, where *board_number* is one of the following, as determined by the –b option:

A system board number, `0` to `15`

A centerplane half, `0` or `1`

A control board, `0` or `1`

A centerplane support board, `0` or `1`

A memory module, `0` to `15`

	An I/O module, `0` to `15`
–p *part_number*	Assign the specified part number to the board, where *part_number* is expressed in the form *ccc-aaaa-ss-rr*, and where: *ccc* is a 3-digit numeric category code. Currently, this code is 612 *aaaa* is a 4-digit numeric base part number (assembly number) *ss* is a 2-digit numeric part number suffix *rr* is a 2-digit numeric revision
–s *serial_number*	Assign the specified serial number to the board, where *serial_number* is a string of three to 12 printable, non-space characters.

EXAMPLES

EXAMPLE 1 Using the **board_id** command:

Display information about Board 1 on the system board.

```
board_id -b sb -n 1
```

Assign part number 501-2341-12-01 and serial number 37KR59 to Board 1 on the system board.

```
board_id -w -b sb -n 1 -s 37KR59 -p 501-2341-12-01
```

Display information about Board 1 on the centerplane. Since neither –r nor –w are specified, board_id assumes –r (the default). And –r ignores both –s and –p.

```
board_id -b cp -n 1 -s 37KR59 -p 501-2341-12-01
```

NAME

bringup – configure and boot the domain

SYNOPSIS

bringup [–f] [–F] [–p*proc*] [–Q*boot_proc*] [–aghvCL] [[–A] {on | off}][–l*level*] [[–D] {on | off}][–X*blacklist_file_pathname*] [*boot_args*]

DESCRIPTION

`bringup` executes the following steps to boot the domain specified in the SUNW_HOSTNAME environment variable. If any step fails, `bringup` displays an error message and aborts.

1. Executes the `power`(1M) command to check whether the domain is powered on. If so, `bringup` proceeds; if not, it displays a message instructing the user to turn on power to the domain, then aborts. Note that if –f is used, `bringup` does not execute the `power`(1M) command.
2. Executes `check_host`(1M) to determine whether the domain is already up. If not, `bringup` boots the domain. If the domain is already up, `bringup` displays a message to that effect and asks if it should continue. Type **n** and it aborts; type **y** and it asks if you are executing `bringup` because of a hung domain. Your response to this question is recorded for problem-tracking purposes only; in either case, `bringup` boots the domain.
3. Checks whether any other domains are up, or being brought up, to determine whether it should execute `hpost`(1M) with or without the –C (configure centerplane) option in the next step. If another domain is being brought up and is configuring the centerplane, this instance of `bringup` waits until that domain has finished configuring the centerplane, then proceeds. If no other domains are up or being brought up, `bringup` displays a message informing you that it intends to configure the centerplane, and asks you whether it should proceed. (If you specified –f on the command line, `bringup` continues without this message.)

   - If you type **y**, `bringup` configures the centerplane and continues. If the centerplane is already configured – for example, due to an earlier domain that no longer exists – it is reconfigured. No harm is done, but your bringup may take a little longer.
   - If you type **n** and the centerplane is already configured, `bringup` continues without reconfiguring the centerplane.
   - If you type **n** and the centerplane is *not* already configured, `hpost`(1M) most likely will fail.

     Note: If no domains are up and you do not know whether the centerplane is configured, the safest response is **y**.

     At this point `bringup` determines whether the domain was created with a version of Solaris earlier than Solaris 2.6. If so, `bringup` skips the next step and proceeds to Step 5. Otherwise, it executes the next step, then Step 5.

4. Executes `domain_unlink`(1M) on the respective domain to verify that it is not a member of an IDN.

   `bringup` displays the following message during this period, which can take several seconds depending on the size of the domain's respective IDN and the state of those domain members:

   `Checking` *domain* IDN configuration...

   When this operation is completed, `bringup` displays the word `done`.

   If an error occurs during the IDN operation, `bringup` displays `error` along with the captured output of the `domain_unlink`(1M) command. If another IDN operation is in progress at the time, it displays `busy`, then waits for the previous IDN operation to complete.

   If the `domain_unlink`(1M) command fails because multiple IDN members are in unknown states, you may have to execute it manually to unlink those domains using a single command prior to performing the `bringup`.

   Note that the unlinking of the domain from its respective IDN is a transient condition; the MIB is not updated. Once the domain boots, if the proper IDN software is present the SSP Event Detection Daemon, `edd`(1M), will detect the availability of the domain and automatically perform a `domain_link`(1M) to reconnect the domain to its IDN.

5. Executes `hpost`(1M) with its `–C` option to configure the centerplane then the domain, or without it to configure just the domain.
6. Starts `obp_helper`(1M) and `netcon_server`(1M) to proceed with the OBP and operating system boot sequence and set up the network console.
7. Updates the MIB according to the final domain configuration.

OPTIONS

The following options are supported:

`–f` Force execution, even if the domain is already up.

`Warning:` `bringup` passes `–C` to `hpost`(1M) when appropriate, regardless of whether you specify it on the `bringup` command line. However, if you specify `–f` with `–C`, `bringup` passes `–C` even if one or more other domains are up, causing those other domains to be reset. Therefore, be extremely careful about specifying `–C` and `–f` together. If you do use `–f`, `bringup` does not check whether the domain is powered up.

`–F` Do not use this option on the command line. It is reserved for automatic reboot scripts; see `edd`(1M).

`–h` Print a brief description of `bringup` command line options.

–Q — Send a request for hpost(1M) to execute a faster, limited version of POST.

Note - Do not execute –Q on the command line. It is invoked by the SSP software (see edd(1M)) when a domain reboot is requested as a result of the uadmin(2) system call (which is used by the reboot(1m), uadmin(1m), shutdown(1m), and init(1m) commands) or an Openboot-initiated reset. Depending on the state of the domain, manual use of bringup -Q may cause a bringup failure.

–L — Send a request for hpost(1M) to use its –s and –v20 options, sending all output to the syslog.

–v — Send a request for hpost(1M) to use its –v70 option, which produces more detailed information.

–S — bringup passes this option to netcon_server(1M).

boot_args — bringup assumes that arguments other than those described above are boot arguments and passes them to obp_helper(1M), which passes them verbatim to the OpenBoot boot(1M) command.

Arguments after "—" are always passed to obp_helper(1M). This is useful for arguments that conflict with bringup(1M) arguments or for names that being with a "-". For example, both of these commands are equivalent and the –D argument is passed to obp_helper(1M).

```
bringup–D

bringup—D
```

–a, –g, –l*level*, –p *proc*, –C, –X *blacklist_file_pathname*

bringup passes these options to hpost(1M).

–A **{on | off}** , –D **{on | off}**

bringup(1M) passes these options to obp_helper(1M).

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`check_host`(1M), `domain_unlink`(1M), `hpost`(1M), `netcon_server`(1M), `obp_helper`(1M), `snmpd`(1M), `sys_reset`(1M) in the *Sun Enterprise 10000 SSP Reference Manual*

`boot`(1M) in the *SunOS Reference Manual*

NAME

cb_prom – upgrade PROM

SYNOPSIS

cb_prom –d *file* [–f] –h *hostname*

cb_prom –e –h *hostname*

cb_prom –p *file* –h *hostname*

cb_prom –r –h *hostname*

CAUTION

Only authorized service providers should use this command.

DESCRIPTION

cb_prom manipulates the contents of the Sun Enterprise 10000 Control Board Flash PROM on the primary control board. This PROM is designated for field upgrades to the boot firmware required to download the Control Board Executive (CBE).

OPTIONS

The following options are supported:

–d *file* **[**–f**]**	Dump the contents of the Flash PROM to *file*. By default, –d does not overwrite *file* if it already exists. You can use –f, the force option, to override this safeguard.
–e	Erase the contents of the Flash PROM.
–h *hostname*	Execute the cb_prom command on the control board named *hostname*.
–p *file*	Program the Flash PROM with the contents of *file*.
–r	Display the revision of the Flash PROM currently in use on the control board.

NOTES

An upgrade to or erasure of the boot firmware takes effect only after a control board reboot; for example, via cb_reset(1M).

NAME

cb_reset – reset and reboot hung control board

SYNOPSIS

cb_reset [–v] [–p *platform_name*] [*control_board_name*]

DESCRIPTION

In its default form `cb_reset` sends a fixed Ethernet packet to the certain control board(s) of the Sun Enterprise 10000 system specified by the SUNW_HOSTNAME environment variable, causing those boards to reset and reboot themselves. `cb_reset` sends reset packets to all network links that are up. You can use `ifconfig`(1M) with its –`au` option to see a list of such links.

OPTIONS

The following options are supported:

–v	(Verbose mode) Display a message confirming that the packet has been sent, along with the Ethernet addresses of both the source (the SSP that sent the packet) and destination (the control board that received it). When used, this option must appear first on the command line.
–p *platform_name*	Send the Ethernet packet to the control board of the specified Enterprise 10000 system (*platform_name*).
control_board_name	Send the Ethernet packet to the specified control board, where the name of the *control_board_name* is specified in `cb_config`(4). If –p is also specified, `cb_reset` sends the packet to the specified control board on the specified Sun Enterprise 10000 system. Otherwise, it sends it to the specified board on the system specified by the SUNW_HOSTNAME environment variable.

FILES

The following files are supported:

`$SSPVAR/.ssp_private/cb_config` – control board configuration file

SEE ALSO

`edd`(1M), `ssp_startup`(1M)

NAME

cbs – control board server

SYNOPSIS

cbs

CAUTION

Do not execute this program manually. It is automatically invoked by the SSP startup script and periodically monitored for re-start.

DESCRIPTION

The `cbs` server program provides central access to the Sun Enterprise 10000 control board for client programs running on the SSP. It has sole access to the Sun Enterprise 10000 JTAG scan database and is the lock manager for all JTAG operations.

`cbs` converts client requests to CBMP (Control Board Management Protocol) messages and passes them to the Control Board Executive (CBE) running on the Sun Enterprise 10000 control board. It relies on `domain_config`(4) and `cb_config`(4) to determine which platform it is to manage and which control board it is to use for communication.

The communication protocol between CBE and `cbs` allows SSP applications to retrieve and modify control board resources, perform JTAG scan operations, and submit monitoring scripts.

Each SSP can run only one instance of `cbs` at a time.

FILES

The following files are supported:

`$SSPVAR/data/Ultra-Enterprise-10000` – Sun Enterprise 10000 JTAG scan database

`$SSPVAR/pids/cbs.pid` – process ID file

`$SSPVAR/.ssp_private/domain_config` – domain configuration file

`$SSPVAR/.ssp_private/cb_config` – control board configuration file

SEE ALSO

`machine_server`(1M), `domain_config`(4), `cb_config`(4)

NAME

check_host – determine whether the domain is up

SYNOPSIS

check_host [–q]

DESCRIPTION

`check_host` is called from the `bringup`(1M) script to check whether the domain specified by the environment variable SUNW_HOSTNAME is up. If *all* configured processors are running the operating system, `check_host` sees the domain as up, displays the message `Host is UP`, and returns a status 0. If *any* configured processors are *not* running the operating system, `check_host` considers the domain down, displays `Host is DOWN`, and returns a status 1.

OPTIONS

The following options are supported:

`–q` (Quiet Mode) Report the exit status, but do not echo any information to stdout.

NOTES

Processors are configured by `hpost`(1M), which is run by `bringup`(1M).

`bringup`(1M) does not execute if the domain is up. **`bringup -f`** overrides this safety feature.

SEE ALSO

`bringup`(1M), `edd`(1M), `hpost`(1M), `snmpd`(1M)

NAME

deleteboard – detach a specific board from its current domain

SYNOPSIS

deleteboard –b *board_number* [–h] [–r *retry_count*] [–q] [–t *timeout*]

DESCRIPTION

`deleteboard` attempts to detach the board specified by *board_number* from its current domain. `deleteboard` performs several Dynamic Reconfiguration (DR) detachment operations that write progress messages to the platform log on the SSP and to standard output.

If `deleteboard` fails, it writes error information to standard error, rolls back the state of the specified board to be in its original domain, and returns a nonzero exit status. If the specified board is already detached and powered on, `deleteboard` returns an exit status of zero, indicating success.

The `–r` option specifies the maximum number of retry attempts, *retry_count*, in the event that a detach operation fails. The `–t` option specifies a time period in seconds, *timeout*, between retry attempts. If you specify the number of retries, but no timeout, there is no delay between retry attempts. If you specify a timeout, you must also specify the number of retries.

To be eligible for detachment, a board must be powered on and in one of the following states:

- In a domain
- In an intermediate detach state (after the drain stage but before complete detachment)

If the specified board is not in a domain or has already been detached, `deleteboard` returns an exit value of zero, indicating success. A DR platform error message indicates that the board is already detached.

If any board in the current domain is in an intermediate attach or unknown state, the `deleteboard` operation fails and returns an exit status of 5. If a board other than that specified by *board_number* is in an intermediate detach state, the `deleteboard` operation fails and returns an exit status of 6.

OPTIONS

The following options are supported:

`–b` *board_number*	System board number, 0 through 15, to be detached from its domain.
`–h`	Write a usage message for `deleteboard` to standard error.
`–r` *retry_count*	The number of retry attempts to be performed in the event an detach fails.
`–t` *timeout*	Time in seconds before next retry attempt.

`-q`	Quiet mode, which means informational messages are not written to standard output.

EXAMPLES

The following example detaches system board 2 from its current domain. If the first attempt to detach the board fails, two retries will be made, if necessary, with a wait time of 15 minutes between retries.

```
deleteboard -b 2 -r 2 -t 900
```

EXIT STATUS

The following exit values are returned:

`0`	Successfully detached the specified board from its domain.
`1`	Command syntax error.
`2`	Failed to acquire DR lock.
`3`	Failed to release DR lock.
`4`	Failed to initialize DR communications.
`5`	Board or current domain is not eligible for detachment.
`6`	A board in the current domain is in an intermediate detach state.
`7`	DR daemon will not allow the detach because it will either remove the last processor from the domain or drop the domain below some minimum memory requirement.
`8`	Failed during memory drain operation.
`9`	The drain operation failed to make progress over some period of time.
`10`	Failed in attempting to complete the detach.
`15`	Failed to dynamically link with DR library.

ERROR MESSAGES

If `deleteboard` fails, it writes diagnostic messages to standard error.

FILES

The following files are used by `deleteboard`:

`/var/opt/SUNWssp/.ssp_private/domain_config`

Software domain configuration

`/var/opt/SUNWssp/adm/messages`

Platform log file

NOTES

`deleteboard` prevents more than one DR attach or detach command from executing concurrently.

If the specified board is found in an intermediate detach state, `deleteboard` completes the detach operation.

SEE ALSO

`complete_detach`(1M), `dr`(1M), and `drain`(1M) in the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Guide*

`addboard`(1M), `moveboard`(1M), and `showusage`(1M) in the *Sun Enterprise 10000 SSP Reference Manual*

Sun Enterprise 10000 Dynamic Reconfiguration User Guide

Sun Enterprise 10000 SSP User Guide

NAME

domain_create – create new domain or recreate old one

SYNOPSIS

domain_create –d *domain_name*

domain_create –d *domain_name* [–b *sysboard_list*] [–o *os_version*] [–p *platform_name*] [–t *platform_type*]

DESCRIPTION

You can use the `domain_create` command to create a domain, or to recreate a domain that once existed but was subsequently removed via `domain_remove`(1M). The `domain_history`(1M) command displays a list of domains removed by `domain_remove`(1M).

When creating a new domain, you must specify the –d, –b, –o and –p options. `domain_create` creates the new domain and assigns it the name specified by –d.

When recreating a domain, you must use only the –d option. `domain_create` recreates the domain according to its original specifications. If it can find no information about a previously existing domain of that name, `domain_create` returns an error.

Note - If optional parameters are not used on the command line, the command uses the original specification for the domain from the `domain_history` file. If a value cannot be found in the `domain_history` file, an error is returned.

You can execute `domain_create` from the command line, or by selecting Configuration –> Domain –> Create from within Hostview (see `hostview`(1M)).

No historical information is maintained regarding IDN membership for a previously existing domain. When you create a domain, it does not belong to an IDN.

CONDITIONS

Execute `domain_create` only when *all* of the following are true:

- The composition of system boards specified for the domain have, at minimum, a network interface, SCSI interface, and sufficient memory to support an autonomous system.
- The domain name is defined in the host's `/etc/hosts` file.
- The system boards are all present and none are in use.

OPTIONS

The following options are supported:

–d *domain_name*	Either recreate the removed domain named *domain_name*, or create a new domain and assign it the name *domain_name*.

–b *sysboard_list*	Include the specified system boards in the new domain. The system boards are listed by board number and are separated by a space.
–o *os_version*	The version of the SunOS operating system – e.g., 5.5.1, 5.6 or 5.7 – to be running on the domain.
–p *platform_name*	The name of the Sun Enterprise 10000 system whose boards are to be used for the new domain.
–t *platform_type*	The type of system from which the domain originates. The default is the platform type of the machine from which the domain originates. The platform type for the Sun Enterprise 10000 system is `Ultra-Enterprise-10000`.

EXAMPLES

EXAMPLE 1 Creating a Domain

The following command creates a domain named junior, which is configured with three boards numbered 0, 2 and 9, and that runs version 2.5.1 of the operating system. junior's boards came from the machine whose platform name is dad.

```
domain_create -d junior -b 0,2,9 -o 2.5.1 -p dad
```

RETURN VALUES

Upon successful completion `domain_create` returns a zero value and reconfigures the SSP to see the new domain; otherwise, it returns a non-zero value.

NOTES

The `domain_create` command can create only one domain at a time.

After creating a domain from the command line (this step is not necessary when doing so from Hostview), set the SUNW_HOSTNAME environment variable as follows to ensure proper operation of the SSP command set:

setenv *SUNW_HOSTNAME domainname*

You can now bring up the domain via the `bringup`(1M) command.

Note that a new operating system patch may involve a new version of a host-dependent SSP binary. If so, the patch README file will have the operating system version number you should use to create the domain that will run the new operating system patch.

SEE ALSO

`domain_history`(1M), `domain_remove`(1M), `domain_rename`(1M), `domain_status`(1M), `domain_switch`(1M)

NAME

domain_history – display or remove domain history

SYNOPSIS

domain_history [–d *domain_name*] [–r]

DESCRIPTION

When invoked with no options `domain_history` displays the `domain_history`(4) file, which contains configuration information about all domains that once existed, but were removed via `domain_remove`(1M).

OPTIONS

The following options are supported:

–d *domain_name* — Display configuration information for only the specified domain.

–r — Remove all configuration information from the `domain_history`(4) file. If you also specify –d, this option removes only the information about the specified domain.

Note - Since recreating a domain is easier than creating a new one, exercise restraint in using the –r option.

RETURN VALUES

Upon successful completion `domain_history` returns a zero value; otherwise, it returns a non-zero value.

EXAMPLES

EXAMPLE 1 Displaying Configuration Information

The following command displays configuration information about the previously existing domain named junior.

```
domain_history -d junior
```

EXAMPLE 2 Removing History File Entries

Removes the entry for the previously existing domain named junior from the `domain_history`(4) file.

```
domain_history -d junior -r
```

ERRORS

domain_name `does not exist.`

`domain_history` could not find an entry of the specified name in the `domain_history`(4) file.

SEE ALSO

`domain_create`(1M), `domain_remove`(1M), `domain_rename`(1M), `domain_status`(1M), `domain_switch`(1M), `domain_history`(4)

NAME

domain_link – link domains to form, or expand, an InterDomain Network (IDN)

SYNOPSIS

domain_link *domain_name_a domain_name_b*

DESCRIPTION

The `domain_link` command links Dynamic System Domains, referred to as domains in this document, only on the Sun Enterprise 10000 server. You can use the `domain_link` to create, or expand, an IDN within the same server. The system boards that host memory in the domains must contain at least one processor. If one of the specified domains is already a member of an IDN, the `domain_link` command adds the other specified domain to that IDN. If both domains are members of different IDNs, the `domain_link` command combines those IDNs into a single, larger IDN. If neither domain is part of an IDN, the `domain_link` creates a new IDN with both domains as members.

You can specify only two domain names when you use the `domain_link`(1M) command. If you want to create a multi-domain IDN, you must first link two domains, then you can link the other domains to the IDN by specifying the name of one of the domains in the IDN with the name of the new domain.

Each IDN has one master domain, which maintains the Shared Memory Region (SMR), as described in the *Sun Enterprise 10000 InterDomain Networks User Guide*. The master domain is dynamically chosen by the IDN subsystem.

Note - If the domains are up and running, you must set the `idn-smr-size` variable at the OBP prompt and reboot the domain before you attempt to link the domain to an IDN.

If one domain goes down, the others can continue to communicate over the IDN. Although all of the domains within an IDN are logically connected in a point-to-point configuration, the IDN feature also supports broadcasting at the network interface level. For more information about arbstops and IDNs, see the *Sun Enterprise 10000 InterDomain Networks User Guide*.

OPTIONS

The following options are supported.

domain_name_a Represents the name of the first domain that you want to link. The domain could be a member of an existing IDN, or it could be a domain that is not already linked to an IDN.

domain_name_b Represents the name of the second domain that you want to link. The domain could be a member of an existing IDN, or it could be a domain that is not already linked to an IDN.

EXIT STATUS

The following exit values are returned:

`0` Domain link successful.

`nonzero`	Domain link unsuccessful.

ENVIRONMENT VARIABLES

Before you attempt to create an IDN or to link a domain to an existing IDN, ensure that the SUNW_HOSTNAME shell environment variable is set to the name of a domain that supports the IDN feature. For this variable, the platform name does not work with IDN commands.

SEE ALSO

`domain_create`(1M), `domain_unlink`(1M), `domain_history`(1M), `domain_remove`(1M), `domain_rename`(1M), `domain_status`(1M)

Sun Enterprise 10000 InterDomain Networks User Guide

NAME

domain_remove – remove an existing domain

SYNOPSIS

domain_remove –d *domain_name*

DESCRIPTION

`domain_remove` dismantles the specified domain, reconfiguring its system boards to a *physically present and not in use* state on the Enterprise 10000 system. You should not remove a domain until it has been shutdown, and you cannot remove a domain that is a member of an IDN. See the **`Caution`** in the ERRORS section, below.

`domain_remove` displays the following message, giving you the option of saving certain information:

```
The following subdirectories contain domain-specific information,
such as messages files, configuration files, and hpost dump files.
You may choose to keep these directories if you still need this
information. This domain may be recreated with or without this
information being saved.
```

`domain_remove` then asks whether you want to keep the specific subdirectories. After you respond it does as you ask, then proceeds.

After `domain_remove` completes, the boards are available for a subsequent `domain_create`(1M) or DR Attach.

You can execute `domain_remove` from the command line, or you can do so via Hostview. To use Hostview to remove a board, select a board in the domain you wish to remove and choose Configuration –> Domain –> Remove. You cannot remove a domain when it is running the operating system.

If interrupted during its configuration-verification phase, `domain_remove` aborts. After this phase, once it has begun the reconfiguration, it cannot be interrupted.

RETURN VALUES

Upon successful completion `domain_remove` returns a zero value; otherwise, it returns a non-zero value.

ERRORS

The specified domain is running the operating system.

```
Domain 'domain_name' is up, must be
shutdown before removing. Continue anyway (y/n)?
```

CAUTION: If you tell `domain_remove` to continue, removing a domain that is running the operating system, it will do so, crashing the domain and potentially causing loss of data.

The specified domain is a member of an IDN. It must be unlinked from the IDN before it can be removed.

```
domain_remove:  Domain (domain_name)
is linked to an IDN. Domain must be unlinked before
```

it can be removed. See domain_unlink(1M).

NOTES

Each execution of the domain_remove command can remove only one domain.

Upon successful removal of the configuration, the corresponding obp_helper(1M) and netcon_server(1M) daemons are also terminated.

SEE ALSO

domain_create(1M), domain_history(1M), domain_unlink(1M), domain_rename(1M), domain_status(1M), domain_switch(1M), netcon_server(1M)

NAME

domain_rename – rename an existing domain

SYNOPSIS

domain_rename –d *domain_name* [–n *new_domain_name*] [–o *new_os_version*]

DESCRIPTION

`domain_rename` changes the name or version of a domain, or both. If you specify the –n option, `domain_rename` changes the name from the domain name specified by the –d option to the domain name specified by the –n option. If you specify the –o option, `domain_rename` changes the version number of the SunOS operating system for the domain.

You can execute `domain_rename` from the command line or from Hostview. Your attempt fails if the target domain already is running the operating system.

When renaming a domain you must reconfigure the Internet address and other configuration information on the Sun Enterprise 10000 system domain and also on the SSP to enable the system to recognize the domain's new name. Otherwise, the domain cannot be reached and controlled by the SSP.

If interrupted during its configuration-verification phase, `domain_rename` aborts. You cannot interrupt it after it has completed this phase and begun the reconfiguration.

OPTIONS

The following options are supported:

–d *domain_name*	Name of the domain to be changed.
–n *domain_name*	New name for the domain specified in –d *domain_name*.
–o *new_os_version*	The version of the SunOS operating system – e.g., 5.5.1, 5.6, 5.7, or 5.8 – to be running on the domain.

RETURN VALUES

Upon successful completion `domain_rename` returns a zero value (otherwise, it returns a non-zero value). You now can bring up the domain via the `bringup`(1M) command.

ERRORS

domain_name `does not exist.`

`domain_rename` cannot find the specified domain name.

new_domain_name `already exists.`

The name you are trying to assign the domain is already in use.

NOTES

After renaming a domain via this command you must set the SUNW_HOSTNAME environment variable as follows to ensure proper operation of the SSP command set:

setenv SUNW_HOSTNAME *new_domain_name*

SEE ALSO domain_create(1M), domain_history(1M), domain_remove(1M), domain_status(1M), domain_switch(1M)

NAME

domain_status – display domain status

SYNOPSIS

domain_status [–d *domain_name*]

DESCRIPTION

`domain_status` displays the `domain_config`(4) file, which contains the following information for each configured domain: domain name, platform type, platform name, operating system, system boards, and Inter-Domain Network (IDN) information.

OPTIONS

The following options are supported:

–d *domain_name* Show information about only the specified domain.

