Ultra Enterprise 10000 SSP 3.0 Reference Manual

Sun Microsystems Computer Company
A Sun Microsystems, Inc. Business
2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

Part No: 805-0077-10
Revision A, March 1997

NAME

Intro – Enterprise 10000 SSP administration

DESCRIPTION

This section describes commands, scripts, and programs executed in the Enterprise 10000 SSP environment.

autoconfig(1M)	scantool / interconnect auto config
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
domain_create(1M)	create new domain or recreate old one
domain_history(1M)	display domain history
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
edd(1M)	event detector daemon
edd_cmd(1M)	send a command to **edd**(1M)
fad(1M)	file access daemon
fan(1M)	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
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
sigbcmd(1M)	send commands from SSP to domain

snmpd(1M)	system SNMP proxy agent
ssp_config(1M)	configure SSP control boards
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

autoconfig – scantool / interconnect auto config

SYNOPSIS

autoconfig [*board_name*]

DESCRIPTION

CAUTION: Only authorized service providers should execute this command.

autoconfig checks, in turn, the chip revisions of all configured system boards and the two centerplane halves in the Enterprise 10000 system. It then looks at the file **$SSPVAR/data/Ultra-Enterprise-10000/common/board/sysboard/chip.ids** for system boards, or **$SSPVAR/data/Ultra-Enterprise-10000/common/board/centerplane/chip.ids** for centerplane halves, 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 **$SSPVAR/data/Ultra-Enterprise-10000/***platform_name***.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.

OPTION

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

$SSPVAR/data/Ultra-Enterprise-10000/common/board/sysboard/chip.ids

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

$SSPVAR/data/Ultra-Enteprise-10000/common/board/centerplane/chip.ids

Contains the IDs for all the chips on a centerplane.

$SSPVAR/data/Ultra-Enteprise-10000/common/board/sysboard/board.sigs

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

$SSPVAR/data/Ultra-Enteprise-10000/common/board/centerplane/board.sigs

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

$SSPVAR/data/Ultra-Enteprise-10000/$SUNW_HOSTNAME.config

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

$SSPVAR/data/Ultra-Enteprise-10000/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.

$SSPVAR/data/Ultra-Enteprise-10000/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.

$SSPVAR/data/Ultra-Enteprise-10000/common/board/centerplane/rev*?*/*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*

DESCRIPTION

Caution: This command is for use by Sun Manufacturing personnel only.

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

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

–**w** (Write) Assign the specified part number and serial number to 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	A 3-digit numeric category code. Currently, this code is 612
aaaa	A 4-digit numeric base part number (assembly number)
ss	A 2-digit numeric part number suffix
rr	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

board_id –**b sb** –**n 1**
Display information about Board 1 on the system board.

board_id –**w** –**b sb** –**n 1** –**s 37KR59** –**p 501-2341-12-01**
Assign part number 501-2341-12-01 and serial number 37KR59 to Board 1 on the system board.

board_id –**b cp** –**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**.

NAME

bringup – configure and boot the domain

SYNOPSIS

bringup [–**f**] [–**F**] [–**p** *proc*] [–**Q** *boot_proc*] [–**gvCL**] [–**A** {**on** | **off**}] [–**l** *level*] [–**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**.
4. Executes **hpost**(1M) with its –**C** option to configure the centerplane then the domain, or without it to configure just the domain.
5. Starts **obp_helper**(1M) and **netcon_server**(1M) to proceed with the OBP and operating system boot sequence and set up the network console.

6. Updates the MIB according to the final domain configuration.

OPTIONS

–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.

Note that if **–f** is used, **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).

–Q Send a request for **hpost**(1M) to execute a faster, limited version of POST. Use **–Q** only as a means to recover from an Ultra Enterprise 10000 software crash.

–L Send a request for **hpost**(1M) to use its **–s** and **–v10** options, sending all output to the syslog.

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

–g, **–l** *level*, **–p** *proc*, **–C**, **–X** *blacklist_file_pathname*
bringup passes these options to **hpost**(1M).

–A {**on | off**}, **–D** {**on | off**}
bringup passes these options to **obp_helper**(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.