RETURN VALUES

Upon successful completion `domain_status` returns a zero value; otherwise, it returns a non-zero value.

NOTES

`domain_status` Any Inter-Domain Networks are identified in the listing by a number, followed by the names of the domains in that network, similar to the following. The column headings are Domain, Type, Platform, Operating System, and System Boards.

```
xf6-b2          Ultra-Enterprise-10000   all-xf6        5.6    2
xf6-b1          Ultra-Enterprise-10000   all-xf6        5.6    1
xf6-b5          Ultra-Enterprise-10000   all-xf6        5.6    5 13
xf6-b6          Ultra-Enterprise-10000   all-xf6        5.7    6 14
xf6-b7          Ultra-Enterprise-10000   all-xf6        5.7    7 15
xf6-b8          Ultra-Enterprise-10000   all-xf6        5.7    0 8
xf6-b4          Ultra-Enterprise-10000   all-xf6        5.7    4 12
xf6-b3          Ultra-Enterprise-10000   all-xf6        5.7    3 11
```

The listing number is simply a tag used in the listing; it is not a persistent identifier for the IDN.

```
0: xf6-b6 xf6-b7 xf6-b8
1: xf6-b3 xf6-b4
```

SEE ALSO

`domain_create`(1M), `domain_history`(1M), `domain_remove`(1M), `domain_rename`(1M), `domain_switch`(1M)

NAME

domain_switch – change domain name in SUNW_HOSTNAME

SYNOPSIS

domain_switch *domain_name*

DESCRIPTION

`domain_switch` is a `csh`(1) alias defined by user ssp's `.cshrc` file. When logged in as user ssp you can execute `domain_switch` on the command line to change the environment variable SUNW_HOSTNAME, making the current SSP session control the specified domain. `domain_switch` also changes the `csh` prompt to reflect the new value of SUNW_HOSTNAME.

Note that the `domain_switch` alias is less robust than the true SSP commands. If you specify more than one *domain_name* argument, only the first is used. `domain_switch` makes no effort to verify that the specified domain name is valid. And if you do not specify a *domain_name* argument, `domain_switch` generates a message similar to the following:

```
xf1-ssp:xf1% domain_switch
Bad ! arg selector
```

EXAMPLES

EXAMPLE 1 To Change the Domain Name

```
xf1-ssp:xf1% domain_switch xf1-deux

Switch to domain xf1-deux

xf1-ssp:xf1-deux%
```

NAME

domain_unlink – unlink domain(s) from an InterDomain Network (IDN)

SYNOPSIS

domain_unlink [–f | –F] *domain_name* ...

DESCRIPTION

The `domain_unlink` command disconnects one or more domains from an IDN. (A single instance of this command cannot remove domains that are linked to different IDNs.) Domains that remain in the IDN continue to communicate over that network in the normal fashion. Before you unlink a domain, you should verify that the associated network interface is not in use.

OPTIONS

The following options are supported:

`–f` Represents the soft force option. With this option, `domain_unlink` attempts to unlink all of the specified domains in the normal manner; however, if a time-out condition occurs because a domain is completely non-responsive, `domain_unlink` uses the `–F` option to remove the link, forcing the domain to be unlinked.

`–F` Represents the hard force option. With this option, `domain_unlink` unlinks the specified domain from all of the other domains in the IDN and does so without synchronizing the disconnects. You should use this option when the specified domain is completely non-responsive or when it has been isolated from the IDN as part of AWOL recovery.

domain_name Represents the name of the domain that you want to unlink. All of the specified domains must be members of an IDN. You can unlink more than one domain at a time; however, if you specify N-1 domains in the IDN, where N equals the total number of domains in the IDN, the `domain_unlink` command unlinks all of the domains in the IDN, effectively dismantling the IDN.

WARNINGS

The force option should be used only as a recovery mechanism when a domain is known to be in an unknown state (AWOL). It should not be used under normal conditions. It could result in an arbstop if the hardware is reprogrammed while the IDN is active. Note that if multiple domains within an IDN are hung, you need to unlink all of the hung domains in a single operation.

ENVIRONMENT VARIABLES

Before you attempt to create an IDN or to link a domain to an existing IDN, ensure that the SUNW_HOSTNAME shell environment variable is set to the name of a domain that supports the IDN feature. For this variable, the platform name does not work with IDN commands.

EXIT STATUS

Upon successful completion, `domain_unlink` returns a zero exit value. Otherwise, it returns a non-zero value, which indicates the domain unlink was not successful.

SEE ALSO

`domain_create`(1M), `domain_link`(1M), `domain_history`(1M), `domain_remove`(1M), `domain_rename`(1M), `domain_status`(1M)

`ifconfig`(1M) in the *SunOS Reference Manual*

Sun Enterprise 10000 InterDomain Networks User Guide

NAME

edd – event detector daemon

SYNOPSIS

edd [–a] [–d]

CAUTION

Do not execute this program manually. It is automatically invoked by the SSP startup script and periodically monitored for re-start.

DESCRIPTION

`edd` is a key component of the Sun Enterprise 10000 system's RAS features. When executed with no options, it checks the `edd.emc`(4) configuration file to determine which event-detection scripts it should use in monitoring the system. Then, if it detects one of those events for which it is checking, `edd` uses the `edd.erc`(4) configuration file to determine whether to take action and, if so, what action to take. If appropriate, it then executes the specified Response Action Script.

Each Sun Enterprise 10000 system has one `edd.emc`(4) file, one `edd.erc`(4) file for global events (those that affect the entire system), and one `edd.erc`(4) file for each domain. The locations of these files are shown below in FILES. Refer to `edd.emc`(4) for a list of the events being monitored and `edd.erc`(4) for a list of the actions to be taken if any of those events occur.

You can prevent `edd` from responding to some events by editing either the `edd.emc`(4) file to disable the uploading of certain event-detection scripts, or editing the `edd.erc` file to prevent `edd` from executing certain Response Action Scripts.

In case of conflict the options to `edd` override any actions specified in the `edd.emc`(4) and `edd.erc`(4) files.

Note for Service Providers: When `edd` produces a dumpfile due to an arbstop or recordstop in a domain that is part of an InterDomain Network (IDN), the dumpfile provides information about all domains in that IDN. For an arbstop, `edd` also saves the bbsram information for the boot processors of the IDN domains in a separate file.

OPTIONS

The following options are supported:

`–a` Take no action; do not invoke any Response Action Scripts.

`–d` Do not upload event monitoring scripts to the active control board.

SIGNALS

`SIGHUP` Reread the event response configuration files. This signal does not affect response actions for previously received events.

FILES

The following files are supported:

`$SSPVAR/etc/`*platform_name*`/edd.emc`

`$SSPVAR/etc/`*platform_name*`/edd.erc`

`$SSPVAR/etc/`*platform_name*`/`*domain_name*`/edd.erc`

SEE ALSO

`cbs`(1M), `edd_cmd`(1M), `hostview`(1M), `snmpd`(1M), `edd.emc`(4), `edd.erc`(4)

NAME

edd_cmd – send a command to edd

SYNOPSIS

edd_cmd [–h] [–e] [–x *cmd*]

CAUTION

This command is for use primarily by service providers. Consult with your service provider before using it.

DESCRIPTION

edd_cmd is a utility tool. When executed with no argument it outputs the last command executed and the current execution state. The last command executed is either **start**, **stop**, **reconfig** (for **rc**), or **uninitialized** (no command executed since startup). The state is started-monitoring, stopped-monitoring, or uninitialized, the initial state on startup.

–h is a help option that provides a summary of command line options. The –e option displays information on system event monitoring and indicates whether event monitoring is enabled or disabled for the platform and domain.

When edd_cmd sends a command to edd(1M), it sets a trap handler that waits for an acknowledgement that edd(1M) has accepted the command.

OPTIONS

The following options are supported:

–h	Displays a summary of the command line options.
–e	Displays event monitoring from the edd.emc and edd.erc files.
–x *cmd*	Execute the specified command, where *cmd* is of the following: start - Make edd(1M) start event-detection monitoring. stop - Make edd(1M) stop event-detection monitoring. rc - Make edd(1M) reread its configuration files and perform event-detection monitoring.

EXAMPLES

EXAMPLE 1 To Start the EDD

```
% edd_cmd
EDD: Command = uninitialized, State = stopped-monitoring

% edd_cmd -x start

% edd_cmd
EDD: Command = start, State = started-monitoring
```

EXAMPLE 2 To Display Event Monitoring Information

```
% edd_cmd -e
EDD: Command = uninitialized, State = uninitialized

                  Monitoring  Platform  Domain
Event             edd.emc     edd.erc   edd.erc  Command

arbstop           enabled     none      enabled  Arbstopact
bulk_power_fail   enabled     enabled   none     BulkPowerFailact
bulk_power_norm   enabled     enabled   none     BulkPowerNormact
cb_power_off      enabled     enabled   none     PowerOffact
```

FILES The following files are supported:

`$SSPETC/snmp/Ultra-Enterprise-10000.mib`

`$SSPVAR/etc/`*platform*`/edd.emc`

`$SSPVAR/etc/`*platform*`/edd.erc`

`$SSPVAR/etc/`*platform*`/`*domain*`/edd.emc`

SEE ALSO `edd`(1M), `snmpd`(1M), `straps`(1M), `edd.emc`(4), `edd.erc`(4)

NAME

fad – file access daemon

SYNOPSIS

fad

CAUTION

Do not execute this daemon manually. The SSP startup script invokes it, then monitors it and restarts it as necessary.

DESCRIPTION

The `fad` daemon provides distributed file access services to SSP clients that need to monitor, read and write changes of SSP configuration files. Only readable files listed in the `fad_files`(4) file can be monitored.

`fad` provides a file-locking service similar to `lockf`(3C). Once a file is locked by a client, subsequent lock requests against the same file by other clients are blocked and queued. When a file lock is released, the next client on the queue is unblocked and serviced.

`fad` relies on other SSP server daemons, including `machine_server`(1M). Each SSP can run only one instance of `fad` at a time.

FILES

The following files are supported:

`$SSPVAR/.ssp_private/fad_files` – SSP configuration file list

`$SSPVAR/pids/fad.pid` – process ID file

SEE ALSO

`machine_server`(1M), `fad_files`(4)

`lockf`(3C) in the *SunOS Reference Manual*

NAME

fan – display or control fan power and speed

SYNOPSIS

fan

fan –p on

fan –p off [–l {front | rear}][–t *FanTrayList*]

fan –s{nominal | fast}

DESCRIPTION

When used without options, the `fan` command displays the speed and power status of the fans. When used with the –s or –p option, it changes their speed or power status. You can use –l and –t with `–p off` to specify which fans are to be turned off. `fan –p on` turns on all fans.

Each Sun Enterprise 10000 system has four fan shelves, two in the front of the cabinet and two in the rear. Each shelf contains four fan trays, for a total of 16 fan trays. Each fan tray can contain two fans (numbered 0 and 1), for a maximum of 32 fans per system.

The fan trays are organized as follows:

Fan trays 0, 1, 2, 3	Shelf 0 at rear-center of the cabinet
Fan trays 4, 5, 6, 7	Shelf 1 at rear-bottom of the cabinet
Fan trays 8, 9, 10, 11	Shelf 2 at front-center of the cabinet
Fan trays 12, 13, 14, 15	Shelf 3 at front-bottom of the cabinet

Every fan tray has an opposite counterpart on a different shelf, but on the same side; one tray sucks air into the cabinet, the other blows it out. For example, Tray 3 of Shelf 0 is opposite Tray 7 of Shelf 1, and Tray 9 of Shelf 2 is opposite Tray 13 of Shelf 3.

OPTIONS

The following options are supported:

–p **on**	Turn on power to all fans.
–p **off**	Turn off power to all fans. See also –l and –t, which you can use with `–p off` to turn off specific fans. In case of conflict, the fans specified by –l take precedence over those specified by –t. See the examples, below.
–l **{front | rear}**	Turn off power to only the specified front or rear fan shelf and, if applicable, to the fan trays specified by –t. Use this option only with `power –p off`.

–t *FanTrayList*	Turn off power to only the specified fan trays and, if applicable, the shelf specified by –l. *FanTrayList* is a space-separated list of fan tray numbers expressed as integers, 0 to 15 inclusive. Use this option only with `power –p off`.
–s **{nominal | fast}**	Set the speed of all fans to either nominal or fast, as specified. All fans in the system run at the same speed. Once you change the fan speed, it remains in effect until changed again via the `fan –s` command. The default speed is nominal.

EXAMPLES

EXAMPLE 1 Setting the Fan Speed

The following command sets the speed of all fans to fast.

```
fan -s fast
```

EXAMPLE 2 Turning Off the Fans

The following command turns off all fans on the two front fan shelves. The command specifies both a fan shelf (–l) and a list of fan trays (–t), and the former takes precedence.

```
fan -p off -l front -t 0 1 2
```

SEE ALSO

`cbs`(1M), `edd`(1M), `hostinfo`(1M), `hostview`(1M), `snmpd`(1M)

NAME

hostinfo – display system information

SYNOPSIS

hostinfo –F|–S|–h|–p|–t

DESCRIPTION

`hostinfo` sends to standard out certain real-time host-relevant data. It queries the `snmpd`(1M) daemon for this information, which `snmpd`(1M) fetches from the Sun Enterprise 10000 system specified in the SUNW_HOSTNAME environment variable.

OPTIONS

The following options are supported:

`–F`	Display fan configuration and status.
`–S`	Display the contents of the signature block for each configured processor.
`–h`	Display the state and signature of each configured processor.
`–p`	Display power supply readings of all system boards, control boards, and fan trays, and of the centerplane and I/O cabinet.
`–t`	Display the ambient temperature of the Sun Enterprise 10000 system specified in the SUNW_HOSTNAME environment variable, and the temperature readings of its control boards and system boards. All temperatures are shown in Centigrade.

FILES

The following files are supported:

`$SSPETC/snmp/Ultra-Enterprise-10000.mib`

SEE ALSO

`snmpd`(1M)

NAME

hostint – interrupt processor, dump kernel core

SYNOPSIS

hostint [–p *proc*] [–f] [–v]

DESCRIPTION

`hostint` causes a hung domain to panic, dump then reboot. When used without its –p option, `hostint` sends an interrupt signal to the boot processor of the domain specified in the SUNW_HOSTNAME environment variable to cause a kernel core dump. If `hostint` fails to panic the boot processor, it will attempt to interrupt another processor in the domain.

Note - Use `hostint` only when the domain is hung and the hang is not detected by `edd`(1M), which would normally do an automatic reboot. You cannot panic an IDN domain unless you use the –f option; however, it can cause a cluster arbstop.

Other steps, both more and less impactful, are available to unhang the domain. Try each of the following procedures, in the order shown, until the hang is fixed:

1. Attempt a reboot from any functional login on the domain. If this works, the domain was not fully hung. Core is not dumped.
2. Attempt to break into OpenBoot from the console by sending a break. You can do so via either `netcon`(1M) or `netcontool`(1M). Once in OpenBoot, issue the command `sync`, which causes a panic dump and reboot.
3. Execute `hostint`. If this step is successful, the domain panics, dumps, then reboots.
4. Execute `hostreset`. If this step is successful, you can then initiate a bringup if the system does not automatically boot.
5. If steps 1 through 4 fail, execute `bringup`(1M) with its –f option.

OPTIONS

The following options are supported:

–p *proc*	Send the interrupt signal to the specified processor rather than the boot processor.
–f	Force a panic in an IDN domain.
–v	Verbose Mode. Display information to stdout.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

FILES

The following files are supported:

`$SSPETC/snmp/Ultra-Enterprise-10000.mib`

SEE ALSO

`bringup`(1M), `netcon`(1M), `snmpd`(1M)

NAME

hostreset – reset a hung domain

SYNOPSIS

hostreset

DESCRIPTION

`hostreset` resets the domain specified by the SUNW_HOSTNAME environment variable, executing an external interrupt (XIR).

Other, less-traumatic steps are available to unhang the domain. Try each of the following procedures, in the order shown:

1. Attempt a reboot from any functional login on the domain. If this works, the domain was not fully hung. Core is not dumped.
2. Attempt to break into OpenBoot from the console by sending a break. You can do so by using either `netcon`(1M) or `netcontool`(1M). Once in OpenBoot, issue the command `sync`, which causes a panic dump and reboot.
3. Execute `hostint`(1M). If this step is successful, the domain panics, dumps, then reboots.
4. Execute `hostreset`. If this step is successful, you can then initiate a bringup if the system does not automatically boot.
5. If steps 1 through 4 fail, execute `bringup`(1M) with its `–f` option.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`bringup`(1M), `netcon`(1M), `edd`(1M), `hostint`(1M)

NAME

hostview – system monitor Graphical User Interface

SYNOPSIS

hostview

DESCRIPTION

Hostview is the Graphical User Interface (GUI) that makes it easy for you to display and modify the Sun Enterprise 10000 system's configuration and environmental data. Hostview spawns SSP commands.

You can start and run Hostview in the background of an SSP Window by issuing the following command:

```
hostview &
```

OPTIONS

Hostview accepts all the standard X Toolkit command line options.

ENVIRONMENT VARIABLES

To run Hostview from another display, make sure the DISPLAY environment variable is set for that display and the environment variable SUNW_HOSTNAME is set to either the platform name or domain hostname. The following example shows one way to do so.

rlogin -l ssp *ssp_host* setenv DISPLAY *local_host*:0 **hostview &**

For more information about Hostview and its various screens, see the *Sun Enterprise 10000 SSP User Guide*.

SEE ALSO

Xll(7) Xlib Resource Manager documentation

NAME

hpost – control and sequence POST through JTAG

SYNOPSIS

hpost [–?][–?postrc|–?blacklist|–?level|–?verbose]

hpost [–aCcfnqs] [–D [*boardmask,*][*path*]] [–D *"comment"*] [–e*xcode*] [–g[*path* | none]][–H*boardmask,refproc*] [–i[*proc*]] [–J*bus_mask*] [–JJ*bus_mask*] [–l*level*] [–p*proc*] [–Q*proc*[*,skipmask*]] [–R{*redlist_file* | none}][–v*level*] [–X{*blacklist_file* | none}][–W[c]] [–Z*proc*]

CAUTION

This command is for use by your service provider only. Improper use can cause catastrophic operating system failure.

DESCRIPTION

The POST (Power On Self Test) program probes and tests the components of uninitialized Sun Enterprise 10000 hardware, configures what it deems worthwhile into a coherent initialized system, and hands it off to OBP (OpenBootProm). `hpost` is the SSP-resident executable program that controls and sequences the operations of POST through the IEEE 1149.1 JTAG scan interface between the Enterprise 10000 domain and the SSP.

Unless –n is the first argument on the `hpost` command line, `hpost` reads an optional file, `.postrc`, and executes the directives in that file before it begins operation with the host (see `postrc`(4)). `hpost` first looks for `.postrc` in the current directory (.). If it does not find it there, `hpost` looks in `$SSPVAR/etc/`*platform_name*`/$SUNW_HOSTNAME`.If it does not find `.postrc` there, it looks in the user's home directory, `$HOME`. **Exception:** If the current directory is `$HOME`, the first element of the search path (.) is skipped. If `hpost` does not find `.postrc` it proceeds without it.

The following command provides a terse listing of `.postrc` file syntax and directives:

```
hpost ?postrc
```

OPTIONS

The following information pertains to the `hpost` options:

- Numerical option arguments are generally assumed to be decimal, but may be given as hex if preceded by `x` or `0x`. Exception: arguments identified as a *mask* are assumed hex.
- *board* is a system board number in the range 0 to 15, inclusive.
- `proc` is a processor number in the range 0 to 63, inclusive, that corresponds to the physical location as (board * 4 + processor_module #).
- Hyphenated flags and their arguments are *not* separated by a space.
- The question mark that precedes the first group of options can be replaced by the letter h. For example, –`?postrc` = –`hpostrc`.

–?	Display a terse description of arguments.
–?postrc	Display a terse description of the .postrc file.
–?**blacklist**	Display a terse description of blacklist(4) / redlist(4) file syntax.
–?**level**	Display a terse description of level number meanings.
–?**verbose**	Display a terse description of verbose number meanings.
–a	Run POST at the alternate diagnostic level. –a is intended as a convenience to be used during certain error reboot scenarios where you want to run POST at a higher-than-normal diagnostic level. If you provide a value for alt_level in the .postrc file, –a attempts to use it as the alternate diagnostic level. If you do not provide a value for alt_level in .postrc, –a attempts to use its default value, which is higher than the diagnostic level used by default when –a is not specified. However, a never causes the diagnostic level to be lower than it would be if –a is not specified. You cannot use –a and –l at the same time. See also –l and postrc(4).
–C	Do the initial configuration of the centerplane. If –C is not invoked, hpost assumes that one or more other domains in the same physical platform have already been configured and are running. hpost probes the centerplane to determine the bus configuration, the only configuration it considers. If hpost cannot determine a valid configuration from this probe, it immediately fails. If –C is invoked, hpost assumes nothing else is running on this platform, and tests and configures the system, including the centerplane, in the bus configuration with the highest figure of merit in the domain in which it is run.

–c	Parse (as hpost normally does) the .postrc, blacklist(4) and redlist(4) files, reporting any syntax errors, then immediately exit. See postrc(4), blacklist(4) and redlist(4).
–D[*boardmask*,] [*path*]	Invoke a special mode of POST that scans out state from the host, dumps it to a binary file, then immediately exits. If *path* is not specified, a default path is used. (The comma after *boardmask* is always required.) hpost cannot examine the dump files, but your service provider and Sun Microsystems can. If specified, *boardmask* is a 20-bit mask of the parts of the machine to be included in the dump. Bits 0 through 15 correspond to the system boards, bits 16 and 17 to the two half-centerplanes, and bits 18 and 19 to the two control boards. If *boardmask* is not specified, hpost assumes its value is FFFFF and includes all parts of the system in the dump. However, hpost then applies a heuristic decision algorithm to recognize system boards that are not present or powered off, and it does not include these boards in the dump file. No such editing is done if a *boardmask* is specified. If –D is specified and the standard input of hpost is a terminal, hpost prompts for a one-line comment to be embedded in the file. If you simply press RETURN, the comment line is empty. If you also specify –d, the prompt for a comment is suppressed. See –d.
–d *"comment""*	Insert the specified comment into the dump file. A space must separate this option from its argument. Quotes are not necessary for a single-word comment. The comment is inserted regardless of the standard input means (terminal, script, etc.). See –D.
–e*xcode*	Print a one-line description of a POST exit code to standard output.
–H*boardmask,refproc*	Run a special mode of POST that prepares one or more boards to be DR Attached to a running

domain. *boardmask* is a 16-bit mask of the boards on which POST is to run. *refproc* is the current master CPU of the target domain, which must be on a different board.

Caution: Do not specify –H on the command line; it should be invoked only by another program as one step in the DR process. See dr(1M).

–i**[***proc***]** Execute in interactive mode. This option is used for debugging. If `proc` (a single processor number) is specified, only that processor is used and `hpost` begins an interactive session. If `proc` is not specified, all processors are used and `hpost` begins a simpler interactive session, just stopping before each phase of debug to ask whether it should execute that phase or continue to the next.

–J*bus_mask* Use the specified bus configuration rather than selecting the configuration based on testing. This support mode of POST, for use only by Sun Microsystems, interprets *bus_mask* as a 6-bit binary mask for the desired bus configuration. The 2 most-significant bits are the data buses, the 4 least-significant bits are the address buses. 3F tells `hpost` to use all buses.

–JJ*bus_mask* This option is similar to –J, but does only the JTAG initialization.

–l*level* Set the diagnostic level for this run of POST. Acceptable values are in the range 7 to 127, inclusive; the default level is 16. `–?level` displays a brief summary.

–n Inhibit reading of the `.postrc` file. If present, this argument must appear first.

–p*proc* Use the specified processor as the preferred boot processor. This request is ignored if the specified processor is not in the final configuration.

–Q*proc* **[**,*skipmask***]**	(Quick POST) Make POST read configuration information from bootbus sram in the indicated processor, then reconfigure, with minimum testing, the system described. This option is intended for quick recovery from software crashes. It is considered to have failed if the specified configuration cannot be effected. The *skipmask* option enables the caller (usually another program) to skip certain steps in the initialization process that it determines are unnecessary. The goal is an even faster recovery. The bits in *skipmask* have the following meaning: 0 - Perform all phases of the reconfiguration (the default) 0x0001 - Do not clear processor IMU tags and do not disable the IMU 0x0002 - Do not clear processor DMU tags and do not disable the DMU 0x0004 - Do not clear processor instruction caches 0x0008 - Do not clear processor data caches 0x0010 - Do not clear processor external caches 0x0020 - Do not clear CIC duplicate tags (DTAGs) 0x0040 - Do not initialize I/O controllers 0x0080 - Do not clear memory
–q	Quiet mode. All screen output is discarded.
–R **{***redlist_file* **| none}**	Use the specified redlist file, or no redlist file (none), rather than the default redlist file. See the Caution, below.
–s	Divert all screen output to *syslog* with appropriate priorities.
–v*level*	Set the message verbosity level. Acceptable values are in the range 0 to 255, inclusive; the default is 20. –?verbose displays a brief summary.
–X **{***blacklist_file* **| none}**	Do not use the default path. none suppresses use of any blacklist(4) file. See the Caution, below.

–W **[c]**	Clear any Recordstop state that may be present in the current domain, and attempt to reenable centerplane ASIC recording. If an Arbstop or other fatal error condition is detected in this domain, it is reported and the Recordstop clear attempt is abandoned. The result of this operation is reported in the `hpost` exit code. If a Recordstop condition exists in a different domain the attempt to reenable centerplane recording will fail. The failure will be reported, but will not cause this –W operation to be considered failed. Use the optional `c` flag with –W for a domain that is part of an Inter-Domain Network (IDN). Invoke –Wc for one and only one domain of the IDN; `hpost` clears Recordstop in that domain and all other domains in the IDN. This mode of `hpost` is normally invoked by an SSP event detection daemon after it creates a state dumpfile. See the –D option.
–Zproc	(Zip POST) Make POST read configuration information from bootbus sram in the indicated processor, and perform only the JTAG initialization of the system to effect the configuration described. This option is intended to allow dumps of the software state after a crash; this state would be destroyed by the normal POST configuration process. `hpost -Z` is considered to have failed if the specified configuration cannot be effected. **Caution:** Be careful when using –R or –X in a production system; other SSP software will not know that a nonstandard `blacklist`(4) or `redlist`(4) file is in use by POST.