ENVIRONMENT

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

SEE ALSO

check_host(1M), **hpost**(1M), **netcon_server**(1M), **obp_helper**(1M), **snmpd**(1M), **sys_reset**(1M) in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

boot(1M) in *man Pages(1M): System Administration Commands* of the *SunOS Reference Manual*

NAME

cb_prom – upgrade PROM

SYNOPSIS

cb_prom [–**d** *file*] [–**e**] [–**p** *file*]

DESCRIPTION

Caution: Only authorized service providers should use this command.

cb_prom manipulates the contents of the 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

–**d** *file*	Dump the contents of the Flash PROM to *file.*
–**e**	Erase the contents of the Flash PROM.
–**p** *file*	Program the Flash PROM with the contents of *file.*

Notes

Execute this command only when logged on as user 'ssp' on the SSP used by the Control Board as a boot (TFTP) server.

The **/tftpboot** directory is used for file transfers between the Control Board and the SSP, and should be writable by user 'ssp'.

NAME

cb_reset – reset and reboot hung control board

SYNOPSIS

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

DESCRIPTION

In its default form **cb_reset** sends a fixed Ethernet packet to the control board(s) of the Enterprise 10000 system specified by the SUNW_HOSTNAME environment variable, causing the boards to reset and reboot themselves.

OPTIONS

–**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 Enterprise 10000 system. Otherwise, it sends it to the specified board on the system specified by the SUNW_HOSTNAME environment variable.

–**v**
: (Verbose mode) Display a message confirming that the packet has been sent, along with the Ethernet addresses of both the source (the Enterprise 10000 system that sent the packet) and destination (the control board that received it).

FILES

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

SEE ALSO

edd(1M), **ssp_startup**(1M)

NAME

cbs – control board server

SYNOPSIS

cbs

DESCRIPTION

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

The **cbs** server program provides central access to the Enterprise 10000 control board for client programs running on the SSP. It has sole access to the 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 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

$SSPVAR/data/Ultra-Enterprise-10000 – Enterprise 10000 JTAG scan database
$SSPVAR/pid/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** considered the domain down, displays `Host is DOWN`, and returns a status 1.

OPTION

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

NOTE

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

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.

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

Note: Many steps are required to create a domain. See the section "Domains" in Chapter 3 of the *Ultra Enterprise 10000 SSP 3.0 User's Guide*.

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 unique in the host's database.
- The system boards are all present and none are in use.

OPTIONS

–**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 single comma or space.

–**o** *os_version*
The version of the operating system to be running on the domain.

–**p** *platform_name*
The name of the 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 Enterprise 10000 system is SUNW,Ultra-Enterprise-10000.

EXAMPLE

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

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.

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.

ERRORS

`One or more of the system boards in` *sysboard_list* `are being used by another domain.`

`The` *domain_name* `already exists.`

`There are no host ids available.`

If you get this message, call your service advisor.

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

–**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

domain_history –**d** junior

Displays configuration information about the previously existing domain named junior.

domain_history –**d** junior –**r**

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

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_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 cannot remove a domain until it has been shutdown.

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

```
The following subdirectories contain domain-specific infor-
mation, 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

domain_name `is still up, must shutdown first.`

The specified domain is running the operating system.

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_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*

DESCRIPTION

domain_rename changes the name of a domain from that specified by –**d** to that specified by –**n**. It does so by updating the SSP configuration.

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

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.

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

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, and system boards.

OPTIONS

–**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.

ERRORS

The *domain_name* `does not exist.`

domain_status was unable to find a domain of the specified name.

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 the current SSP session control the specified domain. **domain_switch** also changes the **csh**(1) 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

```
xf1-ssp:xf1% domain_switch xf1-deux
Switch to domain xf1-deux
xf1-ssp:xf1-deux%
```

NAME

edd – event detector daemon

SYNOPSIS

edd [–**a**] [–**d**]

DESCRIPTION

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

edd is a key component of the 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 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**(4) 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.

OPTIONS

–**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

$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 [–**x** *cmd*]

DESCRIPTION

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

edd_cmd is a utility tool. When executed with no argument it outputs the current execution state – either **started-monitoring** or **stopped-monitoring** – of the **edd**(1M) daemon.

When executed with its –**x** option, **edd_cmd** makes **edd**(1M) start or stop event-detection monitoring, or reread its configuration files. **edd**(1M) uses the **cbs**(1M) daemon to do the starting or stopping.

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

–**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

The following example shows how to use **edd_cmd** to check a state, change that state, then check it again:

```
% edd_cmd
EDD: Control = stop, State = stopped-monitoring

% edd_cmd -x start

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

FILES

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

SEE ALSO

edd(1M), **snmpd**(1M), **straps**(1M)

NAME

fad – file access daemon

SYNOPSIS

fad

DESCRIPTION

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

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

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

$SSPVAR/pid/fad.pid – process ID file

SEE ALSO

machine_server(1M) in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

fad_files(4) in *man Pages(4): Ultra Enterprise 10000 SSP File Formats*

lockf(3C) in *man Pages(3): Library Routines* of the *SunOS Reference Manual*

NAME

fan – control fan power and speed

SYNOPSIS

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

Each 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.

Valid fan speed options are **nominal** and **fast**, and apply to all fans in the system. Once you change the fan speed, it remains at your new setting until changed again via the **fan** –**s** command.

Valid fan power options are **on** and **off**. You can control power to each fan tray.

The fan trays are organized as follows:

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

–**l** {**front | rear**}
Display power and speed of the fans at the front or rear of the machine. If used with –**p**, –**l** specifies whether the fans at the front or rear of the machine are to be turned on or off.

–**s** {**nominal | fast**}
Set the speed of all fans as specified, to nominal or fast. All fans always run at the same speed. Nominal is the default.

–**t** *FanTrayList* Display power and speed of the specified fan trays. If used with –**p**, –**t** changes power to the specified fan trays. *FanTrayList* is a space-separated list of fan tray numbers expressed as integers, 0 to 15 inclusive. If you change power to one fan tray, the opposing fan tray is also affected. See the examples, below.

–**p** {**on | off}** Control power to either the fan shelves specified by the –**l** option or the fan trays specified by the –**t** option. If neither –**l** nor –**t** is specified, power to all fans is affected; if both are specified, the fan shelf takes precedence. See the examples, below.

EXAMPLES

Example 1

Each of the following commands turns on power to both fan trays 10 and 14, which are opposing trays.

```
fan -p on -t 10

fan -p on -t 14

fan -p on -t 10 14
```

Example 2

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

```
fan -s fast
```

Example 3

The following command turns on 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 on -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 Enterprise 10000 system specified in the SUNW_HOSTNAME environment variable.

OPTIONS

–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 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

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

SEE ALSO

snmpd(1M)

NAME

hostint – interrupt processor, dump kernel core

SYNOPSIS

hostint [–**p** *proc*] [–**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. It obtains the boot processor from the **snmpd**(1M) daemon running on the SSP.

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.

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**(1M). If this step is successful **edd**(1M) saves processor state specific to the reset and initiates a bringup.
5. If steps 1 through 4 fail, execute **bringup**(1M) with its –**f** option.

OPTIONS

–**p** *proc* Send the interrupt signal to the specified processor rather than the boot processor.

–**v** Verbose Mode. Display information to stdout.

ENVIRONMENT

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

SEE ALSO

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

FILES

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

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). Upon successful execution, a *resetinfo* file that contains pertinent processor information is generated.

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 via 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 **edd**(1M) saves processor state specific to the reset and initiates a bringup.
5. If steps 1 through 4 fail, execute **bringup**(1M) with its **–f** option.

ENVIRONMENT

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

FILES

$SSPVAR/adm/$SUNW_HOSTNAME/resetinfo

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 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 &

ENVIRONMENT

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 *ssp_host*
setenv DISPLAY *ssp_host***:0**
hostview &

For more information about Hostview and its various screens, see the *Ultra Enterprise 10000 SSP 3.0 User's Guide*.

NAME

hpost – control and sequence POST through JTAG

SYNOPSIS

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

hpost
[–**Ccfnqs**] [–**D**[*boardmask***,**][*path*]] [–**d** "*comment*"] [–**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**]
[–**Z***proc*]

DESCRIPTION

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

The POST (Power On Self Test) program probes and tests the components of uninitialized 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.