USAGE

POST is normally executed by supervisory scripts or programs on the SSP, but may be invoked from the command line in engineering development, manufacturing, or field service applications.

FILES

The following files are supported:

`./.postrc` - Local POST configuration file

`$HOME/.postrc` - User's default POST configuration file

`$SSPVAR/etc/`*platform_name*`/$SUNW_HOSTNAME/.postrc` - Hostname-specific default POST configuration file

`$SSPVAR/etc/`*platform_name*`/blacklist` - Default blacklist file (see –X)

`$SSPVAR/etc/`*platform_name*`/redlist` - Default redlist file (see –R)

`$SSPVAR/adm/$SUNW_HOSTNAME/post/post`*mmdd.hhmm*`.log` - Default log file (see –g)

`$SSPVAR/adm/$SUNW_HOSTNAME/xfstate`*mmdd.hhmm.ss* - Default dump file (see –D)

`$SSPVAR/adm/$SUNW_HOSTNAME/hpost.lock` - POST lock file

`$SSPOPT/release/Ultra-Enterprise-10000/*/*/*/hostobjs/*.elf` - Path to download (host-resident) POST executable files.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

DIAGNOSTICS

An exit status in the range 0 to 63, inclusive, indicates successful configuration. The exact value is the number of the processor whose bootbus sram contains the POST-to-OBP handoff structures that describe the configuration. Values outside the range 0 to 63 indicate that the system was not configured. (Those values outside the range have been codified for use by Sun Microsystems, and the information is described in a restricted-use header file. However, the –e option will provide a description of any value.)

SEE ALSO

`obp_helper`(1M), `redx`(1M), `blacklist`(4), `postrc`(4), `redlist`(4)

NAME

machine_server – multi-purpose server

SYNOPSIS

machine_server

CAUTION

Do not execute this daemon manually. It is automatically invoked by the SSP startup script and periodically monitored for re-start.

DESCRIPTION

The `machine_server` daemon performs the following functions:

- Services TCP port registration requests from `netcon_server`(1M) and UDP port registration requests from the Sun Enterprise 10000 SNMP agent, `snmpd`(1M).
- Fields `netcon_server`(1M) and `snmpd`(1M) port lookup requests from various SSP client programs and returns the port number.
- Ensures that error messages are routed to the proper messages file.

Each SSP can run only one instance of `machine_server` at a time.

FILES

The following files are supported:

`$SSPVAR/pids/machine_server.pid` - process ID file

`/etc/services` - reserved port specification

SEE ALSO

`netcon_server`(1M), `snmpd`(1M)

NAME

moveboard – detach a board from its domain and attach it to a specific domain

SYNOPSIS

moveboard –b *board_number* –d *domain* [–h] [–r *retry_count*] [–q] [–t *timeout*]

DESCRIPTION

`moveboard` attempts to detach the board specified by *board_number* from its current domain (referred to as the source domain) and then attach it to the domain specified by *domain* (referred to as the target domain). `moveboard` performs several Dynamic Reconfiguration (DR) detachment and attachment operations that write progress messages to the platform log on the SSP and to standard output.

If `moveboard` fails, it writes an error message to standard error, attempts to roll back the state of the specified board to be in its original domain, and returns a nonzero exit status. If the specified board is already detached and powered on, `moveboard` attempts to attach it to the specified domain. If the specified board is already in the specified domain, `moveboard` returns an exit status of zero, indicating success.

The `–r` option specifies the maximum number of retry attempts, *retry_count*, in the event that an attach operation fails. The `–t` option specifies a time period in seconds, *timeout*, between retry attempts. If you specify the number of retries, but no timeout, there is no delay between retry attempts. If you specify a timeout, you must also specify the number of retries.

To be eligible for detachment, a board must be powered on and in one of the following states:

- In a domain
- In an intermediate detach state (after the drain stage but before complete detachment)

If a board in the source domain is in an unknown or intermediate attach state, the `moveboard` operation fails and returns an exit status of 5.

If there is another board in the same source domain as the board specified by *board_number*, and if that board is in an intermediate detach state, the `moveboard` operation fails and returns an exit status of 6. The `moveboard` operation cannot be completed until that board in the source domain is completely detached.

If a board in the target domain is in an unknown or intermediate detach state, the `moveboard` operation fails and returns an exit status of 11.

OPTIONS

The following options are supported:

`–b` *board_number*	System board number, 0 through 15, to be moved.
`–d` *domain*	The domain to which the board is to be moved.

–h	Write a usage message for `moveboard` to standard error.
–r *retry_count*	The number of retry attempts to be performed in the event a move fails.
–t *timeout*	Time in seconds before next retry attempt.
–q	Quiet mode, which means informational messages are not written to standard output.

EXAMPLES

The following example moves system board 2 from its current domain to a domain named `xf3-b2`. If the first attempt to move the board fails, two retries will be made, if necessary, with a wait time of 15 minutes between retries.

```
moveboard -b 2 -d xf3-b2 -r 2 -t 900
```

EXIT STATUS

The following exit values are returned:

0	Successfully moved the specified board to the specified target domain.
1	Command syntax error.
2	Failed to acquire DR lock.
3	Failed to release DR lock.
4	Failed to initialize DR communications.
5	Board or source domain is not eligible for detachment.
6	A board in the source domain is in an intermediate detach state.
7	DR daemon will not allow the detach because it will either remove the last processor from the domain or drop the domain below some minimum memory requirement.
8	Failed during memory drain operation.
9	The drain operation failed to make progress over some period of time.
10	Failed in attempting to complete the detach.
11	Board or target domain not eligible for attachment.
12	A board in the target domain is in an intermediate attach state.

13 Failed in initial attachment stage.

14 Failed in complete attachment stage.

15 Failed to dynamically link with DR library.

ERROR MESSAGES

If `moveboard` fails, it writes diagnostic messages to standard error.

FILES

The following files are used by `moveboard`:

`/var/opt/SUNWssp/.ssp_private/domain_config`

Software domain configuration

`/var/opt/SUNWssp/adm/messages`

Platform log file

NOTES

`moveboard` prevents more than one DR attach or detach command from executing concurrently.

If the specified board is in an intermediate detach state, `moveboard` completes the detach operation and attempts to attach the board to the specified domain.

SEE ALSO

`complete_attach`(1M), `complete_detach`(1M), `dr`(1M), `drain`(1M), and `init_attach`(1M) in the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual*

`addboard`(1M), `deleteboard`(1M), and `showusage`(1M) in the *Sun Enterprise 10000 SSP Reference Manual*

Sun Enterprise 10000 Dynamic Reconfiguration User Guide

Sun Enterprise 10000 SSP User Guide

NAME

netcon – network console

SYNOPSIS

netcon [–d|–f|–g|–l|–r] [–e *esc_char*]

DESCRIPTION

The `netcon` command creates a remote connection to the domain host console program, making the SSP window in which the command is executed a Console Window for the domain specified by the SUNW_HOSTNAME environment variable. See also `netcontool`(1M), a menu-driven program that executes `netcon`.

Many Domain Console Windows (also called *console sessions*) can be open simultaneously on systems throughout the network, and all can read corresponding domain host output. But only one at a time can have write permission for a particular domain. Write permission is in either Unlocked or Locked mode.

Unlocked Write permission is not very secure, as it is taken away if another console session is started via **`netcon –g`**, **`netcon –l`** or **`netcon –f`**, or if `~@`, `~&` or `~*` is executed in another Domain Console Window. (Tilde commands are described in the USAGE section, below.)

Locked Write permission is more secure. It can be taken away only if another console session is opened via **`netcon –f`**, or if `~*` is executed in another Domain Console Window. In both cases, the new console session is an Exclusive Session, and all other sessions are killed.

If you execute `netcon` with no options at a time when no other console sessions for that domain have write permission, your Domain Console Window comes up with Unlocked Write permission. If another session already has write permission, your session comes up Read Only.

`netcon` can utilize either the standard network interface or the JTAG interface for Domain Console Window communication. You can manually toggle between these interfaces via the tilde-equals (~=) command. Doing so is useful if the network becomes inoperable, in which case `netcon` console sessions hang.

Note that automatic Domain Console Window resizing is not supported across the JTAG interface, so input/output performance is slower than it is over the network interface.

OPTIONS

All options are mutually exclusive except the –e, which can be used with any of the others. If you specify more than one of the other options, `netcon` with the option that carries the highest precedence, as follows: –f, –l, –g, then –r.

–d Turn on debug mode. Do not use this option without first consulting your service advisor.

–e *esc_char*	Change the default escape character (~) to *esc_char*, where *esc_char* can be any character except the following: @ # ^ & * () = | ’[; " ’< > . ?
–f	(Force option) Open a Domain Console Window with Locked Write permission, terminate all other open sessions, and prevent new sessions from being opened. This option starts Exclusive Session mode. Use it only when you need exclusive use of the console; e.g., for a private debugging session. **Note:** To restore multiple-session mode, either release the lock (by using ~^) or terminate the **netcon –f** session (by using ~.). See USAGE, below.
–g	(Grab option) Open a Domain Console Window with Unlocked Write permission. If another session has Unlocked Write permission, the new Domain Console Window takes it away. If another session has Locked permission, this request is denied and a read-only session is started.
–l	(Lock option) Open a Domain Console Window with Locked Write permission. If another session has Unlocked Write permission, the new Domain Console Window takes it away. If another session has Locked permission, this request is denied and a read-only session is started.
–r	(Read Only option) Open a Domain Console Window with Read Only permission.

USAGE

In a Domain Console Window, a tilde (~) that appears as the first character of a line is interpreted as an escape signal that directs `netcon` to perform some special action, as follows:

~#	Break to OBP or `kadb`
~.	Disconnect and exit the `netcon` session.
~@	Acquire Unlocked Write permission; see –g.
~^	Release write permission.
~?	Show status of all open console sessions and the communication path currently in use.

~=	Toggle the communication path between the network and JTAG interfaces. You can use ~= only in Private mode (by using ~*).
~&	Acquire Locked Write permission; see –l. You may issue this signal during a read-only or Unlocked Write session.
~*	Acquire Locked Write permission, terminate all other open sessions, and prevent new sessions from being opened; see –f. To restore multiple-session mode, either release the lock or terminate this session.

NOTES

To enable `netcon` session logging, uncomment the local1.debug line in the `/etc/syslog.conf` file, then force `syslogd`(1M) to reread its configuration file by executing `kill -HUP 'cat /etc/syslog.pid'`. When `netcon` session logging is enabled, `netcon`(1M) session output is recorded in the `/var/opt/SUNWssp/adm/`*domain_name*`/netcon` file.

If you use a **`kill –9`** command to terminate a `netcon` console session, the window or terminal in which the `netcon` command was executed goes into raw mode, and appears hung. To escape this condition, type `^j`, then `stty sane`, then `^j`.

In the Domain Console Window, `vi`(1) runs properly and the escape sequences (tilde commands) work as intended *only* if the environment variable TERM has the same setting as that of the Netcon Window. For example, in the window in which the `netcon` command was run, TERM is set to `xterm`, TERM in the Domain Console Window must be set to `xterm`, too.

If you get the following error message, `bringup`(1M) has not been run on the domain:

`netcon_server is not running for` *domain_name*.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`netcontool`(1M)

`boot`(1M), `cvcd`(1M) in the *SunOS Reference Manual*

NAME

netcon_server – network console server daemon

SYNOPSIS

netcon_server [–S] –p *boot_proc*

netcon_server [–S] –r

CAUTION

This daemon is executed by `bringup`(1M). Do not execute it on the command line.

DESCRIPTION

`netcon_server` manages communications between the various SSP Console Windows (also called netcon connect sessions) and the specified boot processor of the corresponding domains specified by each SSP Window's SUNW_HOSTNAME environment variable.

When the domain is up, `netcon_server` acts as a relay between the various Domain Console Windows and the `cvcd`(1M) daemon running on the domain side. When the domain is down, it is a relay between the Domain Console Windows and OBP.

`netcon_server` behaves like a daemon; it spawns a copy of itself and the parent process exits, effectively disconnecting itself from the controlling terminal.

OPTIONS

The following options are supported:

–S	Suspend **`netcon session`** logging if syslogd's local1 facility is enabled in `/etc/syslog.conf`.
–p *boot_proc*	Use the specified boot processor.
–r	Restart `netcon_server`, obtaining the domain boot processor from the `snmpd`(1M) daemon. Execute **`netcon_server –r`** only when `netcon_server` has died and the host is either up or at the OBP prompt.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`bringup`(1M), `ssp_startup`(1M), `netcon`(1M), `thermcal_config`(1M) in the *Sun Enterprise 10000 SSP 3.2 Reference Manual*

`cvcd`(1M) in the *Solaris Reference Manual for SMCC-Specific Software*

NAME

netcontool – network console tool

SYNOPSIS

netcontool

DESCRIPTION

`netcontool` is a GUI-driven interface to the `netcon`(1M) command. It accepts configuration information necessary for console connection to the domain specified by the SSP's SUNW_HOSTNAME environment variable, makes that connection, and lets you press buttons that correspond to the escape sequences recognized by the Netcon program. These sequences are described in `netcon`(1M).

When you execute the `netcontool` command, the Domain Console Window is displayed with the following buttons enabled:

`Exit` — Press to exit `netcontool`.

`Configure` — Press to display the Console Configuration dialog box (described below).

`Connect` — Press to begin the Netcon session. If more configuration information is required before the connection can be made, a message to that effect is displayed. In this case, press the `Configure` button.

The Console Configuration dialog box, which is displayed when you select `Configure`, lets you specify the following information:

`Terminal Type` — Press the appropriate button to select the terminal type. Note that, in the Domain Console Window, `vi`(1) runs properly and the escape sequences (tilde commands) work as intended *only* if the terminal type specified here is the same as the TERM environment variable of the Domain Console Window. For example, if TERM for the Domain Console Window is `xterm`, it must be `xterm` (shown as `Xterm`) here. The default is `Xterm`.

`Session Type` — Press the `Read Only Session`, `Unlocked Write` (**netcon** **–g**), `Locked Write` (**netcon** **–l**) or `Exclusive Session` (**netcon** **–f**) button.

The default setting is Read Only. However, your new session comes up this way only if at least one other session is already running; if yours is the first console session to be started, it comes up in Unlocked Write mode, even if you selected the `Read Only` button.

When you click the `Done` button, the `Connect` button in the window becomes available, and you can press it to start the session. If you have not supplied sufficient configuration information, `netcontool` displays an error message that states the problem.

When you press `Connect`, `netcontool` executes the `netcon`(1M) program, the Domain Console Window appears in the specified terminal type, and all buttons in the `netcontool` Window except `Configure` and `Connect` become available. These buttons correspond to the escape sequences described in the `netcon`(1M) man page. The escape sequences cannot be entered from the keyboard while `netcontool` is in use.

The Domain Console Window acts independently of the `netcontool` Window during window-management operations.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`netcon`(1M)

NAME

obp_helper – download OpenBoot to system memory

SYNOPSIS

obp_helper [–eivqr] [–o*Filename*] [–d*Filename*] [–m*boot_proc*] [–A {on | off}][–D {on | off}][*boot-arguments*]

DESCRIPTION

Note - obp_helper is normally executed by the bringup(1M) script, not on the command line. You may execute it on the command line *only* as **obp_helper –e** or, under certain conditions, **obp_helper –r**.

---

obp_helper provides an environment in which OpenBoot can run. It then downloads OpenBoot and provides time-of-day and EEPROM simulation services to it.

obp_helper executes until OpenBoot exits or a bringup(1M) or sys_reset(1M) is executed.

obp_helper behaves like a daemon; it spawns a copy of itself and the parent process exits, effectively disconnecting itself from the controlling terminal.

OPTIONS

The following options are supported:

–e	Display the current settings of EEPROM options alterable by obp_helper.
–i	Invalidate the boot arguments in EEPROM.
–v	Execute in verbose mode.
–q	Execute in quiet mode.
–r	Restart obp_helper. Execute **obp_helper –r** only if OpenBoot is already executing on the system, but obp_helper has died or otherwise disappeared.
–o *filename*	Download the specified file instead of the standard OpenBoot executable.
–d *filename*	Download the specified file, not the standard download_helper executable.
–m *boot_proc*	Make the processor selected by *boot_proc* the boot processor.
–A **[on | off]**	Enable (on) or disable (off) OpenBoot auto-boot mode. The –A option has the same effect as a setenv auto-boot?

`[true | false]` command to OpenBoot; it alters the state of the `auto-boot?` flag in the OpenBoot simulated EEPROM. When auto-boot is enabled, the `bringup`(1M) command brings up the system completely, using the appropriate boot disk as determined by other OBP variables. When auto-boot is disabled, `bringup`(1M) completes in such a manner that the OBP prompt (`OK`) is displayed on the `netcon`(1M) Window.

`–D` **`[on | off]`** Enable (on) or disable (off) OpenBoot diagnostic mode. This option has the same effect as the `setenv diag-switch? [true | false]` command to OpenBoot; it alters the state of the `diag-switch?` flag in the OpenBoot simulated EEPROM.

boot-arguments Pass boot arguments verbatim to the OpenBoot boot command. You can specify standard OpenBoot device aliases such as **`disk`** and **`net`**. These arguments affect the current boot only; subsequent boot commands use the standard OpenBoot *boot-device* and *boot-file* arguments.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

SEE ALSO

`bringup`(1M)

`boot`(1M) in the *SunOS Reference Manual*

NAME

power – control power

SYNOPSIS

power

power [–sb *list*] [–cb 0|1] [–csb *csb_list*] [–p *p_list*]

power –off [–f] [–all] [–sb *sb_list*] [–csb *csb_list*] [–p *p_list*] [–cb 0|1] [–d|–q]

power –off –B [–f] [–d|–q]

power –off –ps *ps_list* [–d|–q] (Service Providers Only)

power –on [–all] [–sb *sb_list*] [–csb *csb_list*] [–p *p_list*] [–d|–q]

power –m

power –m *margin_list* –sb *sb_list* [–d|–q] (Service Providers Only)

power –m *margin_list* –s –sb *sb_list* [–d|–q] (Service Providers Only)

power –m *margin_list* [–d|–q] (Service Providers Only)

power –m *margin_list* –s [–d|–q] (Service Providers Only)

power –v [–all] [–sb *sb_list*] [–csb *csb_list*] [–p *p_list*] [–ps *ps_list*] [–d|–q]

DESCRIPTION

The power command enables you to view and control certain aspects of the Sun Enterprise 10000 system's power supplies and I/O cabinets. You can use it to power on and off the power supplies of individual boards or I/O cabinets, and to view or set their voltage margins for over-voltage/under-voltage monitoring.

When used with no options, power displays the status of all 48-volt power supplies, I/O cabinet power supplies, control board and centerplane support board power supplies, and all the individual system board power supplies. See EXAMPLES, below.

The power status of the centerplane support boards and the system boards is displayed as follows:

```
Good 48V Bulk Power Supplies:                    0 2 3 4
Number of Good 48V Bulk Power Supplies:          4 (no redundancy)
Required 48V Power Supplies for 8 System Boards:  4
Number of Good Peripheral Cabinet Power Supplies: 0
 Centerplane Support Board Average Voltages (V):
 CSB#     5VDC Vcc HK     3.3VDC Vdd HK       3.3VDC Vdd Core
```

```
----  -----------------  -------------  -------------------------
 0     5.047     5.037       3.501       3.311    3.301    3.297
 1     5.017     5.017       3.491       3.303    3.302    3.302

System Board Average Voltages (V):
     3.3VDC    5VDC     3.3VDC       VDC        5VDC
SB#   Vdd     Vcc HK    Vdd HK     Vdd Core     Vcc
---  -------  -------  --------   ----------  --------
 0    3.301    5.086    3.407        2.601      5.008
 1    3.300    5.022    3.423        2.601      5.013
 2    3.300    5.015    3.439        2.607      5.010
 3    3.301    5.035    3.434        2.599      5.013
 4    3.290    5.015    3.423        1.897      5.005
 7    3.301    5.052    3.418        2.604      5.003
 8    3.296    5.059    3.429        2.596      5.000
 9    3.302    5.025    3.412        2.594      5.008

Control Board Average Voltages (V):
       5VDC        5VDC       3.3VDC      5VDC Vcc     5VDC
CB#    Vcc        Vcc HK      Vdd HK     Peripheral   Vcc Fans
---  --------    --------    ---------   ----------  ---------
 0    5.113       5.049       3.420       5.194       5.091
```

where:

- `5.0VDC` and `3.3VDC` in the first column are the nominal voltage for each power supply. `VDC` means Voltage Direct Current. In the centerplane listing `5.0VDC HK` is repeated twice and `3.3VDC Vdd` is repeated three times, indicating that the voltage `5.0VDC HK` is measured at two different locations and `3.3VDC Vdd` is measured at three.

  In the system board listing, the value of `Vdd Core` depends on the type of processor installed. This value can be between 2.5VDC and 3.7VDC, inclusive. JTAG accesses an analog-to-digital converter to read the resistor on the processor module to determine the actual value.

- `HK` means HouseKeeping power. This power is on whenever the 48-volt power is being supplied to the board.

- `Vdd` is a common abbreviation for +3.3 volt DC power.

- `Vcc` is a common abbreviation for +5.0 volt DC power.

- The values in the second column are the actual voltage readings provided by the analog-to-digital converter accessed via JTAG.

Note - When housekeeping power is applied to the machine, the power supplies on the control boards are turned on automatically. The system boards and centerplane support boards are powered on following the execution of the `power` command. You cannot power off a system board within a domain that is a member of an InterDomain Network (IDN); you must first use `domain_unlink`(1M) to unlink such a domain. Use `power –f` to force a power off even if the board is part of an IDN; however, it can cause a cluster arbstop. You cannot power off the entire Sun Enterprise 10000 host cabinet if there is an IDN.

OPTIONS

See the SYNOPSIS line, above, for acceptable option combinations.

`–sb` *sb_list*
: Display, turn on, turn off or validate power to the power supplies for the specified system boards. *sb_list* represents a space-separated list of integers, 0 to 15, inclusive, that refer to system board numbers. This option is also used for margining; see `–m`.

Note - When `–sb` is specified, the `power` command does not control housekeeping power on the system board. Also, you cannot power off the power supply of a system board that is part of a domain that is running the operating system.

`–cb 0|1`
: Turn off, display status of, or validate power to the power supplies for the specified control board, either 0 (rear) or 1 (front). Note that an active control board can turn off only a non-active control board for removal; the `power` command does not turn them both off. After the secondary control board has been powered off, do not attempt to use the `power` command to obtain the status of the control board; the registers cannot be read. Visually inspect the board LEDs before removing it from the system.

`–csb` *csb_list*
: Turn on, turn off or validate power to the power supplies for the specified centerplane support boards, where *csb_list* is a space-separated list of integers, each either 0 (rear) or 1 (front).

Note - When –csb is specified, the power command does not control housekeeping power on the centerplane support board.

–p *p_list*

Turn on, turn off or validate power to the power supplies for the specified remotely controlled AC sequencer and its attached peripherals. *p_list* represents a space-separated list of integers, 0 to 4, inclusive, that refer to the switches (remote power control units) on the host that control the power distribution units for the peripherals. For more information, service providers can see the *Ultra Enterprise 10000 System Overview*, a printed document that is part of the service document set.

–all

Turn on, turn off or validate power to all system boards, including centerplane support boards. If you specify –all with –sb, –csb or both, –all takes precedence.

–d

(Debug Mode) Display verbose trace messages. –d and –q are mutually exclusive.

–on

Turn on the system boards, including centerplane support boards, that belong to the domain specified by the SUNW_HOSTNAME environment variable. See also the options –sb, –csb and –all.

power -on turns on the specified board's power supplies, then queries the system to determine whether it has a sufficient amount of power. If so, those power supplies remain on; if not, power turns them off again and displays a message.

–off **or** –f –off

Turn off the individual power supplies of the centerplane support boards and the system boards that belong to the domain specified by the SUNW_HOSTNAME environment variable. See also the options –sb, –csb, –cb, –p, –ps, and –all.

power -off turns off the specified board's power supplies, but only if affected domains are

not running the operating system. If an affected domain is running the operating system, `power` displays a message to that effect and does *not* turn off the power. This process gives you the opportunity to gracefully shut down the domain before turning off its power. To immediately turn off power to the domain, overriding this protective feature, use `–f–off`.

`–off –B`

Turn off power to the entire Sun Enterprise 10000 host cabinet and remotely controlled I/O cabinets. To restore power you must manually turn on the switches (up to four per cabinet) on the AC input modules that feed the 48-volt power supplies.

`–ps` *ps_list*

Display, turn off (service providers only), or validate power to the specified 48–volt power supplies, where *ps_list* is one or more integers that represent the 48-volt power supplies -0 to 7, inclusive - to be validated or disabled.

The Sun Enterprise 10000 dual power grid provides up to 16 bulk power supplies. The power supplies are divided into two grids, where each grid contains up to 8 power supplies. The power supplies in the primary grid are associated with power supplies in the secondary grid. For example, a relationship exists between power supplies 0 and 8, 1 and 9, and 2 and 10. If you turn off a single power supply using the `power` command, both the specified power supply and its associated power supply are turned off. All bulk power supplies can be turned off by powering off power supplies 0 through 7.

The command `power –off –ps` *ps_list* displays the following message:

```
WARNING: This system requires x 48V power
supplies for a current load of y system boards
(at N+1 redundancy). You are disabling z out of
w 48V power supplies. This system will be able
to normally operate a load of u system boards
(including N+1 redundancy). Continue? (y/n)
```

where

x is the number of required power supplies

y is the number of system boards in the system

z is the number of power supplies to be powered off

w is the number of power supplies in the cabinet

u is the number of system boards that will be supported after power to the specified system boards is turned off

Warning: Powering off too many 48–volt power supplies may crash the system. Use the information displayed in the warning message and the chart below to determine whether turning off the specified power supplies is safe.

```
1 sys bd needs 4 ps  9  sys bds need 7 ps
2 sys bds need 4 ps  10 sys bds need 7 ps
3 sys bds need 5 ps  11 sys bds need 7 ps
4 sys bds need 5 ps  12 sys bds need 7 ps
5 sys bds need 5 ps  13 sys bds need 8 ps
6 sys bds need 6 ps  14 sys bds need 8 ps
7 sys bds need 6 ps  15 sys bds need 8 ps
8 sys bds need 6 ps  16 sys bds need 8 ps
```

`–q` (Quiet Mode) Inhibit any traffic to standard out. `–q` and `–d` are mutually exclusive.

`–v` (Validate Mode) Inspect the individual power supplies of system boards and centerplane support boards that belong to the domain specified by the SUNW_HOSTNAME environment variable to determine whether they are currently enabled.

The `–v` option is intended for use only by scripts. It generates a 0 exit code if all the target devices are powered up. Note that `–v` merely inspects the power control bits; it does not check voltages.

You can use `–v` with `–sb` or `–csb`, or with `–p` to specify the devices to be validated.

`–m` Display the current margin values saved in the `ssp_resource`(4) file.

`–m` *margin_list* `–s –sb` *sb_list*

For use by service providers only.

–m *margin_list* –sb *sb_list*

For use by service providers only.

The **power -m** *margin_list* **-s sb | csb** command sets, in the ssp_resource(4) file, the values specified in *margin_list* for all system boards (sb) . If –s is also specified, the change takes place immediately. Otherwise, the change takes effect when the boards are next powered on. Note that no hyphen is required before sb if –s is used.

Warning: Margining power supplies more than +/- 5 percent may cause damage to the hardware. Note that Vcore values for system boards may not be margined.

In both cases (**power -s** and **power -sb**), specify *margin_list* as a space-separated list of elements in one of the following forms:

supply.+*margin*

supply.*margin* (same as +)

supply.-*margin*

where *margin* is an integer between 1 and 5, inclusive, that represents the percentage of margin the supply voltage should be adjusted from nominal, and *supply* is a keyword Vcc, Vdd or Vcore that represent system board voltages as follows:

Vcc, 5 volts

Vdd, 3.3 volts

Note that the two forms of this command use margin levels differently. Immediate margining uses the margin to adjust the *current* power level by the specified margin percentage. Automatic margining uses the margin to adjust the *nominal* voltages by the specified margin percentage.

EXAMPLES

EXAMPLE 1 Applying Power

Apply power to power supplies on system boards 0 and 2.

```
power -on -sb 0 2
```

EXAMPLE 2 Turning Off the Power Supplies

Turn off power supplies to all system boards in the domain specified by the SUNW_HOSTNAME environment variable.

```
power -off
```

EXAMPLE 3 Changing the Margin Voltage Value

Change the margin voltage value for all system boards as follows: increase Vcc by 2 percent, decrease Vdd by 2 percent. Note that this change in margining will take effect when the system boards are next powered on.

```
power -m Vcc.+2 Vdd.-2 -s sb
```

EXAMPLE 4 Increasing Target Values

Increment by 3 percent the target values of Vcc, and Vdd supplies on system boards 5, 6 and 7. The change occurs immediately, while these system boards remain on.

```
power -m Vcc.+3 Vdd.+3 -sb 5 6 7
```

EXAMPLE 5 Applying Power to Peripherals

Apply power to the peripherals remotely controlled by remote power control units (or switches) 2 and 3 on the host.

```
power -p 2 3 -on
```

NAME

redx – remote emulation debugger

SYNOPSIS

redx [–chnlq?] [*file* [*arg*]...]

redx [–q] –x*redx_interpreter_cmd*...

CAUTION

Only your service provider should execute this command except during software installation and updates. If installation instructions provided by Sun Microsystems tell you to use this command, do so exactly as instructed. **Improper use can cause catastrophic operating system failure.**

DESCRIPTION

`redx` is a debug and maintenance support program for the Sun Enterprise 10000 system. It is associated with the POST program, but is generally useful for other low-level hardware and firmware debugging.

A command interpreter, `redx` reads commands interactively, from script files, or when used with the –x option, from the invoking command line. It features extensive on-line help. To access this help from within the program, type `help` or `?`, and `redx` lists available command names and explains the other help facilities.

At start-up `redx` reads the optional start-up script, `.redxrc`, if it is present in the current directory. If `redx` does not find it in the current directory, it looks in the user's home directory. If `redx` cannot find the script in either directory, or if the –n option is specified, the script is not used.

OPTIONS

Dash arguments can be grouped, as in –qn, or issued separately, such as –q –n. They can appear in any order, except that –x, when used, must be the last dash option. Numeric arguments are assumed to be decimal, or hex if prefaced with `0x` or `x`.

–c — Do not use the curses library functions for input/output. The command history and scrollable output window functions are not available in this mode. –c makes `redx` usable when only simple tty connections to the SSP are available.

–h — Print help listing.

–l — Initiate in Local mode, with no access to JTAG. Use this option for parsing functions, or for safe, offline examination of hardware dump files.

–n — Do not read the startup file, `.redxrc`.

–q — Use quiet mode, suppressing normal screen output.

–x *redx_interpreter_cmd ...* Execute the specified interpreter command (or commands) from the command line, then exit. Characters special to the shell must be escaped (see sh(1)). This option implies –n.

–? Print help listing.

EXAMPLES

EXAMPLE 1 Displaying Option Explanations

```
redx -?
```

EXAMPLE 2 Displaying More Information

```
redx -x ?
```

NAME

showusage – display DR and board resource information

SYNOPSIS

showusage –b *board_number* –r *report_type* [–r *report_type* ...]

showusage –b *board_number* –s

DESCRIPTION

`showusage` used with the –r option displays information on `stdout` about board resources, system resources, and DR. It presents the information in a tabular format.

`showusage` used with the –s option returns an integer exit status indicating the state of the specified board. If the specified board is in a domain, `showusage` writes the domain name to `stdout`. The –s option is intended for use in ADR scripts. The –s option can also be used to determine the eligibility of a board for an ADR operation before performing any ADR operations.

OPTIONS

The following options are supported:

–b *board_number* — System board number, 0 through 15, for which status information is to be obtained.

–r *report_type* — The type of information to be displayed. *report_type* can be one of the following:

`CPU`	Show processor information for the board.
`DRAIN`	Show the progess of any active drain operation.
`IO`	Show information for any devices attached to this board.
`OBP`	Display the board configuration as OBP sees it. You can use the OBP display on a board that has been initially attached, but not yet completely attached. The OBP display might not be as accurate as the CPU, MEM, and IO displays for boards that are in use.
`MEM`	Show the memory configuration for this board.
`UNSAFE`	Display all suspend unsafe devices that are open in the

specified domain. If this option is used, the –b option is ignored.

You must use multiple –r options to specify more than one type of report. For example, to see processor and device information for board 12, you would use the following command:

showusage -b 12 -r CPU -r IO

–s

Report board state as exit status and write domain name on stdout. If the specified board does not currently belong to a domain, no text is written to stdout.

EXIT STATUS

The showusage command returns different sets of exit status values depending on the command option specified.

The following exit values are returned when the –s option is specified:

0 Board is not present and is not in a software domain. A software domain is an active domain listed in the domain_config file, but the CPU is not running.

1 Board is not present and is in a software domain.

2 Board is present, powered off, and not in a software domain.

3 Board is present, powered on, and not in a software domain (the board is eligible for attachment).

4 Board is present, powered off, and in a software domain.

5 Board is unconfigured (present, powered on, and in a software domain, but not in a hardware domain). A hardware domain is a booted domain.

6 Board is configured (present, powered on, and in both a software and hardware domain), but not in a detachable DR state. For a description of the conditions that can prevent board detachment, see the *Sun Enterprise 10000 Dynamic Reconfiguration User Guide*.

7 Board is configured (present, powered on, and in both a software and hardware domain), and possibly detachable. For a description of

conditions that can prevent board detachment, see the *Sun Enterprise 10000 Dynamic Reconfiguration User Guide*.

`15` Failed to dynamically link with DR library.

`255` `showusage` syntax error.

The following exit values are returned when the `–r` option is specified:

`0` Successfully displayed report.

`255` `showusage` syntax error.

ERROR MESSAGES

If `showusage` fails, it writes diagnostic messages to `stderr`.

SEE ALSO

`dr`(1M) and `drshow`(1M) in the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual*

`addboard`(1M), `deleteboard`(1M), and `moveboard`(1M)

Sun Enterprise 10000 Dynamic Reconfiguration User Guide

Sun Enterprise 10000 SSP User Guide

NAME

sigbcmd – send commands from SSP to domain

SYNOPSIS

sigbcmd [–svif] [–t *retries*] [–p *proc*] *cmd*

sigbcmd [–vif] [–t *retries*] –p *proc* –r

CAUTION

Only authorized service providers should use this command. **Improper use may disrupt activity on the domain, causing unexpected results and, possibly, catastrophic operating system failure.**

DESCRIPTION

The SSP utility `sigbcmd` performs one of the following actions: sends the given command, *cmd*, to the Sun Enterprise 10000 domain specified in the SUNW_HOSTNAME environment variable; displays status of a mailbox (–s); or resets the mailbox flag of a processor to `empty` (–r). If *cmd* expects a response, `sigbcmd` also retrieves that response from the domain.

`sigbcmd` uses the signature block mailbox interface to communicate with the kernel for execution of commands. This mailbox resides in BBSRAM.

OPTIONS

`sigbcmd` executes only one command, then exits. Therefore, the –r, –s and *cmd* options are mutually exclusive. If you specify more than one at a time, only one is executed, in the following order of precedence: –r, –s, *cmd*.

cmd — An ASCII name representing the command to send to the domain. Currently supported commands are `obp` (make domain enter OBP) and `panic` (make domain panic).

Note - You cannot send a `panic` or `obp` command to an IDN domain unless option –f was used. You must first use `domain_unlink` to unlink such a domain.

–s — Display the current status – the `flag`, `cmd`, and `data` fields – of the mailbox. Only the first 32 bytes (SIGB_MBOX_SIZE/2) of the data field are displayed.

–v — Execute in verbose mode, printing some debugging information. The default is terse mode.

–i — Query the user to confirm execution of the command. The default is no query.

–f — Force execution, overriding the mailbox flag. If –f is not set (the default) and something is detected in the mailbox (*i.e.*, flag != SIGB_MBOX_EMPTY), `sigbcmd` does not write out the mailbox command. Using –f will force it to do so. This option will force execution against an IDN domain.

However, a panic in an IDN domain can cause a cluster arbstop.

`-r` Reset to `empty` the mailbox flag of the target processor specified by `-p` *proc*.

`-t` *retries* If a response is expected, read the mailbox for that response the specified number of times, where *retries* is an integer, with one-second pauses between reads. If a response is never detected, an error message appears stating so. The default is 10 retries, which takes about 10 seconds.

`-p` *proc* Make the specified processor the target of the given command. The default is to pick any present and available processor, usually the boot processor.

EXAMPLES

EXAMPLE 1 Going to OBP

Make target domain enter OBP.

```
sigbcmd -p 4 obp
```

EXAMPLE 2 Sending Commands

Send a `domain panic` command to processor 4. Pause or acknowledgement from the user before actually sending the command.

```
sigbcmd -i -p 4 panic
```

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

NOTES

This command assumes the domain's operating system is coherent enough to receive the mailbox command (via NMI level 15 interrupt).

Be careful: this utility does not check for permissions.

NAME

snmpd – SNMP proxy agent for Sun Enterprise 10000 platform

SYNOPSIS

snmpd [–h] [–p *port*] [–X]

CAUTION

Do not execute this daemon manually; it is automatically invoked. snmpd(1M) is a platform type server. An instance of snmpd is periodically invoked by the SSP startup script for each platform being managed and monitored for restart.

DESCRIPTION

snmpd(1M) is an SNMPv1 (Simple Network Management Protocol, Version 1) proxy agent for the Enterprise 10000 system. See the Sun Enterprise 10000 MIB (Management Information Base) definition file, *$SSPETC/snmp/Ultra-Enterprise-10000.mib*, for a defined list of objects serviced by this agent.

snmpd(1M) supports the SNMP requests set(1), get(1) and getnext(1) .

Each instance of snmpd(1M) allocates its own communications port and registers with machine_server(1M). Other SNMP managers wishing to communicate with a specific SNMP agent must also use machine_server(1M) to find the port number assigned to the agent. Some number of UDP ports are reserved for this purpose in the /etc/services file.

For Enterprise 10000 MIB information that is SSP-resident, snmpd(1M) relies on the file access daemon, fad(1M), to supply the information. Enterprise 10000 MIB information that corresponds to system resources is retrieved by using the control board server, cbs(1M).

OPTIONS

The following options are supported:

–h — Prints a short summary of snmpd(1M) command line options.

–p *port* — Use the specified UDP port for incoming SNMP requests. If –p is not specified, the first instance of snmpd(1M) allocates UDP port 5050 for communication. Subsequent instances allocate arbitrary UDP ports. If Solstice Enterprise Agent (SEA or snmpdx) is not running, snmpd(1M) also listens to the standard SNMP port number, 161. machine_server(1M) daemon for port assignment.

–X — Do not daemonize snmpd. Keep the process running in the foreground. This option is used for debugging. When this option is not being used, snmpd(1M) sends stdout and stderr output to /var/tmp/snmdp/snmpd.log.

FILES

The following files are supported:

`$SSPETC/snmp/agt/Ultra-Enterprise-10000.snmpd.cnf` - configuration file

`$SSPETC/snmp/Ultra-Enterprise-10000.mib` - Sun Enterprise 10000 MIB definition file

`$SSPETC/snmp/Ultra-Enterprise-10000.dat` - Sun Enterprise 10000 MIB data file

`$SSPVAR/pids/snmpd-`*platform_name*`.pid` - process ID file

`$SSPETC/snmp/sea/ssp_snmp.reg` - SEA registration for SSP 3.2`snmpd`(1M)

`$SSPETC/snmp/sea/ssp_snmp.rsrc` - SEA resource file for SSP 3.2`snmpd`(1M)

`/var/tmp/snmpd/core` - core file, if any, for `snmpd`(1M)

`/var/tmp/snmpd/snmpd.log` - log file which stores stdout and stderr messages unless redirected to the screen. The `–X` option redirects stdout and stderr messages to the screen.

`/var/tmp/snmpd/snmpd.stats` - usage statistics file for `snmpd`. This file is generated whenever the snmpd daemon receives the SIGUSR1 signal. (e.g., “kill —USR1 551”, where 551 is the process ID of `snmpd`(1M) .) This file contains the number of gets, sets and traps since the `snmpd`(1M) process began or since it last received a SIGUSR1 signal.

SEE ALSO

`cbs`(1M), `fad`(1M), `machine_server`(1M), `straps`(1M)

NAME

ssp_backup – back up the SSP environment

SYNOPSIS

ssp_backup *directory_name*

DESCRIPTION

`ssp_backup` creates a `cpio`(1) archive of files that maintain the operational environment of the SSP. You can use `ssp_backup` to maintain the same environment on the main SSP and spare SSP by backing up the main SSP and restoring the archive file on the spare SSP with `ssp_restore`(1M). Run `ssp_backup` on the main SSP.

When you change the SSP environment, for example by shutting down a domain, you must run `ssp_backup` again to maintain a current backup file for the SSP.

The name of the backup file is `ssp_backup.cpio`.

If any errors occur, `ssp_backup` writes error messages to `$SSPVAR/ssp_backup.out`.

Note - The size of the backup file can be very large (20 Mbytes or more). You can minimize the size of the backup file by deleting SSP log files or message files that are no longer needed from the `/var/opt/SUNWssp/adm` directory.

OPERANDS

The following operands are supported:

directory_name The name of the directory in which the backup file is to be created.

FILES

The following files are supported:

`$SSPVAR/ssp_backup.out` – `ssp_backup` log file.

SEE ALSO

`ssp_restore`(1M)

Sun Enterprise 10000 SSP User Guide

NAME

ssp_config – configure SSP control boards

SYNOPSIS

ssp_config [spare | cb]

DESCRIPTION

`ssp_config` sets up some SSP configuration files and provides tftpboot information for the control boards. It is automatically executed by the operating system start-up scripts the first time the SSP is booted after the SSP software is installed. You can also execute it manually when logged in as root.

When executed by start-up scripts, `ssp_config` prompts for the name of the system (platform name), number of control boards in the system, the name of control board 0, and the name of control board 1. It asks for the primary control board, and attempts to automatically determine the IP addresses of the control boards by using name services or local files. If it fails in this attempt, it prompts for them, too. The system will not fully initialize until `ssp_config` has the information it needs.

OPTIONS

The following options are supported:

`spare` Configure the spare SSP as the primary SSP. Use this option if your SSP fails and you have a secondary (spare) SSP or other SPARCstation available. For more information, see the Sun Enterprise 10000 Installation section of the *Solaris 7 Sun Hardware Platform Guide*.

`cb` Change the control board setup in the `cb_config`(4) file. Executing `ssp_config cb` initiates a series of prompts that enable you to change which control board is considered the primary control board, or to swap, add or delete a control board. You cannot change the platform name.

Caution: Be sure to refer to the *Sun Enterprise 10000 SSP User Guide* for important information before using **`ssp_config cb`**.

FILES

The following files are supported:

`/etc/inetd.conf`

`$SSPETC/ssp_env.sh`

`$SSPVAR/.ssp_private/cb_config`

`$SSPVAR/etc/`*platform_name*`/edd.emc`

`$SSPVAR/etc/`*platform_name*`/edd.erc`

`/tftpboot`

/.SSP_DEFAULTS

SEE ALSO

`bringup`(1M), `cb_config`(4)

NAME

ssp_downgrade – prepares the SSP and Sun Enterprise 10000 for downgrading to a previous SSP release.

SYNOPSIS

ssp_downgrade [–h] [–v]

DESCRIPTION

`ssp_downgrade` prepares the SSP for downgrading to a previous release of SSP software. You must run this command if you are running SSP 3.3 and are restoring or installing a previous SSP release (SSP 3.2 or earlier).

OPTIONS

The following options are supported:

`–h`	Prints a short summary of `ssp_downgrade` (1M).
`–v`	Runs in verbose mode, where some debugging information is printed.

EXAMPLES

EXAMPLE 1 Performing an SSP Downgrade

```
xf4-ssp:xf4% ssp_downgrade
Setting control board 0 Vcc Fan voltage.
Control board 0 Vcc Fan voltage set to 5.091013
Setting control board 1 Vcc Fan voltage.
Control board 1 Vcc Fan voltage set to 5.093463
```

EXIT STATUS

The following exit values are returned:

`0`	Downgrade successfully completed.
`Non-zero`	An error occurred.

SEE ALSO

`ssp_startup`(1M)

NAME

ssp_restore – restore the SSP environment

SYNOPSIS

ssp_restore [*filename*]

DESCRIPTION

`ssp_restore` restores the operational environment of the SSP from a backup file created by `ssp_backup`(1M). You can use `ssp_restore` to maintain the same environment on the main SSP and spare SSP by backing up the main SSP with `ssp_backup`(1M) and restoring the archive file on the spare SSP with `ssp_restore`. You can also use `ssp_restore` to restore the SSP environment after the SSP software has been installed on a new disk.

Any required translation of configuration files, such as `/var/opt/SUNWssp/.ssp_private/sso_to_domain_hosts`, is automatically performed.

If any errors occur, `ssp_restore` writes error messages to `$SSPVAR/ssp_restore.out`.

Note - If the main SSP environment has changed since the backup file was created, for example by shutting down a domain, you must run `ssp_backup`(1M) again to maintain a current backup file for the SSP. You must run `ssp_restore` with this new backup file to main the same environment on both SSPs.

OPERANDS

The following operands are supported:

filename The name of the backup file that was created by `ssp_backup`(1M). If the specified file is not in the current directory, the filename must contain the full pathname for the file.

FILES

The following files are supported:

`$SSPVAR/ssp_restore.out` – `ssp_restore` log file.

SEE ALSO

`ssp_backup`(1M)

Sun Enterprise 10000 SSP User Guide

NAME

ssp_startup – invoke SSP daemons

SYNOPSIS

ssp_startup

DESCRIPTION

Note - Do not execute this script from the command line. The `ssp_startup` script is normally initiated automatically on each reboot as defined in `/etc/init.d/ssp` and linked to `/etc/rc2.d/S99ssp`

The `ssp_startup` script starts the following Sun Enterprise 10000 system daemons. These daemons must be started in the proper order for your particular configuration, which may differ from the example shown here.

- `edd`(1M)
- `cbs`(1M)
- `snmpd`(1M)
- `machine_server`(1M)
- `fad`(1M)
- `straps`(1M)
- `obp_helper`(1M)
- `netcon_server`(1M)

`ssp_startup` then monitors and restarts them, as necessary.

When `ssp_startup` completes, it leaves a message similar to the following in the `$SSPLOGGER/messages` file:

```
Startup of SSP programs complete
```

You can use `domain_create`(1M) to create the domain, then execute `bringup`(1M) to bring it up.

FILES

The following files are supported:

`$SSPETC/ssp_startup.main` - controls which programs are started during initial start-up

`$SSPETC/ssp_startup.restart_main` - controls which programs are are monitored and restarted if necessary

NAME

ssp_unconfig – deconfigure the SSP

SYNOPSIS

ssp_unconfig

DESCRIPTION

Note - `ssp_unconfig` is primarily used by Sun Microsystems just prior to shipping systems. Only customers who are very familiar with their Sun Enterprise 10000 server should use it and, even then, only after consulting with their service advisor.

`ssp_unconfig` is used to reconfigure the SSP in a new environment, or to change host names and IP addresses or platform names. `ssp_unconfig` prompts for the name of the domain for which the first eeprom.image was created by use of the serial ID. See the `–s` option of `sys_id`(1M).

Note - Before executing this command make certain that all domains have been halted and removed by using `domain_remove`(1M), and save the eeprom.image files. These files may not be restored to their corresponding domains when those domains are recreated.

FILES

The following files are supported:

`$SSPETC/ssp_env.sh`

`$SSPVAR/.ssp_private/cb_config`

`$SSPVAR/.ssp_private/domain_config`

`$SSPVAR/.ssp_private/domain_history`

`$SSPVAR/.ssp_private/ssp_to_domain_hosts`

`$SSPVAR/etc/`*platform_name*`/edd.erc`

`$SSPVAR/etc/`*platform_name*`/edd.emc`

`/.SSP_DEFAULTS`

`/tftpboot`

SEE ALSO

`ssp_config`(1M), `domain_remove`(1M)

`halt`(1M), `shutdown`(1M) in the *SunOS Reference Manual*

NAME

straps – SNMP trap sink server

SYNOPSIS

straps

CAUTION

Do not execute this server command manually. It is automatically invoked and monitored for restart by the SSP startup script.

DESCRIPTION

The `straps` server listens to the SNMP trap port for incoming trap messages and forwards received messages to all connected clients.

The SNMP trap port is UDP port number 5051. The port is a privileged port and can be opened only once; therefore, `straps` is necessary to serve multiple SNMP-manager-type applications on the SSP. Essentially all clients that need to receive SNMP traps must establish a connection to the `straps` server. They do so by connecting to an AF_UNIX domain stream socket and getting the SNMP trap packet in raw binary form.

Each SSP can run only one instance of `straps` at a time.

FILES

The following files are supported:

`$SSPETC/snmp/straps` - stream socket device file

`$SSPVAR/pids/straps.`*pid* - process ID file

SEE ALSO

`snmpd`(1M)

NAME

sys_clock – display/change/set system clock frequencies

SYNOPSIS

sys_clock

sys_clock –m *ctrl_board_num* [–i *ic_freq*] [–p *proc_multiple*] [–s]

sys_clock [–i *ic_freq*] [–p *proc_multiple*] [–s]

CAUTION

When executing this command on the command line, do so in its default form, with no options, unless you are an expert user. These options attempt to reset or change target frequencies and such attempts can result in poor performance or a system crash.

DESCRIPTION

When executed with no command-line options, `sys_clock` displays both target and actual clock frequencies of the interconnect and JTAG, and the processor-to-interconnect clock ratio.

The SSP package is distributed with a set of target clock frequencies intended to maximize performance of the Sun Enterprise 10000 system. These target frequencies are maintained by the `snmpd`(1M) agent. Normally, centerplane support and system boards receive their clock values from one of the two possible control boards. The default source clock is the primary control board.

`sys_clock` is executed automatically during system initialization.

OPTIONS

The following options are supported:

Note - When executed with its –i or –p options, `sys_clock` only changes the target settings maintained by the `snmpd`(1M) agent. These settings are passed to the Sun Enterprise 10000 system the next time `sys_clock –s` is executed. To use these options to immediately change the settings of the Sun Enterprise 10000 system, include –s on the same command line.

–i *ic_freq*	Set the target interconnect clock frequency to the specified value, in hertz. Valid values range from 40000000 (40.0Mhz) to 120000000 (120.0 Mhz), inclusive.
–m *ctrl_board_num*	Program the boards with the number of the control board that is providing the system clock.
–p *proc_multiple*	Set the target processor-to-interconnect clock ratio to the specified values, which is one of the following: `two-to-one`, `three-to-one`, `three-to-two`, or `five-to-two`.

–s	Like –m, Program the boards with the number of the control board that is providing the system clock; unlike –m, also set the hardware clock frequencies to the target frequencies maintained by the snmpd(1M) agent. **sys_clock –s** is executed automatically each time the Sun Enterprise 10000 system is powered on.

EXAMPLES

EXAMPLE 1 Displaying the Target and Actual System Clock Frequencies

```
% sys_clock
Current Clock Frequencies:
--------------------------
Interconnect: 83.25 Mhz
Processor:   333.01 Mhz
JTAG:          5.20 Mhz

Targeted Clock Frequencies:
---------------------------
Interconnect: 83.33 Mhz
Proc Clock Ratio: three-to-two
JTAG:          5.00 Mhz
```

Note - The JTAG clock frequencies will only be displayed on control boards that support the display of JTAG frequencies.

EXAMPLE 2 Modifying the Target Clock Frequencies

```
% sys_clock -i 83333333 -p two-to-one
```

EXAMPLE 3 Setting the System Clock Frequencies

```
% sys_clock -s
```

ERRORS

`Error: Failed to get SSP host name for domain xf4.`

The snmpd(1M) agent may not be running. This message might be returned when no domains are configured and the SUNW_HOSTNAME environment variable is set to a domain name. Try setting SUNW_HOSTNAME to the platform name.