–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**.

–f Ignore failure on creation of the POST lock file, which prevents more than one POST process from running simultaneously in the same domain.

–**g**{*path* | **none**}
Create or do not create (**none**) a log of screen output and send it to the specified *path*. If *path* is not specified the log is placed in:

$SSPVAR/adm/$SUNW_HOSTNAME/post/post*mmdd.hhmm***.log**

–**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** 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 (see **DIAGNOSTICS**, below).

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.

This mode of **hpost** is normally invoked by an SSP event detection daemon after it creates a state dumpfile. See the −**D** option.

−**Z***proc* (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

./.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

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.)

SEE ALSO

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

NAME

machine_server – multi-purpose server

SYNOPSIS

machine_server

DESCRIPTION

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

The **machine_server** daemon performs the following functions:

- Services TCP port registration requests from **netcon_server**(1M) and UDP port registration requests from the 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

$SSPVAR/pid/machine_server.pid – process ID file

/etc/services – reserved port specification

SEE ALSO

netcon_server(1M), **snmpd**(1M)

NAME

netcon – network console

SYNOPSIS

netcon [**–f | –l | –g**]

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 specified domain. 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 Domain Console windows are running for that domain, your session comes up with Unlocked Write permission. If you do so when one or more other sessions are running, your Domain Console Window comes up in Read Only mode, regardless of the permission settings of the other sessions.

OPTIONS

netcon executes with only one option at a time. If you specify more than one option, it runs with the option that carries the highest precedence, as follows: **–f**, **–l**, then **–g**.

–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 (via **~^**) or terminate the **netcon –f** session (via **~.**).

–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. See NOTES, below.

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.
- **~&** 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

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.

ENVIRONMENT

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

SEE ALSO

netcontool(1M), in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

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

boot(1M) in *man Pages(1M): System Administration Commands* of the *SunOS Reference Manual*

NAME

netcon_server – network console server daemon

SYNOPSIS

netcon_server –**p** *boot_proc*

DESCRIPTION

Caution: Do not execute this daemon manually.

netcon_server is a daemon executed by **bringup**(1M). It 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.

ENVIRONMENT

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

SEE ALSO

bringup(1M), **ssp_startup**(1M), **netcon**(1M) in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

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 WConfigure button.

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

Terminal Type
Press the WXterm, WShell Toolf1 WCommand ToolF1 button. 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.

NOTE: 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 WDone button, the WConnect 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 WConnect, **netcontool** executes the **netcon (1M)** program, the Domain Console Window appears in the specified terminal type, and the all buttons in the **netcontool** Window except `Configure` and WConnect 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

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.

OPTIONS

–**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. This 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.

–**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

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

SEE ALSO

bringup(1M) in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

boot(1M) in *man Pages(1M): System Administration Commands* of the *SunOS Reference Manual*

NAME

power – control power

SYNOPSIS

power

power –**v** [–**sb** *sb_list*] [–**cb 0 | 1**] [–**csb** *csb_list*] [–**p** *p_list*] [–**ps** *ps_list*] [–**d | –q**]

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

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

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

power –**m**

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

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

power –**B** –**off** [–**d | –q**]

DESCRIPTION

The **power** command lets you view and control certain aspects of the 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 supports boards and the system boards is displayed as follows:

```
Centerplane:1
     //average voltage//
           5.0VDC HK:  5.086099V
           5.0VDC HK:  5.041999V
           3.3VDC HK:  3.494873V
           3.3VDC Vdd: 3.302002V
           3.3VDC Vdd: 3.299561V
           3.3VDC Vdd: 3.300781V
System Board:0
     //average voltage//
           3.3VDC Vdd: 3.281250V
           5.0VDC HK:  5.039549V
           3.3VDC HK:  3.424072V
           Vdd Core:   3.341064V
           5.0VDC Vcc: 5.015049V
```

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 an common acronym for +3.3 volt DC power.
- `Vcc` is a common acronym 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.

OPTIONS

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

–B –off Turn off power to the entire 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 supples.

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

–cb 0|1 Turn turn off 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.

–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.

–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**.

–**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.