FILES

The following file is supported:

`$SSPETC/snmp/Ultra-Enterprise-10000.mib` - the Enterprise 10000 MIB definition file

SEE ALSO

cbs(1M), snmpd(1M), ssp_resource(4)

NAME

sys_id – display or change system ID in SSP IDPROM

SYNOPSIS

sys_id

sys_id –d [–f *filename*]

sys_id –x [–f *filename*]

sys_id –m [*yy*] *yymmddhhmm* [–f *filename*]

sys_id –k *key* –s *serial_no* [–f *filename*]

sys_id –k *key* –h *host_id* [–f *filename*]

DESCRIPTION

sys_id displays or changes the host ID and Ethernet number on an Sun Enterprise 10000 SSP's IDPROM and saves the changes in the EEPROM image file.

The management and relationship of the system ID (machine serial numbers, host IDs, and Ethernet numbers) are kept in a file on the SSP as part of the IDPROM portion of the EEPROM image.

The IDPROM portion of the EEPROM image is encrypted to prevent tampering. However, you can change the encrypted ID in the EEPROM image file for system ID modifications. To do so you must first obtain a *key* from Sun Microsystems. The key restricts you from changing the system ID to anything other than the designated machine serial number.

To ensure uniqueness, Enterprise 10000 system host ID and Ethernet numbers are derived from the FOA / serial number. The Enterprise 10000 system's serial number comes from a block of numbers assigned to Sun Microsystems by SPARC International.

SYSTEM ID

The sys_id command must be executed on a SSP where the host name and the $SUNW_HOSTNAME environment variable match the ones provided to Sun (when a key is requested). The existing host ID (obtained from the IDPROM) and the –s *serial_number* or –h *host-id* must also match.

OPTIONS

When invoked with no arguments sys_id displays the host ID as specified in the EEPROM image file (located in the default location) in decimal format.

–d — Display all IDPROM fields. The date and time of manufacture are displayed in Universal Coordinated Time, the peculiar acronym for which is UTC.

–m **[***yy***]***yymmddhhmm* — Set the manufacturing date and time of the IDPROM as the specified year (the first two digits

are optional), month, day, hour and minute. If the first *yy* is not specified and the year is specified as a two-digit number between 69 and 99, inclusive, `sys_id` interprets the year as 1969 through 1999. If the *yy* is not specified and the year is specified as a number between 00 and 68, the year is expanded to 2000 through 2068.

`sys_id -m` can be executed only once.

`–f` *filename*

Override the default location of the EEPROM image file and use *filename* as the alternate location and name. `–f` uses the specified location regardless of whether a read or write of the file is being performed. The default location and name of the EEPROM image is `$SSPVAR/etc/`*platform* `name/$SUNW_HOSTNAME/eeprom.image`

`–k` *key*

Change the system ID. Before executing `sys_id` with this option you must obtain from Sun the *key*, which restricts where and how this command can be run. The `–k` option is the only way to get or change IDs for extra domain hosts (multiple domains). Values between `0x80A66000` and `0x80A66FFF` must be used for extra domains.

When executing `sys_id` with the `–k` option, you must also specify either `–s` or `–h`.

`–s` *serial_no*

Change the system ID. When executed with the `–s` option, `sys_id` changes the serial number portion of the host ID entry in the IDPROM to the specified serial number. You must obtain this number from Sun. You can then specify the serial number in either decimal or hex format. To specify it in hex, use a prefix of `0x` or `0X`. Valid serial numbers are in the range `0xA65000-0xA65FFF`.

`–h` *host*

Change the host ID. When executed with the `–h` option, `sys_id` changes the host ID portion of the host ID entry in the IDPROM to the specified host ID number. You must obtain this number from Sun. You can then specify the host ID number in either decimal or hex format. To specify it in hex, use a prefix of `0x` or `0X`. Valid

host ID numbers are in the range `0x80A65000-0x80A66FFF`.

`-x` Display the serial number in hex format.

NAME

sys_reset – reset the domain

SYNOPSIS

sys_reset [–F|–f]

DESCRIPTION

`sys_reset` resets all system boards that reside within the domain specified in the SUNW_HOSTNAME environment variable. If only one domain contains all system boards, `sys_reset` executes a full system reset, which resets the centerplane and all boards except the control board.

This command does not do the requested reset if the domain you are attempting to reset is running the operating system.

ENVIRONMENT VARIABLES

The environment variable SUNW_HOSTNAME must be set to the name of the domain.

OPTIONS

The following options are supported:

–F **or** –f — Force a system reset, even if the domain is running the operating system.

NAME

thermcal – read or write thermistor calibration

SYNOPSIS

thermcal [–r | –w] –b sb –n *sb_number*

thermcal [–r | –w] –b cp –n *cp_number*

CAUTION

Do not execute this command. It is for exclusive use by Sun Engineering and authorized service providers.

DESCRIPTION

`thermcal` reports the thermistor calibration values in the EEPROMs for ASICS on the system board and centerplane.

OPTIONS

The following options are supported:

–b **sb** –n *sb_number*	Read or write to the specified system board, where *sb_number* is an integer `0` to `15`, inclusive, that represents the system board number.
–b **cp** –n *cp_number*	Read or write to the specified centerplane, where *cp_number* is an integer, either `0` or `1`, that represents the centerplane number.
–r	Read the thermistor calibration. This is the default.
–w	Write the thermistor calibration. Only Sun Microsystems personnel are authorized to use this option. To properly calibrate the board you must first power it off (see `power`(1M)), let it sit idle for at least 30 minutes, then run `thermcal -w`. `thermcal` powers the board back on when invoked to calibrate.

NAME

thermcal_config – create SSP thermistor calibration data file

SYNOPSIS

thermcal_config

thermcal_config –u

thermcal_config –b sb –z*n*

thermcal_config –b csb –z*n*

CAUTION

This command is executed automatically by edd(1M). It is not normally executed on the command line.

DESCRIPTION

When executed with no options, `thermcal_config` generates the SSP configuration file, `thermcaldata.tcl`. This TCL file contains the calibration values for non-precision, discrete thermistors used on ASIC components on system boards and centerplane halves. The SSP uses this file for ASIC temperature query.

OPTIONS

The following options are supported:

–u — Update the existing `thermcaldata.tcl` file with any changes.

–b **sb** –z *n* — Update only the specified system board with any changes; ignore all other changes. *n* is the system board number, 0 to 15.

–b **csb** –z *n* — Update only the specified centerplane support board with any changes; ignore all other changes. *n* is the centerplane support board number, 0 or 1.

FILES

The following file is supported:

`$SSPVAR/etc/`*platform*`name/cbobjs/thermcaldata.tcl`

SEE ALSO

edd(1M)

CHAPTER

SSP File Formats

NAME

Intro – SSP 3.3 files, etc.

DESCRIPTION

This section contains miscellaneous man pages for files, scripts, etc. that execute in the SSP environment.

LIST OF FUNCTIONS

`blacklist`**(4)**	list of system resources not to be booted
`cb_config`**(4)**	list of machines managed by the SSP
`cb_port`**(4)**	list of communication ports for cbe
`domain_config`**(4)**	list / description of configured domains
`domain_history`**(4)**	list / description of removed domains
`edd.emc`**(4)**	event monitor file
`edd.erc`**(4)**	event response file
`fad_files`**(4)**	file access daemon files
`postrc`**(4)**	hpost 1M properties file
`redlist`**(4)**	list of system resources not to be touched
`ssp_resource`**(4)**	SSP processes resource file
`ssp_to_domain_hosts`**(4)**	hostname / domainname file

NAME

blacklist – list of system resources not to be booted

DESCRIPTION

`$SSPVAR/etc/`*platform_name*`/blacklist` is an ASCII file that enables the system administrator (or root) to restrict, from the SSP, the configuration of the host system. It lists components POST cannot use at boot time. POST reads the `blacklist` file before preparing the system for booting, and passes along to OBP a list of only those components that have been successfully tested; those on the blacklist are excluded.

In the `blacklist` file:

- Keywords are not case-sensitive.
- Any part of a line that starts with # is a comment.
- Numbers are assumed decimal unless preceded by `0x`, which indicates hexadecimal. Exception: a board number entered as one of [a - f] or [A - F] is assumed hexadecimal.
- Each line has one and only one keyword.
- The same keyword can be used on more than one line.
- Each keyword has one or more arguments. Each argument is shown as an integer and multiple integers are separated by a period.

Keywords

All value ranges shown below are inclusive.

`–sysbd` *board*

Do not test or configure the specified system board, where *board* is an integer, 0 to 15.

`–proc` *board.pmod*

Do not test or configure the specified processor within the specified system board, where *board* is an integer, 0 to 15 and *pmod* is an integer, 0 to 3.

`–abus` *abus*

Do not test or configure the specified address bus, where *abus* is an integer, 0 to 3. The meaning of this command is that the corresponding CIC ASIC on all system boards is marked black. See the keyword `cic`, below.

`–dbus` *dbus*

Do not test or configure the specified 72-bit half of the 144-bit data router, where *dbus* is an integer, 0 or 1. The meaning of this command is that the

corresponding half of the local data router on all system boards is marked black. See the keyword `ldpath`, below.

`–ioc` *board.ioctl*

Do not test or configure the specified I/O controller on the specified system board, where *board* is an integer, 0 to 15 and *ioctl* is an integer, 0 or 1.

`–scard` *board.ioctl.slot*

Do not test or configure the specified I/O adapter card within the specified I/O controller, which is within a system board. *board* is an integer, 0 to 15, *ioctl* is an integer, 0 or 1, and *slot* is an integer, 0 to 3.

Note - The range 0–3 for scard represents the syntactically correct values in the blacklist file, which allows for the possibility of future I/O modules. At this time, the only supported module types have either two SBus cards, or one PCI card, per I/O controller. Blacklisting scards that don't exist is quietly ignored.

`–mem` *board*

Do not test or configure memory on the specified system board, where *board* = 0 to 15.

`–mgroup` *board.group*

Do not test or configure the specified group of memory DIMMs within the specified system board, where *board* is an integer, 0 to 15 and *group* is an integer, 0 to 3.

`–mlimit` *board.group.MBytes*

Restrict memory configuration of the specified group of DIMMs on the specified system board to the specified value (which is less than its actual value), where *board* is an integer, 0 to 15, *group* is an integer, 0 to 3 and MBytes is an integer, 64 or 256. This keyword may be useful in benchmarking where, for example, one might simulate memory with 8MB DIMMs (a 64MB group) on a machine that actually has 32MB DIMMs (a 256MB group). This level of control granularity is required because one board with 256MB memory groups behaves differently than four boards each with 64MB memory groups.

`–cplane` *half_centerplane*

Do not test or configure the specified Enterprise 10000 half-centerplane, which contains two address buses and 72 bits of the global data router, where *half_centerplane* is an integer, 0 or 1. The meaning of this command is equivalent to the combination of the `abus` and `dbus` commands for the buses contained in this half-centerplane.

`pc` *board.pc*

Do not test or configure the specified port controller ASIC within the specified system board, where *board* is an integer, 0 to 15 and *pc* is an integer, 0 to 2.

`xdb` *board.xdb*

Do not test or configure the data buffer ASIC within the specified system board, where *board* is an integer, 0 to 15 and *xdb* is an integer, 0 to 3.

`cic` *board.cic*

Do not test or configure the specified coherent interface controller ASIC within the specified system board, where *board* is an integer, 0 to 15 and *cic* is an integer, 0 to 3. (*cic* corresponds to an address bus on that board).

`ldpath` *board.dbus*

Do not test or configure the specified 72-bit half of the 144-bit local data router within the specified system board, where *board* is an integer, 0 to 15 and *dbus* is an integer, 0 or 1.

EXAMPLES

EXAMPLE 1 A Sample Blacklist File

```
# Sun Microsystems, Inc.
sysbd 3 5 0xA         # Disable system boards 3, 5 and 10.
sysbd 3 5 A           # Disable system boards 3, 5 and 10.
PROC 4.0 6.2          # Disable Processor 0 on System Board 4, and
                      # Processor 2 on System Board 6.
ScarD 3.0.1           # Disable I/O Adapter 1 on I/O Controller 0 on
                      # System Board 3.
mem 2                 # Disable all memory on System Board 2.
mlimit 0xE.2.64       # Restrict use of Memory DIMM Group 2 on System
                      # Board 14 to 64MB.
cIc 1.2               # Disable CIC ASIC 2 on System Board 1.
```

SEE ALSO

`hpost`(1M), `redlist`(4)

NAME

cb_config – list of machines managed by the SSP

CAUTION

Do not edit this file manually. It is automatically maintained by domain-management tools and commands. To make a control board change, use `ssp_config`(1M) with its `cb` option.

DESCRIPTION

The `$SSPVAR/.ssp_private/cb_config` file identifies the machines that are managed by the SSP and contains the names of both the primary and secondary control boards. Each one-line entry in this file represents one machine, and has the following colon-separated fields. (Spaces have been added to the following example for clarity. The file must NOT contain these spaces.)

platform_name : *platform_type* : `cb0_`*hostname* : *status0* : `cb1_`*hostname* : *status1*

where:

platform_name	The name of the Sun Enterprise 10000 machine. This is a logical name only and does not represent a bootable domain.
platform_type	The platform type of the machine. In this release the *platform_type* is always `Ultra-Enterprise-10000`.
`cb0_`*hostname*	The hostname that has been assigned to the control board in Slot 0 of the machine.
status0	A value that indicates whether cb0 is the primary control board. If the value of *status0* is `P`, cb0 is the primary; if its value is anything else, it is not.
`cb1_`*hostname*	The hostname that has been assigned to the control board in Slot 1 of the machine.
status1	A value that indicates whether cb1 is the primary control board. If the value of *status0* is `P`, the *status1* field should be empty. If *status0* is anything other than `P`, *status1* should be `P`. If one of the control board hostnames is missing, the corresponding status field should also be empty.

EXAMPLES

EXAMPLE 1 A Sample `cb_config` File

```
xf2:Ultra-Enterprise-10000:xf2-cb0:P:xf2-cb1:
```

In platform xf2, xf2-cb0 is the control board in slot 0, and it is configured as the primary control board. An alternate control board, xf2-cb1, is in slot 1.

FILES

The following file is supported:

`$SSPVAR/.ssp_private/cb_config`

SEE ALSO

`cbs`(1M), `ssp_config`(1M), `domain_config`(4)

NAME

cb_port – list of communication port for cbe

DESCRIPTION

The `$SSPVAR/.ssp_private/cb_port` file identifies the input/ output communication port number for the Control Board Executive (CBE). CBE is a program that runs on the control board, servicing JTAG and other requests. `cbs`(1M), the control board server, is the only SSP program that communicates with CBE.

This file contains the input/output port number on a single line.

FILES

The following file is supported:

`$SSPVAR/.ssp_private/cb_port`

SEE ALSO

`cbs`(1M)

NAME

domain_config – list / description of configured domains

CAUTION

Do not edit this file manually. It is automatically maintained by domain-management tools and commands. Editing it manually can cause catastrophic system failure.

DESCRIPTION

The `domain_config` file contains a series of one-line entries, each of which represents a currently configured domain that was created by using `domain_create`(1M). `cbs`(1M) uses this file to determine which platform it is to manage. The `domain_status`(1M) command displays the file's contents.

The number of domains that can be listed is unlimited. Each line in the `domain_config` file appears in the following form:

domain_name : *platform_type* : *platform_name* : *os_version* : *idn_info* : *sysbds*

where:

domain_name	The name of a configured domain. This name must be unique across all physical platforms supported by the SSP where this domain configuration file exists.
platform_type	The type of the system on which the domain resides. Currently, the platform type is always `Ultra-Enterprise-10000`.
platform_name	The name of Enterprise 10000 system on which the domain resides.
os_version	The version of Solaris for the Enterprise 10000 that the domain is running (for example, 2.5.1).
idn_info	Identification information use by Inter-Domain Network software to determine which domains are members of the same IDN-based network.
sysbds	The system boards assigned to the domain, listed by board number and separated by spaces.

EXAMPLES

EXAMPLE 1 A Sample `domain_config` File

```
marvin:Ultra-Enterprise-10000:production:2.4.3.9:-1.0:2 4 6 8 10
nebula:Ultra-Enterprise-10000:test:2.5.0.1:-1.0:12 14 15
```

FILES

The following file is supported:

`$SSPVAR/.ssp_private/domain_config`

SEE ALSO

`cbs`(1M), `domain_create`(1M), `domain_remove`(1M), `domain_status`(1M)

`drain`(1M), `init_attach`(1M) in the *Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual*

NAME

domain_history – list / description of removed domains

CAUTION

Do not edit this file manually. It is automatically maintained by domain-management tools and commands.

DESCRIPTION

The `domain_history` file contains a series of one-line entries, each of which represents a domain that has been removed from the system by using the `domain_remove`(1M) command. The `domain_create`(1M) command consults the `domain_history` file to determine whether the domain to be created has existed before.

For a list of currently configured domains, see `domain_config`(4).

Each line in the `domain_history` file appears in the following form:

domain_name : *platform_type* : *platform_name* : *os_version* : *idn_info* : *sysbds*

where:

domain_name	The name of the removed domain.
platform_type	The type of system on which the domain was created. Currently, the platform type is always `Ultra-Enterprise-10000`.
platform_name	The name of Enterprise 10000 system on which the domain was created.
os_version	The version of Solaris for the Enterprise 10000 that the domain was running (for example, 5.7).
idn_info	Not used.
sysbds	The system boards that were assigned to the domain, listed by board number and separated by spaces.

EXAMPLES

EXAMPLE 1 A Sample `domain_history`(4) File

```
nebula : Ultra-Enterprise-10000 : igor : 5.6 : 0.0 : 4 6 8
```

FILES

The following file is supported:

`$SSPVAR/.ssp_private/domain_history`

SEE ALSO

`domain_create`(1M), `domain_remove`(1M), `domain_config`(4)

NAME

edd.emc – event monitor configuration file

DESCRIPTION

The `edd.emc` file is an ASCII file that specifies how the system monitors for certain events. Each system has only one EMC file.

The EMC file is generated from a template file upon invocation of the `ssp_config`(1M) command. The template file resides in `$SSPVAR/.ssp_private/templates/Ultra-Enterprise-10000`, and is named `edd.emc`.

The EMC file contains a series of lines in the following format:

event_type : load_event_monit

where:

event_type — A mnemonic (name string) which corresponds to an event type. See EXAMPLE, below.

load_event_monit — A keyword, either `enabled`, which tells the system to load the event-monitoring script for the event type on the CBE, or `disabled`, which tells it not to.

The fields are separated by a single colon with or without a single space to its right and left. Words or characters that follow a pound sign (#) are treated as comments and are not parsed. The information in the `edd.emc` file is organized as follows:

System Board Temperature Events

System Board Voltage Events

Control Board Temperature Events

Control Board Voltage Events

Centerplane Temperature Events

Centerplane Voltage Events

Centerplane Support Board Temperature Events

Centerplane Support Board Voltage Events

Host Recovery Events

Other Events

System Board Temperature Events

```
sys_brd_temp_norm : enabled    #over-temp readings that go back to normal
sys_brd_temp_warn : enabled    #brd temp which crosses warning threshold
sys_brd_temp_max : enabled     #brd temp which crosses maximum threshold
```

```
sys_brd_temp_911 : enabled     #brd temp which crosses 911 threshold
sys_brd_temp_bad : enabled     #unable to obtain brd temperatures
sys_brd_temp_change : enabled  #delta change in brd temperature
```

System Board Voltage Events

```
sys_brd_volt_norm : enabled    #max/min/bad voltage readings which go normal
sys_brd_volt_max : enabled     #brd voltage which crosses maximum threshold
sys_brd_volt_min : enabled     #brd voltage which crosses minimum threshold
sys_brd_volt_bad : enabled     #unable to obtain brd voltage values
sys_brd_volt_change : enabled  #delta change in brd voltage
```

Control Board Temperature Events

```
cb_temp_norm : enabled    #see system board temperature descriptions
cb_temp_warn : enabled
cb_temp_max : enabled
cb_temp_911 : enabled
cb_temp_bad : enabled
cb_temp_change : enabled
```

Control Board Voltage Events

```
cb_volt_norm : enabled    #see system board voltage descriptions
cb_volt_max : enabled
cb_volt_min : enabled
cb_volt_bad : enabled
cb_volt_change : enabled
```

Centerplane Temperature Events

```
centerplane_temp_norm : enabled     #see system board temperature descriptions
centerplane_temp_warn : enabled
centerplane_temp_max : enabled
centerplane_temp_911 : enabled
centerplane_temp_bad : enabled
centerplane_temp_change : enabled
```

Centerplane Voltage Events

```
centerplane_volt_norm : enabled     #see system board voltage descriptions
centerplane_volt_max : enabled
centerplane_volt_min : enabled
centerplane_volt_bad : enabled
centerplane_volt_change : enabled
```

Centerplane Support Board Temperature Events

```
supp_brd_temp_norm : enabled     #see system board temperature descriptions
supp_brd_temp_warn : enabled
supp_brd_temp_max : enabled
supp_brd_temp_911 : enabled
supp_brd_temp_bad : enabled
supp_brd_temp_change : enabled
```

Centerplane Support Board Voltage Events

```
supp_brd_volt_norm : enabled       #see system board voltage descriptions
supp_brd_volt_max : enabled
supp_brd_volt_min : enabled
supp_brd_volt_bad : enabled
supp_brd_volt_change : enabled
```

Event Types for System Boards, etc.

Certain event types are common among the different components listed above. The following list is organized by event type.

`temp_norm`	When the temperature of a board goes from over-temperature back to normal.
`temp_bad`	This event simply reports that temperature readings were not obtainable.
`temp_change`	When the temperature readings of critical components on the board have changed by a predefined delta (see Predefined Values, below).
`volt_norm`	When the voltage reading of a board goes from a maximum or minimum voltage reading back to normal.
`volt_max`	When the voltage reading of the board crosses a predefined maximum threshold value (see Predefined Values, below)
`volt_min`	When the voltage reading of the board crosses a predefined minimum threshold value (see Predefined Values, below).
`volt_bad`	When the voltage readings are not obtainable.
`volt_change`:	When the voltage readings on the board have changed by a predefined delta (see Predefined Values, below).

IDN Auto-Link Support

Note that the InterDomain Network feature is supported only on the Sun Enterprise 10000.

```
idn_boot : enabled             #cpu signature states indicate idn_boot
idn_halt : enabled                #cpu signature states indicate idn_halt
idn_awol : enabled                #cpu signature states indicate idn_awol
cluster_arbstop : enabled         #idn cluster arbitration stop condition
cluster_recordstop : enabled      #idn cluster record stop condition
```

IDN Event Types

`idn_boot`	When a domain supporting IDN has booted and the respective IDN driver is loaded. This event indicates to edd(1M) that automatic IDN linking of the respective domain with other members within the same IDN is necessary, provided those other member domains have also been booted.
`idn_halt`	When an IDN driver that was previously loaded has been unloaded in the respective domain. This

event works in conjunction with the `idn_boot` event to synchronize automatic IDN linking of domains.

`idn_awol` — When a domain reports that there are some other domains in the IDN that are non-responsive or AWOL. This event indicates to `edd`(1M) to unlink the AWOL domains.

`cluster_arbstop` — When domains within an IDN experience a arbstop. In the standard arbstop event, only state from the boards within the given domain is saved. However, in a `cluster_arbstop`, the state of all boards from all domains within the IDN is saved.

`cluster_recordstop` — When domains within an IDN experience a recordstop. In the standard recordstop event, only state from the boards within the given domain is saved. However, in a `cluster_recordstop`, the state of all boards from all domains within the IDN is saved.

Host Recovery Recovery Events

```
arbstop : enabled            #arbstop condition on a sys brd in a domain
recordstop : enabled         #recordstop condition on a sys brd in a domain
watchdog : enabled           #watchdog condition on a sys brd in a domain
environment_shutdown : enabled
reboot : enabled             #cpu signature states indicate reboot condition
panic1 : enabled             #cpu signature states indicate panic1 condition
panic2 : enabled             #cpu signature states indicate panic2 condition
panic_reboot : enabled       #cpu sig states indicate panic reboot condition
obp_reset : enabled          #cpu sig states indicate OBP reset condition
obpbooting : enabled         #cpu sig states indicate obp/booting condition
heartbeat_failure : enabled  #cpu heartbeat bits indicate heartbeat failure
```

Host Recovery Event Types

`arbstop` — When a system board that belongs to a particular domain experiences an arbitration stop condition.

`recordstop` — When a system board that belongs to a particular domain experiences a record stop condition.

`watchdog` — When a processor (or set of processors) that belongs to a particular domain experiences a watch-dog condition.

`environment_shutdown` — When a processor (or set of processors) that belongs to a particular domain experiences an environmental shutdown condition.

`reboot`	When a processor (or set of processors) that belongs to a particular domain experiences a reboot condition.
`panic1`	When a processor (or set of processors) that belongs to a particular domain experiences a panic1 condition.
`panic2`	When a processor (or set of processors) that belongs to a particular domain experiences a panic2 condition.
`panic_reboot`	When a processor (or set of processors) that belongs to a particular domain experiences a panic reboot condition.
`opb_reset`	When a processor (or set of processors) that belongs to a particular domain experiences an OPB reset condition.
`obpbooting`	When a processor (or set of processors) that belongs to a particular domain goes into OBP booting. This condition occurs when the domain is in the midst of booting.
`heartbeat_failure`	When all the processors that belong to a particular domain experience a heartbeat failure condition.

Other Events

```
signature_change : enabled        #cpu signature has changed states
system_config_change : enabled    #any machine module is inserted/removed
sys_brd_power_on : enabled        #a system board has been powered on
sys_brd_power_off : enabled       #a system board has been powered off
supp_brd_power_on : enabled       #a support board has been powered on
supp_brd_power_off : enabled      #a support board has been powered off
bulk_power_norm : enabled         #bulk p. supply has gone from Failed/OFF to ON
bulk_power_fail : enabled         #bulk p. supply has gone from ON to Failed/OFF
fan_norm : enabled                #fan has gone from Failed to ON or OFF
fan_fail : enabled                #fan has gone from OFF/ON to Failed
```

Event Types for Other Events

`signature_change`	When a processor's signature block changes state.
`system_config_change`	This event describes a situation where any system board, centerplane support board, control board, fan tray, and/or bulk power supply is removed or inserted into the system.

`sys_brd_power_on`	When the power to a system board has switched from off to on.
`sys_brd_power_off`	When the power to a system board has switched from on to off.
`supp_brd_power_on`	When the power to a centerplane support board has switched from off to on.
`supp_brd_power_off`	When the power to a centerplane support board has switched from on to off.
`bulk_power_norm`	When a bulk power supply has gone from a off or failed state to on.
`bulk_power_fail`	When a bulk power supply has gone from on to off or failed state.
`fan_norm:`	When a fan has gone from a failed state to on or off state.
`fan_fail:`	When a fan has gone from a on or off state to failed state.

Predefined Values

Predefined threshold values for over-temperature readings, delta values for temperature readings, and delta values for voltage readings are stored in the SSP's persistent store area; see `ssp_resource`(4).

EXAMPLES

```
#
# Event Monitor Configuration File
#
centerplane_temp_warn : enabled
centerplane_volt_max  : enabled
```

The above example tells the system to enable the two centerplane event types for monitoring.

FILES

The following file is supported:

`$SSPVAR/etc/`*platform_name*`/edd.emc` — Path to an instantiated EMC file

SEE ALSO

`edd`(1M), `ssp_config`(1M), `edd.erc`(4)

NAME

edd.erc – event response configuration file

DESCRIPTION

`edd.erc` files are ASCII files that specify how the system is to respond to certain events. Each domain has its own Event Response Configuration (ERC) file, and one exists for global (system-wide) events. Domain ERC files are responsible for events specific to a system board in that domain. Global ERC files handle events for non-system boards, system boards that are not part of a booted domain, and other non-domain-specific components.

Global ERC files are generated from a template file upon invocation of the `ssp_config`(1M) command. Domain ERC files are generated from a template file by `domain_create`(1M). The template files used by both commands reside in `$SSPVAR/.ssp_private/templates/Ultra-Enterprise-10000`. The global ERC template is named `edd.platform.erc` and the domain ERC template is named `edd.domain.erc`.

Each ERC file contains a series of lines in the following format:

event_type : invoke_action : throttle_timeout : throttle_counter : select_action

where:

event_type

A mnemonic (name string) which corresponds to an event type. See EXAMPLES, below.

invoke_action

A keyword, either `enabled`, which tells the system to invoke the Response Action Script for the event type; or `disabled`, which tells it not to. If this field is blank the system does not invoke the script.

throttle_timeout

A time interval, specified in seconds, that indicates how often *throttle_counter select_action*(s) are able to run. (Additional *throttle_counter select_action*(s) are not permitted to run until *throttle_timeout* seconds have expired.)

throttle_counter

A number that specifies the number of times *select_action* is permitted to run within each *throttle_timeout* interval. After *throttle_timeout* seconds has expired, *throttle_counter select_action*(s) is again permitted to run.

select_action

The Response Action Script to be invoked. This field can contain the script name, with or without a full path, as necessary. The path may contain an environment variable, such as `$SSPOPT, $SSPVAR` or `$SUNW_HOSTNAME`. Optional arguments to the script name can be literals, such as `–v`, or event information: `%e` for event type, `%b` for the board with the event, `%t` for board type, and `%d` for the SNMP trap data. Valid board types (`%t`) are:

```
0 : No type
1 : System board
2 : Control board
3 : Centerplane
4 : Centerplane support board
```

See EXAMPLES, below.

Fields are separated by a single colon with or without spaces to its right and left. Lines that begin with a pound sign (#) are considered comments and are not parsed.

The *throttle_timeout* and *throttle_counter* fields are used to control how often an action should be run. For example, with a *throttle_timeout* value of 600 seconds and a *throttle_counter* value of 3, an action (*select_action*) can run only three times every 10 minutes (600 seconds). This throttling of actions is helpful to reduce the number of repetitive log messages and dump files. Note that similar actions for different system boards, domains, etc. are throttled independently. For example, a `sys_brd_temp_max` event action would not throttle another `sys_brd_temp_max` event action for a different board. Similarly, an arbstop event action would not throttle another arbstop event action for a different domain.

The ERC file can specify more than one Response Action Script for a given event. To designate a secondary Response Action Script, use a second line with the same event-type mnemonic as that of the first line. Response Action Scripts are invoked sequentially (rather than in parallel) in the order they appear in the Event Response Configuration file. If multiple Response Action Scripts exist for an event, you can supply the name and exit status of the previous Response Action Script to the present Response Action Script through the arguments `%p` and `%s`, respectively.

The `edd.platform.erc` and `edd.domain.erc` files together contain the following information. For more information about event types, see `edd.emc`(4). The information in the ERC files is organized as follows:

System Board Events

Control Board Events

Centerplane Events

Centerplane Support Board Events

IDN Events

CBS/CBE Connection Events

System Configuration Change Events

Host Recovery Events

Other Events

System Board Events

```
sys_brd_temp_norm : enabled : 0 : 1 : TempNormact -b %b -e %e -d %d -t %t
sys_brd_temp_high : enabled : 300 : 1 : TempHighact -b %b -e %e -d %d -t %t
sys_brd_temp_warn : enabled : 300 : 1 : TempWarnact -b %b -e %e -d %d -t %t
sys_brd_temp_max : enabled : 300 : 1 : TempMaxact -b %b -e %e -d %d -t %t
sys_brd_temp_911 : enabled : 60 : 1 : Temp911act -b %b -e %e -d %d -t %t
sys_brd_temp_bad : enabled : 300 : 1 : TempBadact -b %b -e %e -d %d -t %t
sys_brd_volt_norm : enabled : 0 : 1 : VoltageNormalact -b %b -e %e -d %d -t %t
sys_brd_volt_max : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
sys_brd_volt_min : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
sys_brd_volt_bad : enabled : 300 : 1 : VoltageBadact -b %b -e %e -d %d -t %t
```

System Board Response Actions

`sys_brd_temp_norm`

Log a message indicating that the board temperature has gone from an over-temperature condition to normal.

`sys_brd_temp_high` **or** `sys_bd_temp_warn`

Log a message indicating that the board's temperature is high.

`sys_brd_temp_max`

Execute the following steps, as necessary, to handle a maximum over-temperature event:

1. If the board is part of an IDN, unlink the domain.
2. If the board is in a domain, shutdown the domain.
3. Power off the board.
4. If the board is in a domain and there are other boards with power in the domain, reboot the domain.

`sys_brd_temp_911`

If the board is part of an IDN, unlink the domain.

Power down the board regardless of whether belongs to a domain.

`sys_brd_temp_bad`

Log a message indicating that the system was unable to obtain the temperature of the board.

`sys_brd_volt_norm`

Log a message indicating that the board's voltage reading has returned to a normal condition.

`sys_brd_volt_max`

Log a message indicating that the board's voltage reading has risen above the maximum threshold.

`sys_brd_volt_min`

Log a message indicating that the board's voltage reading has dipped below the minimum threshold.

`sys_brd_volt_bad`

Log a message indicating that system was unable to obtain the board's voltage reading.

Control Board Events

```
cb_temp_norm : enabled : 0 : 1 : TempNormact -b %b -e %e -d %d -t %t
cb_temp_high : enabled : 300 : 1 : TempHighact -b %b -e %e -d %d -t %t
cb_temp_warn : enabled : 300 : 1 : TempWarnact -b %b -e %e -d %d -t %t
cb_temp_max : enabled : 300 : 1 : TempMaxact -b %b -e %e -d %d -t %t
cb_temp_911 : enabled : 60 : 1 : Temp911act -b %b -e %e -d %d -t %t
cb_temp_bad : enabled : 300 : 1 : TempBadact -b %b -e %e -d %d -t %t
cb_volt_norm : enabled : 0 : 1 : VoltageNormalact -b %b -e %e -d %d -t %t
cb_volt_max : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
cb_volt_min : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
cb_volt_bad : enabled : 300 : 1 : VoltageBadact -b %b -e %e -d %d -t %t
```

Control Board Response Actions

`cb_temp_norm`

Log a message indicating that the board temperature has gone from an over-temperature condition to normal.

`cb_temp_high or temp_warn`

Log a message indicating that the board's temperature is high.

`cb_temp_max`

If the system has fewer than two control boards configured, shutdown all domains in the system and power everything off. If the system has two control boards, power off that the control board that is reading maximum temperature.

`cb_temp_911`

Shutdown the entire system.

`cb_temp_bad`

Log a message indicating that system was unable to obtain the temperature of the board.

`cb_volt_norm`

Log a message indicating that the board's voltage reading has returned to a normal condition.

`cb_volt_max`

Log a message indicating that the board's voltage reading has risen above the maximum threshold.

`cb_volt_min`

Log a message indicating that the board's voltage reading has dipped below the minimum threshold.

`cb_volt_bad`

Log a message indicating that system was unable to obtain the board's voltage reading.

Centerplane Events

```
centerplane_temp_norm : enabled : 0 : 1 : TempNormact -b %b -e %e -d %d -t %t
centerplane_temp_high : enabled : 300 : 1 : TempHighact -b %b -e %e -d %d -t %t
centerplane_temp_warn : enabled : 300 : 1 : TempWarnact -b %b -e %e -d %d -t %t
centerplane_temp_max : enabled : 300 : 1 : TempMaxact -b %b -e %e -d %d -t %t
centerplane_temp_911 : enabled : 60 : 1 : Temp911act -b %b -e %e -d %d -t %t
centerplane_temp_bad : enabled : 300 : 1 : TempBadact -b %b -e %e -d %d -t %t
centerplane_volt_norm : enabled : 0 : 1 : VoltageNormalact -b %b -e %e -d %d - t %t
centerplane_volt_max : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
centerplane_volt_min : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
centerplane_volt_bad : enabled : 300 : 1 : VoltageBadact -b %b -e %e -d %d -t %t
```

Centerplane Response Actions

`centerplane_temp_norm`

Log a message indicating that the board temperature has gone from an over-temperature condition to normal.

`centerplane_temp_high` **or** `centerplane_temp_warn`

Log a message indicating that the board's temperature is high.

`centerplane_temp_max`

Shutdown all remaining domains, then power off the system.

`centerplane_temp_911`

Shutdown down the system.

`centerplane_temp_bad`

Log a message indicating that system was unable to obtain the temperature of the centerplane.

`centerplane_volt_norm`

Log a message indicating that the centerplane's voltage reading has returned to a normal condition.

`centerplane_volt_max`

Log a message indicating that the centerplane's voltage reading has risen above the maximum threshold.

`centerplane_volt_min`

Log a message indicating that the centerplane's voltage reading has dipped below the minimum threshold.

`centerplane_volt_bad`

Log a message indicating that system was unable to obtain the centerplane voltage reading.

Centerplane Support Board Events

```
supp_brd_temp_norm : enabled : 0 : 1 : TempNormact -b %b -e %e -d %d -t %t
supp_brd_temp_high : enabled : 300 : 1 : TempHighact -b %b -e %e -d %d -t %t
supp_brd_temp_warn : enabled : 300 : 1 : TempWarnact -b %b -e %e -d %d -t %t
supp_brd_temp_max : enabled : 300 : 1 : TempMaxact -b %b -e %e -d %d -t %t
supp_brd_temp_911 : enabled : 60 : 1 : Temp911act -b %b -e %e -d %d -t %t
supp_brd_temp_bad : enabled : 300 : 1 : TempBadact -b %b -e %e -d %d -t %t
```

```
supp_brd_volt_norm : enabled : 0 : 1 : VoltageNormalact -b %b -e %e -d %d -t %t
supp_brd_volt_max : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
supp_brd_volt_min : enabled : 300 : 1 : Voltageact -b %b -e %e -d %d -t %t
supp_brd_volt_bad : enabled : 300 : 1 : VoltageBadact -b %b -e %e -d %d -t %t
```

Centerplane Support Board Response Actions

`supp_brd_temp_norm`

Log a message indicating that the board temperature has gone from an over-temperature condition to normal.

`supp_brd_temp_high` **or** `supp_brd_temp_warn`

Log a message indicating that the board's temperature is high.

`supp_brd_temp_max`

Shutdown all running domains, then power off the system.

`supp_brd_temp_911`

Shutdown down the system.

`supp_brd_temp_bad`

Log a message indicating that system was unable to obtain the temperature of the centerplane support board.

`supp_brd_volt_norm`

Log a message indicating that the centerplane support board's voltage reading has returned to a normal condition.

`supp_brd_volt_max`

Log a message indicating that the centerplane support board's voltage reading has risen above the maximum threshold.

`supp_brd_volt_min`

Log a message indicating that the centerplane support board's voltage reading has dipped below the minimum threshold.

`supp_brd_volt_bad`

Log a message indicating that system was unable to obtain the centerplane support board's voltage reading.

IDN Events

```
idn_boot : enabled : 20 : 1 : IDNevent -e %e -d %d
idn_halt : enabled : 20 : 1 : IDNevent -e %e -d %d
idn_awol : enabled : 30 : 1 : IDNevent -e %e -d %d
cluster_arbstop : enabled : 1800 : 3 : Arbstopact -d %d
cluster_recordstop : enabled : 1800 : 3 : Recordstopact -d %d
```

IDN Event Response Actions

`idn_boot`

If at least one other domain in the same IDN has also booted and loaded the IDN software, execute the `domain_link`(1M) command to link the subject domain into the IDN.

`idn_halt`

Update internal IDN state information to enable the IDN event-handling routines to maintain accurate status of all IDN member domains.

`idn_awol`

The local domain reports AWOL domains present. If the status of the AWOL domains are down, unlink them from the respective domain so that the remaining IDN member domains can commence communication again.

`cluster_arbstop`

Do an `hpost`(1M) dump of all IDN member domains and save the bbram information for the boot processors, then do a complete bringup of all IDN member domains.

`cluster_recordstop`

Do an `hpost`(1M) dump of all IDN member domains, then attempt to clear the record stop of all IDN member domains.

CBS/CBE Connection Events

```
cbe_connected : enabled : 0 : 1 : actionsysclock
cbe_connected : enabled : 0 : 1 : actioncb
cbe_connected : enabled : 0 : 1 : PowerFailRebootact
```

CBE/CBS Connection Response Actions

Set the system clock (if necessary) and fan speed, then re-establish control board heartbeat. All previously booted domains are checked for operating system and power status to determine if a domain must be rebooted due to a

power failure condition. This event represents the condition where the SSP's CBS daemon and the Control Board Executive (CBE) lose connection.

Host Recovery Events

```
arbstop : enabled : 900 : 3 : Arbstopact -d %d
recordstop : enabled : 900 : 3 : Recordstopact -d %d
watchdog : enabled : 900 : 3 : WatchDogRebootact -d %d
environment_shutdown : enabled : 600 : 1 : Environmentact -d %d
obp_reset : enabled : 300 : 3 : ObpResetact -d %d
reboot : enabled : 300 : 3 : Rebootact -d %d
panic1 : enabled : 900 : 3 : Panicact -t 300 -d %d -e %e
panic2 : enabled : 900 : 3 : Panicact -t 900 -d %d -e %e
panic_reboot : enabled : 900 : 3 : PanicRebootact -d %d
heartbeat_failure : enabled : 900 : 3 : HeartBeatFailact -d %d
```

Host Recovery Response Actions

`arbstop`

Do an `hpost`(1M) dump and do a complete bringup on the domain.

`recordstop`

Do an `hpost`(1M) dump and attempt to clear the record stop.

`watchdog`

Execute the following steps:

1. Dump `resetinfo` of all processors in the domain.
2. Dump the signature block of all the processors in the domain.
3. Do an `hpost`(1M) dump on the domain.
4. Reboot the domain by doing a complete bringup.

`environmental_shutdown`

Log a message indicating that the system detected an environmental shutdown on a specific domain.

`obp_reset`

Log a message indicating that an OBP reset has occurred.

`reboot`

Carry out the user requested reboot by doing a quick bringup.

`panic1`

Sleep for the time specified by the –t option of the Panicact field, then do a quick bringup if the domain is still in a panic1 state.

panic2

Sleep for the time specified by the –t option of the Panicact field, then do a quick bringup if the domain is still in a panic2 state.

panic_reboot

Reboot the system by doing a complete bringup.

heartbeat_failure

Reboot the system by doing a complete bringup.

Other Events

```
sys_brd_power_on : enabled : 0 : 1 : PowerOnact -t %t -b %b
sys_brd_power_off : enabled : 0 : 1 : PowerOffact -t %t -b %b
supp_brd_power_on : enabled : 0 : 1 : PowerOnact  -t %t -b %b
supp_brd_power_off : enabled : 0 : 1 : PowerOffact -t %t -b %b
bulk_power_norm : enabled  : 0 : 1 : BulkPowerNormact -d %d
bulk_power_fail : disabled : 300 : 1 : BulkPowerFailact -d %d
fan_norm : enabled : 0 : 1 : FanNormact -d %d
fan_fail : enabled : 180 : 1 : FanFailact -d %d
system_config_change : enabled : 0 : 1 : SystemConfChangeact -d %d
```

Other Event Response Actions

sys_brd_power_on **and** sys_bd_power_off

Log a message indicating that the system board has been powered on or off.

supp_brd_power_on **and** supp_brd_power_off

Log a message indicating that the centerplane support board has been powered on or off.

bulk_power_norm

Log a message that the 48-volt power supply is on. (Note that 48-volt power is shown as bulk power in some system messages.)

bulk_power_fail

Log a message indicating which 48-volt power supply has failed or is off, then determine if the system can continue operating with the current number of valid power supplies. If not, power down the entire system.

`fan_norm`

Log a message that a fan has gone from a failed state to an on or off state.

`fan_fail`

Log a message that a fan has gone from an on or off state to a failed state.

`system_config_change`

Log a message that a system board, centerplane support board, control board, fan tray, and/or 48–volt power supply has been removed or inserted into the system.

EXAMPLES

EXAMPLE 1 A Global ERC File

```
# Event Response Configuration File
#
centerplane_temp_warn : enabled : 300 : 1 : TempWarnact -b %b -e %e -d %d -t %t
centerplane_temp_warn : enabled : 300 : 1 : fans -hi
centerplane_temp_norm : enabled : 0 : 1 : fans -off
```

The first two lines in the above example of a global ERC file tell the system how it is to respond to an over-temperature event on the centerplane. The first line tells the system to pass the specified information – board number of the board experiencing the event (`%b`), event type (`%e`), SNMP trap data (`%d`), and board type (`%t`) – to an action script named TempWarnact, and to then execute TempWarnact. The second line tells it to turn on the fans at their high-speed setting.

The third line above tells the system to turn off the fans when it sees that the temperature of the centerplane is normal.

FILES

The following files are supported:

`$SSPVAR/etc/`*`platform_name`*`/edd.erc` — Path to an instantiated global ERC file

`$SSPVAR/etc/`*`platform_name/domain_name`*`/edd.erc` — Path to an instantiated ERC file for domain-specific events

SEE ALSO

`edd`(1M), `edd.emc`(4)

NAME

fad_files – file access daemon files

CAUTION

Do not edit this file manually. It is automatically maintained by domain-management tools and commands.

DESCRIPTION

`fad`(1M) consults the `fad_files` file to determine which file access daemon files it should monitor, where those files are located, and what permissions they are assigned. `fad.1M` also uses the information in `fad_files` to synchronize SSP file reads, writes, and notifications of modification. And other SSP daemons and clients across the net also access this file for file access daemon information.

The `fad_files` file is initially installed in `$SSPVAR/.ssp_private`. It normally contains one or more lines of the following format:

type permission path sub_path filename

where:

type	File type, either **`domain`** or **`nondomain`**.
permission	Access privilege, **`r`** (read-only), **`w`** (write only) or **`rw`** (read and write). `fad`(1M) can read or write to a file only if that file has the proper permission.
path	Valid Enterprise 10000 path. *path* can be any of the SSP environment variables such as SSPOPT, SSPVAR and SSPETC.
sub_path	Absolute path under the Enterprise 10000 path. *sub_path* is normally **`etc`**, **`data`** or **`.`** (dot).
filename	The actual `fad`(1M) file name.

Any line that has a pound sign (#) in the first character position is considered a comment line.

EXAMPLES

EXAMPLE 1 A Sample `fad`(4) File

```
#
#     SSP-Enterprise 10000: fad_files example
#
nondomain   rw  SSPVAR   .ssp_private  domain_config
domain      rw  SSPVAR   etc           eeprom.image
nondomain   r   SSPETC   etc           ssp_startup.main
```

NOTES

This file is shipped with the SSP package and is installed with it.

SEE ALSO

`fad`(1M)

NAME

postrc – (.postrc) hpost properties file

CAUTION

Many of the `.postrc` directives are appropriate only in an engineering or manufacturing environment. Others may be appropriate for field use, but only by a qualified service provider or properly trained system administrator. **Users without specific training in the use of these features should not use them.**

DESCRIPTION

The `.postrc` file is an ASCII text file that contains the various configurable properties of the `hpost`(1M) program; it controls options in POST.

Some `hpost`(1M) functions can also be controlled from the command line. Such functions are clearly marked in the descriptions that follow. Command line arguments take precedence over commands in the `.postrc` file, which take precedence over built-in defaults. For a terse reminder of the `.postrc` options and syntax, enter **`hpost --?postrc`**.

Unless –`n` is the first argument on the `hpost` line, `hpost` reads the optional file `.postrc`, and executes the directives in that file before it begins operation with the host. If `hpost` does not find `.postrc` it proceeds without it. `hpost` first looks for `.postrc` in the current directory (.). If it does not find it there, `hpost` looks in `$SSPVAR/etc/`*platform_name*`/$SUNW_HOSTNAME`. If it does not find `.postrc` there, it looks in the user's home directory, `$HOME`. Exception: If the current directory is `$HOME`, the first element of the search path (.) is skipped. This exception allows a domain-specific `.postrc` file to take precedence over the user's default `$HOME/.postrc` when running from the user's home directory, while still allowing use of a local `./.postrc` to take precedence as long as the user is currently in any directory other than the user's home directory.

Some of the commands described below are accepted only when the local file `./.postrc` is used; such commands are clearly marked. These commands are never available when the current directory is `$HOME`, as the `./.postrc` search path is skipped in such cases.

The following rules apply to the `.postrc` file:

- Keywords are case-sensitive.
- Numeric arguments are decimal by default, taken as hex if preceded by `0x` or `x`.
- Keywords and optional arguments are delimited by whitespace.
- Any part of a line that begins with a pound sign (#) is considered a comment and ignored.

PROPERTIES

The following list of properties is provided in alphabetical order. Each of these commands can also be classified by the appropriate user level, as designated by the number in parentheses at the end of the title line. The levels are:

Level 1	System Administrator Commands. These commands control functionality either intended for system administrator control or safe because their effect is essentially benign (for example, `logfile`).
Level 2	Commands for use by Sun Microsystems and service providers only.
Level 3	Commands for use by Sun Microsystems POST developers (for debug) and Sun Microsystems engineers only.

`allow_lab-only_components` **(Level 3)**

SMI engineering labs sometimes need to operate systems containing hardware that is not acceptable for customer machines. These components are often early prototype parts with known bugs or speed problems, for example. To be sure such parts cannot inadvertently find their way into a production system, POST normally rejects and fails these parts immediately, based on their Component ID (CID) values. To support the engineering lab requirement, the `allow_lab-only_components` command instructs POST not to perform this CID-based rejection, and to accept certain specific values which are recognized but identified as lab-only. This command does not disable all CID checks, it just expands the list of acceptable CIDs.

`allow_sys_errctl` **(Level 3)**

When JTAG-configuring the PC, CIC, or MC ASICs, always enable system register write/clear access to error status, error mask, and force error registers. This property is normally enabled only when diagnostic code (such as POST) is run, and not in the final configuration.

`alt_level` *level* **(Level 1)**

Use *level* as the POST level, rather than the default alternate level, if the –a command argument is invoked. However, if the `level` command is also invoked, and specifies a higher value, that higher value is used.

This –a mechanism is intended for certain error reboot scenarios, to run higher-than-normal levels of diagnostics. However, regardless of the level

specified by `alt_level`, –a will not result in `hpost` running at a lower level than if –a were not invoked.

`arb_flip_ctr_limit` *value* **(Level 3)**

All of the Enterprise 10000 arbitration ASICs reverse the priority order of their clients' requests after some number of grants to improve the fairness of the arbitration algorithm. The number of grants is programmable when the ASIC is configured. A default value is normally used. This command allows Sun Microsystems Engineering to experiment with this value.

`asic_ignore` *asic_type* **(Level 3)**

Do not test, configure or dump the named ASIC type. This command allows Sun Microsystems Engineering to exercise POST code in development systems in which certain ASICs do not exist, which would otherwise cause POST to fail immediately. To get a list of the available *asic_type* keywords, use `hpost -?postrc`.

`assume_ldarb_rev_2` **(Level 3)**

Do not check the CID of LDARB ASICs to understand how to set their configuration; just assume they are not rev 1. This command supports a special mode of POST (see `scantool_simulate`) used to provide the JTAG configuration as input to Verilog ASIC simulation, in which no ASIC input is possible.

`blacklist_file` {`file | none`} **(Level 2)**

Command Line Equivalent: –X Use the specified `blacklist`(4) file rather than the default, `$SSPVAR/etc/`*platform_name*`/blacklist`. `file` can be the path to the `blacklist`(4) file or `none`. If it is `none`, POST reads no blacklist file.

Be careful when using `blacklist_file` on a production system; other SSP software does not know when POST is using a nonstandard `blacklist`(4) file.

`board_red_any_fail` **(Level 2)**

If any failure is detected on a board, immediately fail and effectively redlist all the resources on the board. This line is intended to prevent further configuration of the board, so that a postmortem analysis of the stop can be performed. See also `system_red_any_fail` and `redlist`(4).

`bus_shuffle_mode` {`0|1|2`} **(Level 3)**

A facility in the PC ASIC allows the assignment of addresses to GABs to be varied from the default, which has it based on PA[8:6], with the exact assignment depending on how many and which GABs are configured. This default is shuffle mode 0. Shuffle mode 1 uses PA[8:6] XOR PA[18:16], shuffle mode 2 uses PA[8:6] XOR PA[20:18]. This command allows Sun Microsystems Engineering to experiment with the effect of this shuffle capability.