–**ps** *ps_list* – **For use by service providers only**
Turn off 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 command **power -off** –**ps** *ps_list* displays the following message:

```
Warning: This command will disable x 48-volt power
supplies, leaving y supplies left for the z physically
present system boards. Continue? (y/n)
```

where:
x is the number of 48-volt power supplies to be disabled
y is the number of 48-volt power supplies to be left enabled
z is the number of system boards physically present in the system

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 3 ps 9 sys bds need 6 ps

2 sys bds need 3 ps	10 sys bds need 6 ps
3 sys bds need 3 ps	11 sys bds need 7 ps
4 sys bds need 4 ps	12 sys bds need 7 ps
5 sys bds need 4 ps	13 sys bds need 7 ps
6 sys bds need 5 ps	14 sys bds need 8 ps
7 sys bds need 5 ps	15 sys bds need 8 ps
8 sys bds need 5 ps	16 sys bds need 8 ps

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

–sb *sb_list* 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.

–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 | csb** – **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 or centerplane support boards. The values take effect when the boards are next powered on.

power –m *margin_list* **–sb** *sb_list* immediately margins the power supplies for the system boards specified in *sb_list*.

Warning: Margining power supplies more than +/- 5 percent may cause damage to the hardware.

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 or centerplane support board voltages as follows:

Vcc, 5 volts
Vdd, 3.3 volts
Vcore 2.5 to 3.7 volts

Note that Vcore is displayed as `Vdd Core` in **power** command output.

Vdd Core is a processor's internal voltage power supply. The target voltage depends on the type of processor modules installed.

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

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

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

power –m Vcc.+2 Vdd.-2 Vcore.4 –s sb
Change the margin voltage value for all system boards as follows: increase Vcc by 2 percent, decrease Vdd by 2 percent, and increase Vcore by 4 percent. Note that this change in margining will take effect when the system boards are next powered on.

power –m Vcc.+3 Vdd.+3 Vcore.3 -sb 5 6 7
Increment by 3 percent the target values of Vcc, Vdd and Vcore supplies on system boards 5, 6 and 7. The change occurs immediately, while these system boards remain on.

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

power –off –ps 2 3
Turn off 48-volt power supplies 2 and 3.

NAME

redx – remote emulation debugger

SYNOPSIS

redx [–**chnlq?**] [*file* [*arg* ...]]

redx [–**q**] –**x** *redx_interpreter_cmd* ...

DESCRIPTION

WARNING: 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.**

redx is a debug and maintenance support program for the 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

redx –?

Display explanation of options.

redx –x ?

Display more information.

NAME

sigbcmd – send commands from SSP to domain

SYNOPSIS

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

DESCRIPTION

Warning: 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.**

The SSP utility **sigbcmd** performs one of the following actions: sends the given command, *cmd*, to the 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).

–**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. –**f** makes it do so, regardless.

–**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

sigbcmd –p 4 obp

Make target domain enter OBP.

sigbcmd –i –p 4 panic

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

ENVIRONMENT

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 Enterprise 10000 platform

SYNOPSIS

snmpd [–**p** *port*]

DESCRIPTION

Caution: Do not execute this daemon manually; it is automatically invoked. **snmpd** 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.

snmpd is an SNMPv1 (Sample Network Management Protocol, Version 1) proxy agent for the Enterprise 10000 system. See the 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 supports the SNMP requests **set**, **get** and **getnext**.

Each instance of **snmpd** 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** relies on the file access daemon, **fad**(1M), to supply the information. Enterprise 10000 MIB information that corresponds to system resources is retrieved via the control board server, **cbs**(1M).

OPTIONS

–**p** *port* Use the specified UDP port for incoming SNMP requests. If –**p** is not specified, the first instance of **snmpd** allocates the standard SNMP port number, 161, for communication. Subsequent instances allocate arbitrary UDP ports. **machine_server**(1M) daemon for port assignment.