`cplane_initial_config` **(Level 3)**

Command Line Equivalent: `–C` Start POST in the initial (rather than a subsequent) domain. Several independent domains may be running on an Enterprise 10000 system, but unless they are in `isolated_sysboard` mode, they all share the centerplane ASICs, which can be in only one functional bus configuration. By default, POST assumes it is not running in the first domain being brought up. So it probes the centerplane to determine the bus configuration, and constrains itself to use that configuration. Also, it does not do the functional configuration of the centerplane ASICs.

The usual way to start POST running in the initial domain is via the `–C` command line option. When POST is running in the initial domain, it must establish the bus configuration *and* do the initial centerplane ASIC configuration.

This command (and the `–C` command line option) starts POST in the initial domain, where it must establish the bus configuration *and* do the initial centerplane ASIC configuration.

`cplane_initial_config` is equivalent to the `–C` command line option for the convenience of Sun Microsystems Engineering, Manufacturing and Customer Service technicians who must repeatedly run POST on uninitialized systems.

Do *not* use this command on production systems that run, or may someday run, multiple domains, as it may cause POST running in the domain brought up second to crash anything running in the domain brought up first.

`debug_mask` *mask* **(Level 2/3)**

Use *mask* as the debug mask. This is a bitmask that controls experimental code or obscure behavior modes of POST. It is mostly for POST developer use. The *mask* is ORed into the internal mask, allowing one line/bit, so bits can be added/removed by the commenting out of a list in the `.postrc` file. Because these bits change over time users should execute `hpost --?postrc` for a list of current bits.

`default_ecache_mbytes` **{**`H|1|2|4|8|16|32`**} (Level 2)**

Assume the specified ecache size for all processors if the ecache probe, done as part of the initial processor test phase, is skipped. If the probe is skipped and this value is incorrect, the configured system will not maintain correct cache coherency. The "default default" is 0.5MB – "H".

`default_memgrp_mbytes` **{**`64|256|1024`**} (Level 2)**

Use the specified dimm group size as the default. The default size is used for FOM calculations prior to the dimm probe phase, and also as the assumed value of all good dimm groups if the dimm probe phase is skipped by the `skip_phase` command. The "default default" is 64MB.

`display_fom_calc` **{**`off|on`**} (Level 1)**

If `on`, always make POST print the calculated Figures Of Merit (FOM) for all 45 bus configurations when it is making an FOM calculation to select a configuration. If `off`, never do so. If this command is not present, POST does the printing if the verbose level is above a certain value.

`dom_transgress_err_enbl` **(Level 2)**

Enable reporting of Domain Transgression Errors by the GDARB ASIC when configuring the centerplane, which is done when `hpost` is called with the `–C` option. (See `cplane_initial_config`). A transgression error occurs when a system board's data arbiter requests to send data to a board in a different domain cluster as defined by the GDARB's SMD mask configuration register.

By default, reporting of these errors is disabled to minimize the probability of GDARB asserting a global arbstop when a board fails and causes both a parity error and a transgression error. GDARB does inhibit transgressing requests; it just doesn't report them as errors. This `dom_transgress_err_enbl` command enables this error reporting, for cases where the additional fault detection is preferred over the additional domain fault isolation.

`domain` *boardmask* **(Level 3)**

Declare a set of system boards to be a domain. Domains are disjoint partitions of the system boards in an Enterprise 10000 system that are isolated from each other, except for shared memory (see `command shared_mem`).

This `.postrc` command is not how domains are managed in production systems; it is a static instruction to POST to configure the domain registers

for Sun Microsystems Engineering debug of the domain hardware. This `domain` command is effective only when used in a `.postrc` file in the current directory. If the `.postrc` file used is not in the local directory, POST ignores this command and generates a warning message.

`download_bbsram_verify` {`off`|`on`} **(Level 2)**

If `on`, always verify that each JTAG POST code download to a processor BBSRAM actually occurred. If `off`, never do so. If it is not specified, verification is done only if the level is above a certain value. To do the verification, `download_bbsram_verify` uploads code from BBSRAM and compares it to the code file data. Unlike download, which is done with a JTAG broadcast, the upload and compare must be done serially for each processor.

`download_path` *dirpath* **(Level 2)**

Use the download files for POST that are in the location *dirpath*. The default is `$SSPOPT/release/Ultra-Enterprise-10000/*/*/*/hostobjs`, where `/*/*/*/` indicates a release-level dependency.

`dtag_test_all_procs` **(Level 2)**

Normally, POST assigns each processor to test its own DTAG RAM for each CIC on the board. This process ensures that only resources that can be used are tested, as DTAG RAM for missing or failed processors is not usable for anything else. However, in some environments, such as manufacturing test, Sun Microsystems prefers that all the DTAG RAM be tested, in preparation for future installation of additional processors. This command causes each processor, during the DTAG RAM test phase, to test `all` the RAM on each CIC, not just its own. For the same reasons, this command also causes testing of all physically present DTAG SRAM locations, not just those that will be used to support the size of ecache on the currently installed processor modules.

`dump_no_sneeprom` **(Level 3)**

When creating a dump file, either with the `–D` command line option or automatically when certain errors are encountered while POST is running, do not include the contents of the serial number eeproms in the dump. This command can significantly speed up the dump process in some environments. It may be removed in the future.

`dump_on_exit` **[***boardmask* **[***path***]] (Level 2)**

Make POST create a dump file of the system state just before it exits. The *boardmask* controls what boards are dumped, where bits [15:0] control system boards and [17:16] control half-centerplanes. The board is dumped if the corresponding bit is 1. The default is x3FFFF (dump all). This would normally be specified as a hex value. *path* can be used to override the default path for the dumpfile. A *path* can be specified only if a *boardmask* is specified. Both options are similar to those of the –D command line option.

`dump_pathname` *dirpath* **(Level 2)**

Create state dump files in the specified default directory rather than the program default, `$SSPVAR/adm/$SUNW_HOSTNAME`. If a path has been specified by either the command line option (–D) or the `dump_on_exit` command, that path overrides the one specified by `dump_pathname`.

`fom_weight_abus` *float-in_range* **(Level 1)**

Use the specified weight of the number of address buses in the Figure of Merit computation. The float-in range is 1.0 to 5.0; the default is 1.0.

`fom_weight_dbus` *float-in_range* **(Level 1)**

Use the specified weight of the number of data buses (72-bit datapaths) in the Figure of Merit computation. The float-in range is 1.0 to 5.0, inclusive; the default is 1.0.

`fom_weight_mem` *float-in_range* **(Level 1)**

Use the specified weight of memory in the Figure of Merit computation. The float-in range is 1.0 to 5.0, inclusive; the default is 1.0.

`fom_weight_proc` *float-in_range* **(Level 1)**

Use the specified weight of processors in the Figure of Merit computation. The float-in range is 1.0 to 5.0, inclusive; the default is 1.0.

`fom_weight_scard` *float-in_range* **(Level 1)**

Use the specified weight of SBus cards in the Figure of Merit computation. The float-in range is 1.0 to 5.0, inclusive; the default is 1.0.

`force_enbl_cp_driver` **(Level 3)**

Enterprise 10000 system ASICs that connect to the centerplane have both a loopback control and a separately controlled enable for the centerplane drivers. Normally, POST configures these ASICs so that the drivers are disabled when the chip is in loopback, and enabled when it is not. For probing the board signals, it is sometimes useful to drive the centerplane connector with the address and data buses even when in loopback. This command causes all ASICs to have their centerplane drivers enabled, even when configured in board loopback. See also `pc_loopback_lbus_enbl`.

`gamux_seq_err_enbl` **(Level 2)**

Early GAARB revisions, which should never be shipped in production machines, have a design bug that causes sequence errors that will be detected by the GAMUX ASICs when an arbstop occurs for some other reason in one domain. Since these sequence errors are considered global errors, this bug results in an arbstop in all other domains, as well.

By default, detection of these sequence errors is disabled when the GAARBs are rev 1 or 2, and enabled otherwise. The `gamux_seq_err_enbl` command allows explicit control, disabling detection of these errors when set to `0`, enabling it when set to `1` – regardless of GAARB revision.

`interactive_use_postrc_skips` **(Level 2)**

When running in interactive mode (`–i` command line option), the default behavior is to ignore any `skip_phase` or `skip_test` commands in the `.postrc` file, giving the user the option of running or skipping each interactively. In some cases that may not be the desired behavior; the user may not want to have to interactively skip dozens of phases and tests to get to the one of interest. This command causes POST to use any `skip_phase` or `skip_test` commands, and not query the user about whether to run them. See also the `no_skip_phase_covers_npb` command; the implied skips described there are considered to be in the `.postrc` file with respect to this command.

`interconnect_MHz` *float* **(Level 2)**

Use the specified system interconnect frequency, expressed in MegaHertz. This value is used for calculating system configuration parameters (most particularly, memory timing) in teststand environments lacking a real control board that can measure and report this frequency through the SNMP server.

In systems that do have a control board, where use of the SNMP frequency measurement is used, specifying this value causes POST to require that the measured value be within 0.5 percent of this value. If it is not specified, the

measured value must be within -3.0 and +0.5 percent of the target interconnect frequency read from the SNMP server.

See also `proc_freq_ratio`.

`iopc_rev_5_ok` **(Level 2)**

The rev-5 PC asic has a bug that occurs when the asic is used in the I/O position, PC number 2 on any system board. The effects of the bug may be acceptable for certain applications, with or without certain software workarounds installed. POST normally fails these asics, but accepts them if their presence is explicitly acknowledged with this command.

`jtag_lock_trace` **(Level 2)**

Make POST print verbose debugging messages about its actions with respect to the locking and unlocking of JTAG boards and rings.

`isolated_sysboard` **(Level 2)**

Make POST ignore the fact that the centerplane ASICs appear missing or broken, which would normally cause it to fail immediately; and prevent POST from taking the board out of loopback. This function is used in development environments that do not have a centerplane.

`jtag_lock_timeout_secs` *value* **(Level 2)**

Change the default timeout value for obtaining a lock on a JTAG resource from the SSP core services.

`level` *level* **(Level 1)**

Command Line Equivalent: `–l` Use the specified level to determine the amount of testing to be done and the thoroughness with which it is done. Higher levels result in more testing. Valid levels are in the range 7 to 127, inclusive. `hpost -?level` prints a brief summary.

This command is suitable for end user control, but do not set the level lower than normal. Doing so subverts the intent to boot only systems with a low likelihood of undetected hardware problems. Setting it higher than the default (see the `–l` option of `hpost`(1M) may be appropriate for customers willing to trade longer boot times for more thorough diagnostic coverage.

`logfile` **[***path***] (Level 1)**

Command Line Equivalent: –g Create a screen logfile. The default filename is `post`*mmdd.hhmm*`.log` and the default directory is `$SSPVAR/adm/$SUNW_HOSTNAME/post`. If the default path is used and the default directory does not exist, but `$SSPVAR/adm/$SUNW_HOSTNAME` does, POST creates the `post` subdirectory and places the log there.

If *path* is specified and it can be opened for append, POST creates the log there. Otherwise, if the default filename can be opened in *path* used as a directory, POST creates the log there. If a log is requested and none of the above is specified or successful, POST attempts to create the logfile in the current directory with the default name.

The logfile is always opened for append, and is timestamped both when opened and closed, so that a calling script or program can specify a single file to receive logs from a series of POST runs, and to prevent loss of information in the unlikely event of a name conflict.

This command is rated Level 1 because it is fairly benign, other than using up SSP disk space. However, note that in a production environment, POST output is usually diverted (by the `bringup`(1M) script using –`s`) to `syslog`, which may be a more appropriate way to save this information in production environments.

`logger_level{default|emerg|alert|crit|err|warn|notice|info|debug}` **(Level 2)**

POST captures all liblogger messages produced by the libraries it uses, so they can be part of its own displays and logs. Normally, it also overrides the default logger level to WARNING, so that the contents of its displays are not subject to external control. This command causes POST to either not override the current default logger level it inherits from the environment, allowing it to be controlled by the normal logger mechanisms, or to explicitly set the logger level to any desired value.

If the `.postrc` file used is not in the local directory, POST ignores this command and generates a warning message.

`logical_memboard_swap` *slot1 slot2* **(Level 3)**

Exchange the *logical memory board numbers* of the two named physical system boards. Physical memory addresses are assigned by POST starting at 8GB * *logical board number*. By default, the assignment of logical memory board to physical system board is 1-1. Swaps may occur over time, for example as the result of Dynamic Reconfiguration board detach operations, or the sort done by POST as explained in the description of the command `no_memboard_sort`. In a production system SSP, this physical-to-logical

mapping is kept in the SNMP MIB and read by POST each time it runs. This command allows Sun Microsystems Engineering to use non-default mappings in lab environments where SNMP is not running.

Any number of these commands may be present; the swaps occur in the order requested. The only constraints are that the two arguments are different valid system board numbers.

This command is effective only when used in a `.postrc` file in the current directory. If the `.postrc` file used is not in the local directory, this command is ignored and a warning message is generated.

Any swapping caused by this command is overwritten by the map information obtained from SNMP, and effectively ignored, unless that access is inhibited with the command `no_snmp_memmap` or `no_snmp_access`. Such swaps cannot be done in combination with the map obtained from SNMP; the two are mutually exclusive.

`mc_gabfifo_hold_depth` *6-bit value* **(Level 3)**

Override normal MC ASIC configuration of this field, which is the constants register bits [17:12].

`mc_prescale_disbl` **(Level 2)**

Override normal MC ASIC configuration which causes all timeouts to be prescaled by 2**10 system clocks. This field is the Timeout Select register bit 23.

`mc_store_match_to` *5-bit_value* **(Level 2)**

Override normal MC ASIC configuration of this field. This field is the Timeout Select register bits [9:5].

`mc_timing_reg` *reg value* **(Level 3)**

Override normal MC ASIC configuration of the specified memory timing control register [3:0], where register 0 is refresh timing, register 1 and 2 are memory timing 1 and 2, and register 3 is DIMM wire timing.

`mc_xmit_req_to` *5-bit_value* **(Level 2)**

Override normal MC ASIC configuration of this field, which is the Timeout Select register bits [4:0].

`mc_xtra_data_to` *5-bit_value* **(Level 2)**

Override normal MC ASIC configuration of this field, which is the Timeout Select register bits [14:10].

`mem_board_interleave_ok` **(Level 1)**

Permit POST to configure memory so that two boards with identical amounts of memory are interleaved in 256-byte blocks through a common address range, which is twice what either board would otherwise contain.

Interleaving can improve performance by distributing memory accesses of large blocks of contiguous data between two memory controllers. However, interleaving two boards can have the effect of restricting dynamic reconfiguration (DR); in particular, it may prevent an interleaved board from being DR detached.

The default POST behavior is to not interleave boards. This command enables interleaving of any two boards with the same amount of memory.

`memboard_sort_order {low_to_high | high_to_low}` **(Level 2)**

Specify the preferred ordering of assignments of memory Physical Addresses to system boards within this domain. `low_to_high` places low addresses on low-numbered system boards and high addresses on high-numbered system boards. `high_to_low` places addresses on system boards in the opposite order.

If this `memboard_sort_order` command is not present, the order is determined by examination of the version of the Solaris operating environment specified in the SSP's SNMP MIB for this domain. If the version is earlier than Solaris 7, it uses `low_to_high`. However, the Solaris 7 operating environment is the first to use `high_to_low`. If access to the SNMP MIB is also suppressed with the command `no_snmp_access`, the default is `low_to_high`.

The purpose of this ordering is to place kernel memory on the lowest-numbered boards, to allow complex systems to be optimally configured for DR without regard to operating system version.

See also the commands `no_memboard_sort` and `no_memboard_fullsort`.

`memboard_swap_threshold_mbytes` *mbytes* **(Level 2)**

This command defines the minimum amount of memory that must be on the lowest-numbered physical board with memory in the domain, to avoid swapping its physical addresses with those of the board in the domain with

the most memory. See the command `no_memboard_sort` for an explanation of the sort and how it can be disabled.

The objective of this swap is to have enough memory in the Physical Address chunk used for kernel memory in the domain for the DR "caged kernel" to fit within this single board, increasing the likelihood that any board can be DR detached. However, if the first board has a required minimum amount of memory, the swap is not required even if another board has yet more memory, and Sun Engineering desires to avoid this unnecessary swap.

The default and minimum value of this no-swap threshold is 512MB. This command is provided to allow field increases of this threshold if the caged kernel characteristics change. The maximum is 4096 (4GB), which is all the memory a board can contain with the largest supported memory DIMMs.

`memchunk_page_truncate` *#_of_8KB_pages_per_chunk* **(Level 3)**

Limit the amount of memory declared in the memory chunk list passed to OBP and cleared in the final configuration phase. If the number of pages in a contiguous chunk (usually the physical memory of one board or a pair of interleaved boards) is greater than this value, truncate it to the declared size. This command does not affect the configuration of the memory controller, only how much memory is declared to OBP and cleared. It is intended for controlled debug environments that can deal with a portion of configured physical memory not initialized, to save time, and is most useful in an emulation environment.

`memtest_limit` **[***base***]** `limit` **(Level 2)**

Limit the amount of memory tested in each dimm group by the POST memory tests. If only one argument is supplied, it is used as the limit and the base is left at 0.

Both values represent byte offsets within each group, as if it were configured at PA 0 without any interleaving. In general, this does not correspond to the physical addresses a given group is assigned in a fully configured machine, or those used during the memory tests themselves. In particular, memory testing is always done as if the banks were 4-way interleaved, even if not all 4 banks are present. Each processor tests a range of addresses 4 times the number of bytes it is testing, but it tests only 64 bytes out of each 256 within that range, all of which are in the same dimm group.

These limits are applied to every group tested. Since memory is tested only in 64-byte blocks, both arguments must be mod-64. The maximum for both is 0x40000000, or 1GB, which is the full size of a group of the largest dimms

supported. The limit must be greater than the base. If the base is greater than a physical group's size, that group is (quietly) not tested.

`new_cid_rev` *component_id* **(Level 2)**

During its JTAG integrity test phase POST reads the component IDs of all the ASICs, as well as other chips that have a CID, and compares them against tables of acceptable CIDs for these chips. In general, discrepancies result in POST failing the chip. However, if a CID matches a value in the table except for the revision (the most significant hex digit of an 8-digit CID), and the actual revision is higher than the revision in the table, the chip is accepted and POST issues a warning about the *up-level* chip.

This process adds resilience to hardware upgrades without forcing distribution of a POST patch. It also serves notice that POST may not be aware of all features and requirements of all present hardware, so that if any hardware problems occur, the chip-version issue will be immediately obvious.

While the up-level message is only a warning, it may make some users uncomfortable. The purpose of this `new_cid_rev` command is to effectively add an entry to the CID table to suppress the up-level warning. The `.postrc` file may contain multiple instances of this command. POST consults the list of values they create only when it is about to issue an up-level warning message. Thus, you cannot use `new_cid_rev` commands to declare new base CIDs, or to declare that a down-level CID revision is acceptable. Since the base part of the entered CID must match that of at least one of the base CIDs in the compiled table for some chip for this command to have any effect, it is not necessary to specify in this `new_cid_rev` command which chip type is having its table extended.

Service providers should use this facility to suppress the up-level warnings only when notified by Sun Microsystems Engineering that a specific up-level chip is really equivalent to the older revision compiled into POST.

Normally, *component_id* would be entered as an 8-digit (32-bit) hex value. It is checked only to be sure that it satisfies the requirement of any valid CID; that is, to ensure that its least significant digit is 1.

`no_asic_config_check` **(Level 3)**

Skip the normal readback and compare of ASIC CSRs written by JTAG when configuring the system. This command is useful in certain development environments during system simulation.

`no_bbc_error_check` **(Level 2)**

POST usually checks the BootBus error status register in the PC, at the end of major phases, for BootBus parity or access errors. Errors usually result in POST failing the PC. This command suppresses all checks of this register.

`no_bbsram_clear` **(Level 2)**

POST normally writes all of BBSRAM to prevent parity errors if some code accesses an uninitialized address. It does so on the first code download of a file declared to need a signature block, to avoid interfering with the special BootBus SRAM test file. To save download time, POST fills only those segments of BBSRAM that not loaded with code or data, and it does so only once.

This command suppresses this fill, so that areas of BBSRAM not explicitly loaded by the code are left untouched.

`no_check_chain_length` **(Level 2)**

During its JTAG integrity test phase, POST checks that all chips with component IDs have one that is the standard 32-bits long. This simple but useful test ensures that the chip's JTAG scan connection is reliable. This command makes POST skip the chain length test, but still do the probing operations of the JTAG integrity test phase.

`no_dumpfile_on_stop` **(Level 2)**

Make POST suppress its normal action of creating a dumpfile of "interesting" system state if and when it detects an arbstop or recordstop during testing.

If this command is not present, POST dumps the system state, edits it to skip boards that have no interesting state, then analyzes the dump. If POST finds no boards with interesting state, it does not create a dumpfile and just moves on. (For example, data ecc error recordstops that originate in memory may cause POST to simply move on without creating a dumpfile.) See also `dump_pathname`.

If `no_dumpfile_on_stop` is not in `.postrc` but `no_stop_analyze` is, POST simply creates the dumpfile then moves on, neither editing nor analyzing it.

`no_final_flush_reset` **(Level 2)**

The normal action of POST is to pause and flush all non-redlisted processors' master queues in the PC ASIC, then reset each processor at the end of its final configuration. This leaves a state that facilitates download of

OBP code to BBSRAM without any problems executing stale fetches from the POST spin-wait loop. If `no_final_flush_reset` is invoked, the processors are instead left spin-waiting in the POST final configuration code in BBSRAM, which may be useful for postmortems of final configuration processor state.

`no_jtag_locks` **(Level 3)**

SSP software that uses the JTAG communication facilities provided by the Enterprise 10000 control board is normally expected to *lock* the JTAG ring or rings with which it is currently communicating to prevent multiple SSP applications (e.g., POST and `obp_helper`(1M)) from interfering with each other. The locking is done through semaphore-based services provided by the SSP libraries.

This command causes POST to skip use of these locking services, which is useful in certain engineering development environments.

`no_lockfile` **(Level 3)**

Permit multiple POST processes to run simultaneously on the same Enterprise 10000 host domain. POST normally creates the lockfile `$SSPVAR/adm/$SUNW_HOSTNAME/hpost.lock` to prevent multiple instances of POST, since mutual interference is likely to occur, causing both POST processes to fail.

In a normal SSP environment, POST is usually run in daemon mode. It is not always obvious that this is happening and, in a development environment, it often interferes with user-initiated POST processes.

However, occasionally allowing two POST processes to run is considered less of a problem than POST refusing to run due to a stale lockfile. Therefore, if creation of the lockfile fails, POST attempts to validate that it was created by another active POST process. If this attempt fails, POST enters `–f` mode (described below) and proceeds to run, deleting any lockfile when it exits. This process provides automatic recovery from most cases in which a lockfile is left from an abnormally terminated POST process.

The effect of this `no_lockfile` command is to completely disable the lockfile function, both file creation and removal.

Note that this is significantly different from the `–f` command line argument. With `–f`, POST still tries to create a lockfile, but continues even if that fails; and it attempts to delete the lockfile at exit, even if the create failed. The purpose of `–f` is recovery from a stale lockfile that incorrectly appears to belong to an active POST process.

`no_mc_hardreset` **(Level 2)**

The Memory Controller (MC) ASIC has some functions, most notably memory refresh, that are not affected by normal system reset. These functions are cleared only by a *hard reset*, which can be caused via the JTAG TAP reset pin, which occurs during a power-on reset, or by a special hard reset control bit in a JTAG config register. It is possible, particularly when LBIST is run, to leave the MC in a state that requires a hard reset to make it usable.

POST normally asserts a hard reset to all MC ASICs during the initial reset phase. This command causes the hard reset to be skipped, which might be useful in some test environments. This hard reset is also skipped if POST is run in the special –Z reconfiguration mode.

`no_memboard_fullsort` **(Level 2)**

Suppress the full sorting of memory Physical Addresses to physical system boards. If this `no_memboard_fullsort` command is invoked, the assignment of Physical Addresses to system boards is taken from the SSP SNMP ConfMemMap MIB element without modification, except for the possible swap described for the command `no_memboard_sort`.

If this `no_memboard_fullsort` command is not invoked, a full sort of addresses within this domain is made in the order described under the description of the command `memboard_sort_order`. This sorted order is then subject to the possible swap described for the command `no_memboard_sort`.

`no_memboard_sort` **(Level 2)**

The default action of POST when running in a domain of two or more boards is to do a partial sort of the boards based on the amount of memory on each board. The object of this sort is to configure the domain such that the physical memory addresses used for kernel memory in the domain are on a board with the largest amount of memory, unless the amount on the lowest board is above a threshold amount. This optimizes the capabilities of some features of Dynamic Reconfiguration, as explained in the description of the command `memboard_swap_threshold_mbytes`.

Doing this sort may require a single physical-to-logical mapping swap of a board that would contain kernel memory in the domain with another board in the domain that has more memory. See the command `logical_memboard_swap` for more information about this mapping. Since the physical-to-logical mapping is maintained by the SNMP agent on the

SSP, if a swap is required, POST must do a write operation to SNMP to inform it of the swap.

This command, `no_memboard_sort`, suppresses both the sorting and the SNMP write access. This suppression is implied if `no_snmp_memmap` or `no_snmp_access` is invoked. See also `no_memboard_fullsort`.

`no_memory_ok` **(Level 2)**

Since the operating system cannot boot without memory, the amount of memory is one component of the configuration Figure of Merit (FOM) used to evaluate configurations. No memory results in a 0 FOM. For board bringup, however, Sun Microsystems sometimes wants POST to continue testing even with no memory installed. This command tells POST to pretend there's one *unit* (currently 64 MByte) of memory when calculating the FOM if there is actually none.

Other features in the processor modules and I/O require valid memory to be tested. Therefore, do not run board tests without memory as standard practice, as these other elements would not be tested. However, this capability is useful upon occasion.

`no_non_proc_boards` **(Level 1)**

Do not allow system boards without a processor on board. By default, Non-Processor Boards (NPBs) are permitted, so that their onboard memory and I/O is not lost because of processor failures. In general, such configurations are fully functional.

In certain situations such NPBs may be undesirable. This command causes POST to remove such boards from the configuration, even though they may have usable I/O and/or memory.

`no_obp_handoff` **(Level 2)**

During the final configuration phase do not create the post2obp structures in BootBus SRAM of the boot processor. This command causes an otherwise successful run of POST to exit with POST_EXIT_NOCONFIG instead of a boot processor number (0 to 63, inclusive).

`no_poll_board_arbstop` **(Level 3)**

Suppress POST's normal polling for arbstop and recordstop. POST does the polling by reading the Local Address Arbiter (LAARB) error register via JTAG. When this command is used, stops are detected in other ways, most

often by timeouts. The processors, instead of the true cause of the error, are failed, and recovery capability is limited.

`no_poll_timeouts` **(Level 3)**

Disable all timeouts on the host processor RPC services. This is useful when running with breakpoints installed in the resident code of a host processor during Sun Microsystems Engineering's debug of POST. Compare to `poll_timeout_mult`.

`no_procs_ok` **(Level 2)**

Since the operating system cannot boot without processors, the number of processors is one component of the configuration Figure of Merit (FOM) POST uses to evaluate configurations. No processors results in a 0 FOM. For board bringup, however, Sun Microsystems Engineering sometimes wants POST to continue running even with no processors installed. This command tells POST to pretend that there is one processor in calculating the FOM if, actually, there are none. When no processors are present, POST can do only a very limited number of functions. But it can do a JTAG ASIC configuration, which is useful in certain bringup environments.

`no_scard_probe` **(Level 2)**

Do not probe for missing scards. *scard* is a generic term for an I/O adapter card, either SBus or PCI bus. If this command is invoked, all I/O card slots architecturally defined on the specific I/O module present are considered populated. This knowledge affects the Figure of Merit calculation done by POST, and is reported to OBP, allowing it to probe any slots declared populated by POST.

In the case of SBus cards, this probe-skipping also can be done by using the `skip_test` command for the appropriate probe test. However, there is no `test` for the probe in the case of a PCI I/O module; the probe is done by JTAG from the SSP.

`no_scards_ok` **(Level 2)**

Since the operating system cannot boot without I/O, the number of SBus cards (or, more generally, the number of populated I/O slots) is one component of the configuration Figure of Merit (FOM) used to evaluate configurations. No SBus cards results in a 0 FOM. For board bringup, however, Sun Microsystems Engineering sometimes wants POST to continue testing even with no SBus cards installed. This flag tells POST to pretend that one SBUS card is present when it is calculating the FOM if, actually, there are none.

`no_skip_phase_covers_npb` **(Level 2)**

Where possible, POST tests the system resources in board loopback, using other onboard resources. This technique improves POST's ability to isolate failures and generally allows faster test execution, because more tests can run simultaneously on different system boards.

If a system board isn't fully populated or has experienced onboard failures, POST must use offboard resources. For example, a board with no good processors (a Non-Processor Board, or NPB) must use offboard processors to test its memory and I/O. A board with no good memory (A Non-Memory Board, or NMB) must use offboard memory to run tests of processor and I/O versus memory. A board with a single processor must run crosscall interrupt tests against processors on other boards.

When a base test phase, such as onboard memory tests, has been skipped due to use of the `skip_phase` command, most technicians would assume that all the memory tests would be skipped. That assumption would lead to confusion if the NPB memory tests then take over and test the on-board memory later in POST. Therefore, the default behavior of POST is that when such a base phase has been skipped, all phases that test the same resources offboard are also skipped automatically.

This is usually the desired behavior but, in some cases, it might not be. Forcing all tests to run out of loopback, while slower, can be an excellent stress test of the centerplane interconnect. Also, the technician or engineer may want to skip the base test, but run the NPB test while troubleshooting a particular problem. To support this, the `no_skip_phase_covers_npb` command suppresses the default implied skip of offboard phases when the base phase is skipped.

Note that while the command uses the abbreviation NPB, it actually affects all such offboard test phases, such as those for NMBs and xcall tests for single-processor boards.

Note also that the implied skip mechanism described operates only at the phase level. If all tests in a base phase have been skipped because of the `skip_test` command, the phase is not considered skipped.

`no_snmp_access` **(Level 2)**

Prevent POST from accessing the SSP's SNMP agent to obtain various information normally required for it to run. This command is provided to support various Sun Microsystems lab environments where the SNMP agent may not be running. It is effective only when used in a `.postrc` file in the current directory. The next component in the search path for `.postrc` requires the *platform_name*, which is obtained from the SNMP agent. If the

`.postrc` file used is not in the local directory, POST ignores this command and generates a warning message.

Many of the items of information normally obtained from SNMP can be provided through the `.postrc` file, or through program defaults. See the following commands: `interconnect_MHz`, `proc_freq_ratio`, `domain`, `logical_memboard_swap`, `platform_name`, and `no_snmp_freq_read`.

`no_snmp_invalidate_bootproc` **(Level 2)**

By default, when a successful invocation of POST returns a boot processor number in the range [0,63], POST invalidates the SNMP value of the bootproc for the domain in which it is running before it starts interaction with the host. (Exception: POST does not invalidate the SNMP value when run in the special –`H` Dynamic Reconfiguration mode.) By invalidating this value, POST causes other SSP monitoring software, particularly the Event Detector Daemon (EDD), to stop checking for arbstops and other events in this domain, possibly interfering with this `hpost`(1M) run.

The `no_snmp_invalidate_bootproc` command suppresses this invalidation. `no_snmp_invalidate_bootproc` is implied by `no_snmp_access`.

Note that POST never sets this SNMP element to a valid value, it only invalidates it. Setting it to a valid bootproc value is normally done by `bringup`(1M). Therefore, when POST is run other than by `bringup`(1M), the SNMP bootproc MIB element is invalidated and left that way.

`no_snmp_memmap` **(Level 2)**

Prevent POST from accessing the SSP's SNMP agent to obtain or modify the physical-to-logical memory board mapping. This prevention is implied if `no_snmp_access` is invoked. See the command `logical_memboard_swap` for more information about this mapping.

This command is effective only when used in a `.postrc` file in the current directory. If the `.postrc` file used is not in the local directory, this command is ignored and a warning message is generated.

`no_snmp_freq_read` **(Level 2)**

Prevent POST from accessing the SSP's SNMP agent to obtain the platform interconnect and processor frequency values. This prevention is implied if `no_snmp_access` is invoked. When either `no_snmp_freq_read` or `no_snmp_access` is used, POST takes these values from the `.postrc`

commands `interconnect_MHz` and `proc_freq_ratio`, or from built-in default values.

`no_stop_analyze` **(Level 2)**

POST's normal action when detecting an arbstop or recordstop condition while running most tests is to read and analyze the status in the various ASICs, and to fail components when the fault can be determined. This command suppresses this function.

`pc_color_freq` *code* **(Level 3)**

The PC ASIC has a two-bit configuration parameter that controls the frequency with which PC changes the value of arbitration color, a property of the arbitration fairness algorithm. A default value is normally used. This command allows Sun Microsystems Engineering to experiment with this parameter. This field is bits [19:18] of the configuration register.

`pc_errmask` **{**`0|1 | io0|io1`**}** *bit_number* **(Level 3)**

Allow individual bits in the error mask registers of PC ASICs to be forced to 0 during configuration, disabling the specified error from causing an arbstop. Multiple instances and versions of this command may be present; the bits to be masked are accumulated. If a number (0 or 1) follows this command, it affects only processor PCs; if the io form (io0 or io1) is used, it affects only I/O PCs. The 0 or 1 indicates which error mask register is specified. *bit_number* is an integer in the range 0-31, inclusive.

`pc_grant_to` *6-bit pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field, which is the Timeout/Hold config register bits [10:6].

`pc_grant_to_io` *6-bit_pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field, which is the Timeout/Hold config register bits [10:6] for I/O PCs. This command overrides any `pc_grant_to` command.

`pc_loopback_always` **(Level 3)**

Force PC loopback mode every time any PC is configured.

`pc_loopback_lbus_enbl` **(Level 3)**

During the initial processor module tests, POST uses a facility of the PC ASIC called PC loopback mode. In this mode the PC does not send transactions through the CICs, but instead loops them back internally. This process allows better fault isolation, as you know that any failures in these tests are not related to the CICs or the PC-CIC signals.

As with board loopback, the PC has separate controls for PC loopback and driver enables for the local bus to the CIC. POST normally configures this so the drivers are disabled in PC loopback mode. This command causes these drivers to be enabled in PC loopback mode, so that they may be probed during debug. Compare this command to `force_enbl_cp_driver`.

`pc_master_pio_req_limit` *2-bit_code* **(Level 3)**

Override normal PC ASIC configuration of this field for processor PCs. (It is always 0 for I/O PCs.) This is the Timeout/Hold config register bits [24:23].

`pc_master_read_to` *6-bit_pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field, which is the Timeout/Hold config register bits [5:0]. Bit [5], the most significant bit of this field, is the prescale enable for all of the master read, interrupt, and slave response timeouts in the PC.

`pc_master_read_to_io` *6-bit_pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field for I/O PCs. This field is the Timeout/Hold config register bits [5:0]. `pc_master_read_to_io` overrides any `pc_master_read_to` command.

`pc_pc_hold` *5-bit_value* **(Level 3)**

Override normal PC ASIC configuration of this field. This field is the Timeout/Hold config register bits [22:18].

`pc_slave_response_to` *5-bit value* **(Level 2)**

Override normal PC ASIC configuration of this field, using the specified 25-bit value for the Timeout/Hold config register bits [31:27]. Note that this timer uses the prescale enable for the master read timeout; see `pc_master_read_to`.

`pc_slave_response_to_io` *5-bit value* **(Level 2)**

Override normal PC ASIC configuration of this field for I/O PCs, using the specified 5-bit value for the Timeout/Hold config register bits [31:27]. This command overrides any `pc_slave_response_to` command.

`pc_slave_wr_to` *6-bit_pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field. This field is the Timeout/Hold config register bits [16:12].

`pc_slave_wr_to_io` *6-bit_pse/value* **(Level 2)**

Override normal PC ASIC configuration of this field for I/O PCs. This field is the Timeout/Hold config register bits [16:12]. `pc_slave_wr_to_io` overrides any `pc_slave_wr_to` command.

`pc2_iopc_cic_perr_disable` **(Level 2)**

Cause POST to disable I/O PC ASICS from detecting parity errors on the CIC control buses. This is required for operation if any boards in the system have the "magic wire" ECO. This is generally true for all boards that have a PC below Rev 4 in the I/O position (PC 2 on that board), since such PCs do not work reliably without this ECO.

`phase_time_report` **(Level 2)**

Report the elapsed time for each POST phase.

`platform_name` *string* **(Level 2)**

See command `no_snmp_access`. If that command is invoked, this command can provide the platform name, normally obtained from the SNMP agent. If `no_snmp_access` is not invoked, POST ignores this command. If `no_snmp_access` is invoked and this command is not present, POST uses `$SUNW_HOSTNAME` as the platform name.

This command is effective only when used in a `.postrc` file in the current directory. If the `.postrc` file used is not in the local directory, POST ignores this command and generates a warning message.

`poll_timeout_mult` *mult_factor* **(Level 2)**

The timeout values used when tasks are assigned to host processors are multiplied by this integer value, which is in the range 1 to 100, inclusive. Sometimes, when very verbose messages have been enabled in a system

with many processors, a healthy processor is so delayed by the poll and display overhead that it exceeds the normal time allowed, causing it to be marked as failed. This command effectively extends all such timeout allowances. It could also be used as a field workaround for inappropriately short timeout values. See also `servmgr_time_report` and `no_poll_timeouts`.

`port_idle_use_pause` **(Level 3)**

Use the less flexible PORT_PAUSE PC BootBus feature instead of the PORT_IDLE feature.

`print_all_errors` **(Level 2)**

POST normally prints only the first message or two from any particular test; the test then quits. This command causes it to print all error messages and to continue testing.

Turning this function on can cause so many failure messages that the test timeout is exceeded, and a perfectly good processor can be marked as failed. To compensate, use the `poll_timeout_mult` command, as well.

`proc_freq_check_percent` {*float_%* | `off`} **(Level 2)**

During the JTAG integrity test phase, POST reads the maximum rated speed of each processor, and compares it to the speed of the system-wide processor clock distributed from the control board. If the clock speed exceeds the rated speed by greater than the specified percent (*float_%*), POST fails the processor. The default is 0.5%. If the argument is `off`, POST does not do the check.

`proc_freq_ratio` {`2 | 2:1 | 3 | 3:1 | 3:2`} **(Level 2)**

Use the specified ratio of processor frequency to system interconnect frequency. This value is used for reporting processor frequency in the post2obp structure, and for checking against a processor's rated maximum speed, in teststand environments lacking a real control board which can measure and report this frequency.

In systems that do have a control board, specifying this value causes POST to require that the measured ratio be within 0.5 percent of this value. If it is not specified, the measured ratio must be within 0.5 percent of the target ratio read from the SNMP server. See also the command `interconnect_MHz`.

`proc_runslow_warn_percent` **[***float_%* | *off***] (Level 2)**

POST normally checks the processor module's rated speed against the frequency provided by the control board to make sure processors are not being operated faster than rated. See `proc_freq_check_percent`. It also checks that processors are not running significantly slower than rated, which often means that the control board was misconfigured. If the speed is much slower than rated, POSE displays a NOTE about the possible misconfiguration.

The NOTE is printed only if the control board processor frequency is *proc_runslow_warn_percent* below the rated module frequency. The default value is 9%. This command allows it to be set to a different value, or have the check and NOTE disabled completely.

`proc_timestamp` *interval_secs* **(Level 2)**

When printing messages from the downloaded processor code of all host processors (those messages preceded by {*board.module*}), also print a timestamp of when the messages occurred; but print timestamps for any particular processor only when that processor prints a message and at least *interval_secs* has elapsed since the last timestamp for that processor. The form of the timestamp is {*board.module*} {*day/date/time*}, with 1-second resolution.

interval_secs is an integer for the number of seconds in the range of 0 to 86400 (24 hours; 60 times 60 times 24). Specifying 0 causes every message to get a timestamp.

`qt_missrefresh_err_enbl` **(Level 3)**

See the `quickturn` command. POST's default behavior with `quickturn` specified is to mask missed refresh errors in the MC ASIC. `qt_missrefresh_err_enbl` causes these refresh errors to be enabled to cause an arbstop. POST quietly ignores this command if the `quickturn` command is not invoked.

`quickturn` *KHz Mem_refresh_interval* **(Level 3)**

Declare that the system is running in a Quickturn (tm) emulation environment. *KHz* declares the system clock speed, and must be in the range 100 to 2000 (2 MHz), inclusive. *Mem_refresh_interval* is the refresh interval in system clock cycles for configuring the MC, and must be in the range 8 to 0xFFF = 4095. Both arguments must be specified.

Poll timeouts are set at a multiple of their normal value, based on the declared system frequency. The Memory Controller is set to use the specified

refresh interval. Other configuration properties may be changed appropriately when this command is invoked.

`recordstop_poll` **(Level 2)**

When polling for arbstop, also poll for recordstop.

`redlist_file` {*filepath* | `none`} **(Level 2)**

Command Line Equivalent: –`R` Use the values in the specified `redlist`(4) file rather than those in the default `redlist`(4) file, `$SSPVAR/etc/`*platform_name*`/redlist`. If `none` is specified, POST reads no redlist file. See `blacklist`(4) and `redlist`(4).

Be careful when using `redlist_file` on a production system; other SSP software does not know when POST is using a nonstandard `redlist`(4) file.

`scantool_simulate` **(Level 3)**

Intercept normal calls to the JTAG library functions and (crudely) simulate them internal to POST. This command is useful to Sun Microsystems Engineering during development to allow code debug when no real or hardware simulated system is present.

`servmgr_time_report` **(Level 2)**

When a host processor is assigned a task, such as running a test, POST's server manager sets a timeout by which it must complete or be considered failed. This flag causes the server manager to report how much time remained in the timeout when the processor reported back. This information is useful in determining the appropriate value of these timeouts.

`shared_mem` *brd boardmask* **(Level 3)**

Configure the specified system board to provide shared memory to the system boards in *boardmask*. *boardmask* must not include any boards in the specified board's domain, including itself. See the `domain` command for more information. This command does not configure the shared memory BAR/LAR in the CICs; host software normally does so. However, see the command `shmem_barlar`.

`shmem_barlar` *bar lar* **(Level 3)**

Configure the JTAG CIC shared memory base and limit address registers with the given *bar* and *lar* values. This command affects POST only for boards declared by the `shared_mem` command to have shared memory, and only a single set of *bar/lar* values is used for all such boards. The default, and the value for boards without shared memory, is 1 for *bar* and 0 for *lar* (no valid range). For boards with shared memory, these two values are added to the board's base physical address (mod 2**41) and the result is that board's *bar/lar* configuration. These two registers count increments of 64 Kbytes, with implied [15:0] = 0. Any 25-bit value is acceptable for both *bar* and *lar*, and *bar* may be greater than *lar*.

`skip_phase` *phase_name* **(Level 2)**

Skip the named POST phase. Only one phase can be specified per command, but any number of `skip_phase` commands can appear in `.postrc`. To print the available phase names, execute `hpost -?postrc`. These names are also printed as progress messages when POST runs if the verbose level is sufficiently high. See also the `interactive_use_postrc_skips` and `no_skip_phase_covers_npb` commands.

`skip_test` *test_id* **(Level 2)**

Skip the designated test. Only one test can be specified per command, but any number of `skip_test` commands can appear in `.postrc`. *test_id* is a number in the range 0 to 0x1FF, used as index and identifier for a POST test. The Test ID is printed, along with the test name, before each test is run, if the verbose level is high enough. Getting the Test ID in this way is the most appropriate method for users to determine the ID they should use with this command. The Test ID is also printed when tests fail.

This command provides higher resolution control over execution than `skip_phase`. It might also be a field workaround for bugs or hardware changes that cause inappropriate failures of some test.

Some POST tests may be internally declared non-skippable. A check for such declarations occurs at test execution time, rather than when POST reads the `.postrc` command. If a `skip_test` command is present for such a test, POST prints a single message that it is ignoring the command. See also the `interactive_use_postrc_skips` command.

`sysreset_ignore_redlist` **(Level 3)**

Allow redlisting of some system resources that physically do not exist, without causing POST to skip issuing the system reset during the *init_reset*

phase. This command is for use only in certain development environments. See `redlist`(4).

`system_red_any_fail` **(Level 2)**

Upon any POST failure, immediately fail and internally redlist the entire host system. This command is sometimes useful for doing postmortem analysis of failures in a debugging environment.

Since it obviously subverts any possibility of POST managing to configure around failed components, Sun Microsystems does not intend this command for normal use. You can think of it as a more extreme form of the command `board_red_any_fail`.

`trace_print` *board proc_module* **(Level 2)**

Enable the extremely verbose `tprintf` trace prints from the specified host processor. This command is off by default. To control trace prints for multiple processors, use more than one instance of this command.

`verbose` *level* **(Level 1)**

Command Line Equivalent: –`v` Control the amount of progress information printed by POST. Higher verbose levels result in more verbose output. *level* is an integer from 0 to 255, inclusive. For more information about levels, execute **`hpost -?verbose`**.

The system administrator can use this command, but must be aware that high verbose levels slow POST execution. In some extreme cases, it slows it enough to cause timeouts.

`verbose_print` *board proc_module* **(Level 2)**

Control the verbose `vprintf` printing from an individual host processor. This command turns on control for one specific processor. To do so for multiple processors, use multiple instances of this command. This command is enabled for all processors if the verbose level is high enough.

SEE ALSO `bringup`(1M), `hpost`(1M), `blacklist`(4), `redlist`(4).

NAME

redlist – list of system resources not to be touched

CAUTION

Contact your service provider before using this file. Certain changes to this file may cause your system to become unusable.

DESCRIPTION

`$SSPVAR/etc/`*platform_name*`/redlist` is an ASCII file that enables the system administrator or root to restrict, from the SSP, the configuration of the host system. It lists components that POST cannot touch, and whose state POST cannot change. POST reads the `redlist` file before preparing the system for booting, and passes along to OBP a list of only those components that have been successfully tested; those on the redlist are excluded.

The `redlist` file is very restrictive, and used in a very limited way, mostly in specialized debug in the lab. For example, a Sun Microsystems engineer may use it while working with experimental hardware that is missing components with which POST expects to communicate, even if just to freeze them.

Redlisted components are also considered effectively blacklisted.

Redlisting components carries a price in capability and performance. If any component on a board is redlisted, POST cannot reset that board; since some failures require a board reset to clear, the entire board becomes unusable and, in some cases, the entire system can become unusable.

Caution: Never use redlisting if blacklisting will do.

In the `redlist` file:

- Keywords are not case-sensitive.
- Any part of a line that starts with # is a comment.
- Numbers are assumed decimal unless preceded by `0x`, which indicates hexadecimal. Exception: a board number entered as one of [a - f] or [A - F] is assumed hexadecimal.
- Each line has one and only one keyword.
- The same keyword can be used on more than one line.
- Each keyword has one or more arguments. Each argument is shown as an integer and multiple integers are separated by a period.

Keywords

All value ranges shown below are inclusive.

`sysbd` *board*

Do not test or configure the specified system board, where *board* equals 0 to 15.

`proc` *board.pmod*

Do not test or configure the specified processor within the specified system board, where *board* equals 0 to 15 and *pmod* equals 0 to 3.

`abus` *abus*

Do not test or configure the specified address bus, where *abus* equals 0 to 3. The meaning of this command is that the corresponding CIC ASIC on all system boards is marked red. See the keyword `cic`, below.

`dbus` *dbus*

Do not test or configure the specified 72-bit half of the 144-bit data router, where *dbus* equals 0 to 1. The meaning of this command is that the corresponding half of the local data router on all system boards is marked red. See the keyword `ldpath`, below.

`ioc` *board.ioctl*

Do not test or configure the specified I/O controller within the specified system board, where *board* equals 0 to 15 and *ioctl* equals 0 to 1.

`scard` *board.ioctl.slot*

Do not test or configure the specified I/O adapter card within the specified I/O controller, which is within the specified system board. *board* equals 0 to 15, *ioctl* equals 0 to 1 and *slot* equals 0 to 3.

`mem` *board*

Do not test or configure memory on the specified system board, where *board* equals 0 to 15.

`mgroup` *board.group*

Do not test or configure the specified group of memory DIMMs within the specified system board, where *board* equals 0 to 15 and *group* equals 0 to 3.

`cplane` *half_centerplane*

Do not test or configure the specified Enterprise 10000 half- centerplane, which contains two buses and 72 bits of the global data router, where *half_centerplane* equals 0 to 1. The meaning of this command is equivalent to the combination of the `abus` and `dbus` commands for the buses contained in this half-centerplane.

`pc` *board.pc*

Do not test or configure the specified port controller ASIC within the specified system board, where *board* equals 0 to 15 and *pc* equals 0 to 2.

`xdb` *board.xdb*

Do not test or configure the specified data buffer ASIC within the specified system board, where *board* equals 0 to 15 and *xdb* equals 0 to 3.

`cic` *board.cic*

Do not test or configure the specified coherent interface controller ASIC within the specified system board, where *board* equals 0 to 15 and *cic* equals 0 to 3 (*cic* corresponds to an address bus on that board).

`ldpath` *board.dbus*

Do not test or configure the specified 72-bit half of the 144-bit local data router within the specified system board, where *board* equals 0 to 15 and *dbus* equals 0 to 1.

EXAMPLES

EXAMPLE 1 A Sample `redlist`(4) File

```
# Sun Microsystems, Inc.
sysbd 3 5 0xA        # Disable system boards 3, 5 and 10.
sysbd 3 5 A          # Disable system boards 3, 5 and 10.
PROC 4.0 6.2         # Disable Processor 0 on System Board 4, and
                     # Processor 2 on System Board 6.
ScarD 3.0.1          # Disable I/O Adapter 1 on I/O Controller 0 on
                     # System Board 3.
mem 2                # Disable all memory on System Board 2.
mlimit 0xE 2.64      # Restrict use of Memory DIMM Group 2 on System
                     # Board 14 to 64MB.
cIc 1.2              # Disable CIC ASIC 2 on System Board 1.
```

SEE ALSO

`hpost`(1M), `blacklist`(4)

NAME

ssp_resource – SSP processes resource file

CAUTION

Do not edit this file manually. Doing so may interfere with its use by SSP processes.

DESCRIPTION

`ssp_resource` is a general-purpose resource file used by SSP processes. It contains the following information, with period-separated fields:

platform_type.resource_name:resource_value

platform_type.id_name.resource_name:resource_value

where:

platform_type	The platform type of the SSP. Currently, this value is always `Ultra-Enterprise-10000`.
resource_name	The name of the resource. This name must be unique within this file.
resource_value	The value of the resource.
id_name	The name of the SSP process or daemon that currently owns this resource.

EXAMPLES

EXAMPLE 1 A Sample `ssp_resource`(4)

```
Ultra-Enterprise-10000.confMemAddrMap:000102030405 ... OEOF
Ultra-Enterprise-10000.edd.scripts:sysbrdtemp.tcl,centerplanetemp.tcl
Ultra-Enterprise-10000.HIGHasic:60000
```

FILES

The following file is supported:

`$SSPVAR/.ssp_private/ssp_resource`

SEE ALSO

`cbs`(1M), `edd`(1M), `snmpd`(1M)

NAME

ssp_to_domain_hosts – hostname/domainname file

CAUTION

Do not edit this file manually. Doing so may interfere with its use by SSP processes.

DESCRIPTION

`$SSPVAR/.ssp_private/ssp_to_domain_hosts` maintains a list of domain names, each with the name of its assigned SSP hostname to its right, as shown in the example, below.

EXAMPLES

EXAMPLE 1 A Sample `ssp_to_domain_hosts`(4) File

```
#
# Format: domain_name    ssp_hostname
#
xf2   xf2-ssp
ts4   ts4-ssp
```

SEE ALSO

`domain_create`(1M), `domain_remove`(1M), `domain_rename`(1M)