FILES

$SSPETC/snmp/agt/Ultra-Enterprise-10000.snmpd.cnf – configuration file
$SSPETC/snmp/Ultra-Enterprise-10000.mib – Enterprise 10000 MIB definition file
$SSPETC/snmp/Ultra-Enterprise-10000.dat – Enterprise 10000 MIB data file
$SSPVAR/pid/snmpd.*platform_name***.pid** – process ID file

SEE ALSO

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

NAME

ssp_config – configure SSP control boards

SYNOPSIS

ssp_config [*spare*]

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 via name services or local files. It prompts for them, too, if its attempt fails. The system will not fully initialize until **ssp_config** has the information it needs.

When you are logged on as root and executing **ssp_config** manually, you must reboot the system immediately after **ssp_config** completes.

OPTIONS

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 *Ultra Enterprise 10000 System Service Processor 3.0 Installation Guide*, which is part of the Service Documentation Set sent with your Ultra Enterprise 10000 system.

FILES

/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)

NAME

ssp_startup – invoke SSP daemons

SYNOPSIS

ssp_startup

DESCRIPTION

Note: Do not execute this command from the command line. The **ssp_startup** script is normally initiated automatically on each reboot by the SunOS command **inittab**(4).

The **ssp_startup** script starts the following Enterprise 10000 system daemons in the proper order for the current configuration (the order may differ from that shown here, and is important):

- **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.

After **ssp_startup** has completed, you can use **domain_create**(1M) to create the domain, then execute **bringup**(1M) to bring it up.

FILES

$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 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.

Note: Before executing this command make certain that all domains have been halted and removed via **domain_remove**(1M).

FILES

$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) in *man Pages(1M): Ultra Enterprise 10000 SSP Administration Commands*

halt(1M), **shutdown**(1M) in *man Pages(1M): System Administration Commands* of the *SunOS Reference Manual*

NAME

straps – SNMP trap sink server

SYNOPSIS

straps

DESCRIPTION

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

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 162. 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

$SSPETC/snmp/straps – stream socket device file
$SSPVAR/pid/straps.pid – process ID file

SEE ALSO

snmpd(1M)

NAME

sys_clock – display / change / set system clock frequencies

SYNOPSIS

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

DESCRIPTION

Caution: Sun Microsystems discourages execution of this command with any of its command-line options, which attempt to reset or change target frequencies. Such an attempt can result in poor performance or a system crash.

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 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

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 Enterprise 10000 system the next time **sys_clock** –**s** is executed. To use these options to immediately change the settings of the 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** 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** or **three-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 Enterprise 10000 system is powered on.

EXAMPLES

Example 1: Display target and actual system clock frequencies.

```
% sys_clock

Current Clock Frequencies:
--------------------------
Interconnect:  83.98 Mhz
Processor: 167.95 Mhz
JTAG: 5.00 Mhz

Targeted Clock Frequencies:
---------------------------
Interconnect:  83.98 Mhz
Proc Clock Ratio: two-to-one
JTAG:   5.00 Mhz
```

Example 2: Modify target clock frequencies maintained by **snmpd**(1M) to the given values.

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

Example 3: Set system clock frequencies on the Enterprise 10000 system.

```
% sys_clock –s
```

FILES

$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 [–**d**] | [–**f** *filename*] | [–**m** *yymmddhhmm*] | [–**k** *key* {–**s** *serial_no* | –**h** *host_id*}]

DESCRIPTION

sys_id displays or changes the host ID and Ethernet number on an 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. Any other option entered along with this option is ignored.

–**m** *yymmddhhmm*

Set the manufacturing date and time of the IDPROM as the specified year, month, day, hour and minute. If the manufacturing date is already set, –**m** overwrites it and displays a message showing the new date and time.

–**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* [–**s** *serial_no* | –**h** *host*]

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. You must also specify either –**s** or –**h**.

–**s** 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 for hex format. To specify it in hex, use a prefix of **0x** or **0X**. Valid serial numbers are in the range **0xA65000-0xA65FFF**.

–h 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**.

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.

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

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

OPTIONS

–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*

DESCRIPTION

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

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

OPTIONS

–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

DESCRIPTION

Caution: Only authorized service providers should execute this command.

thermcal_config generates the SSP configuration file, **thermcaldata.tcl**, which 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.

Note: Execute this command only when adding a system board or centerplane half that is new to the system – that is, one that was not previously present - or when moving a system board or centerplane half to a different location.

FILES

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