Sun HIPPI/P 1.0 Character Device Interface Reference Manual

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

Part No: 805-7708-10
March, 1999

Contents

CHAPTER

Maintenance Commands

NAME

Intro – HIPPI Administration

DESCRIPTION

This section describes commands executed in the HIPPI environment.

LIST OF COMMANDS

blast(1M)	HIPPI character driver transmitter
hipadmin(1M)	HIPPI configuration program
hippi(1M)	HIPPI control and status utility
hippiarp(1M)	HIPPI ARP (address resolution) display and control
hippid(1M)	HIPPI support daemon
hippidb(1M)	HIPPI driver debug trace display and control
hippidisp(1M)	HIPPI NIC display utility
hippidmpd(1M)	HIPPI dump daemon
hippidnld(1M)	HIPPI driver RunCode download utility
hippistat(1M)	HIPPI hardware statistics
hippitb(1M)	HIPPI driver debug trace display
hippitune(1M)	HIPPI driver debugging and performance tuning utility
sink(1M)	HIPPI character device receiver

NAME

blast – HIPPI character driver transmitter example

SYNOPSIS

blast [–2PMrkCc] [–D *unit*] [–I *ifield*] [–l *size*] [–n *writes-per-pass*] [–m *passes*] [–u *ULP*] [–R *file*] [–U]

AVAILABILITY

SUNWhip

DESCRIPTION

The `/etc/opt/SUNWconn/bin/blast` program provides sample code for testing and using a HIPPI character device. The code includes most of the ioctl() settings for transmitting data. For more information about these ioctl() settings, see the character device interface user's guide and reference. The `blast` sample code also can be used with `sink(1M)` to analyze system performance.

`blast` performs a write operation multiple times. The size of the packet written is specified by *size*. The number of passes is specified by *passes*. The number of times the packet is written during each pass is specified by *writes-per-pass*. For example, to send 800 2MB packets 10 times, you would give the following arguments:

```
-l 0x200000 -n 800 -m 10
```

You can also combine *writes-per-pass* into a single packet by using the `-P` flag. For example, to sent 10 packets, each 1.6 GBytes in size, you could use the following arguments:

```
-l 0x200000 -n 800 -m 10 -P
```

By default, `blast` writes packets that are four KBytes in size.

`blast(1M)` works with the `sink(1M)` sample program, which reads packets. `sink` reads the HIPPI-FP header in addition to the packet data, so the sink packet size must be at least eight bytes larger than the blast packet size.

OPTIONS

All agruments are optional. Default values are as shown.

–2	Run blast in double-threaded mode. In this mode the two threads write to the device simultaneously. This argument cannot be used with `-P` or `-C`.
–P	Encapsulate each pass within a single packet. This argument cannot be used with `-2`.
–r	Send random data rather than printable ASCII characters. If you use this option, the checking option within `sink(1M)>` is inoperative.

–C	Establish a long-term connection, allowing the transfer of multiple packets. This argument cannot be used with -2.
–c	When used with the -r option, cause a new random packet to be generated for each write. This option simulates a real world application.
–D *unit*	Use the specified HIPPI card. This option is used for paltforms that support multiple HIPPI cards.
–I *Ifield*	Set the Ifield for the connection to the specified value. The default is zero. For more information, see the *Sun HIPPI Installation and User's Guide*.
–l *size*	Use the specified size for the buffer passed to each write() call. The default is 4096 bytes.
–n *writes-per-pass*	Use the specified number of writes per pass. The default is 500. When -P is specified, the end of a pass designates the end of a packet. When -C is specified, the end of a pass indicates when the connection is dropped.
–m *passes*	Perform the specified number of passes. The default is one.
–u *ULP*	Use the specified upper layer protocol identifier for the framing protocol header. This identifier must match the upper layer protocol identifier specified by sink(1M). The default is 0x82.
–R *file*	Record performance information in the specified file.
–U	Send unknown-length (infinite) packets. The actual packet length is specified by -1. The packet length must be a multiple of 8 bytes.

SEE ALSO **hippi**(1M), **sink**(1M)

NAME

hipadmin – HIPPI configuration program

SYNOPSIS

hipadmin [–u]

AVAILABILITY

SUNWhip

DESCRIPTION

For each HIPPI card present in the system, `/etc/opt/SUNWconn/bin/hipadmin` interactively prompts the user to enter the IP address, netmask, and HIPPI switch address in the following format:

address netmask switch_address

The information is then stored in:

`/etc/opt/SUNWconn/hippi/hip`*n*`.conf`

where *n* is an integer, 0 through 3, inclusive.

Each time you run `hipadmin` you must then edit the `hippiarp.conf` utility to update the ARP configuration information, then run `/etc/init.d/hippi start`. This sequence causes execution of `hippiarp.conf`.

`hipadmin` also prompts for each NIC's EEPROM update.

`hipadmin` must be executed after the HIPPI package has been installed, and you must be root to do so.

OPTIONS

The following options are supported:

–u Update the NIC(s) EEPROM contents.

SEE ALSO

hippi(1M), **hippitune**(1M), **hippidnld**(1M), boot(1M)

NOTE

In the absence of HIPPI hardware, this utility asks for the number of interfaces to be configured. Based on the response, it creates the aforementioned `hip`*n* files. Then, when the hardware is installed, you need to execute one of the following commands to create HIPPI `/device` nodes and `/dev` links:

`ok boot` *diskname* `-r`

`# drvconfig, devlinks`

NAME

hippi – HIPPI driver RunCode download utility

SYNOPSIS

hippi [on][[short]|[long]] [[fp]|[ph]] [[network]|[loopback]] [[switched]|[direct]] [*unit*]

hippi off [dump] [*unit*]

hippi restart [dump] [*unit*]

hippi status [*unit*]

hippi accept [*unit*]

hippi reject [*unit*]

hippi version

hippi cards

AVAILABILITY

`SUNWhip`

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippi` displays the state of the HIPPI driver and hardware or queries the current status or version of the network cards.

Any user can execute this command to obtain the status, version number, or number of cards on the network, but only super user can execute it with its other options.

OPTIONS

The following options are supported:
on [short | long] [fp | ph] [network | loopback] [switched | direct] [*unit***]**

Load RunCode (firmware) into the HIPPI device and start the device. The system boot processing automatically loads RunCode and starts the device if it is not already running. If the driver is already active, the command fails. When you execute `hippi on` with any of its options (for example, `short` or `long`), the option value is remembered as long as the system remains up, and is reused on the next invocation of `hippi on`. You can use `hippitune(1M)` to permanently set default values.

IP datagrams over HIPPI have a maximum MTU size of 65288 bytes. All HIPPI traffic should be limited to 64-Kbyte packet size when IP datagrams are sent over a HIPPI network. Setting the short option limits packets to 64 kilobytes, while setting the long option permits any size of packets to be sent over the network. `long` also enables you to use all of the connection-control and packet-control facilities.

Use `fp` to set receive processing to HIPPI-FP mode, or `ph` to set it to HIPPI-PH mode. In `fp` mode, the NIC multiplexes the incoming packets based on the value in the ULP field of the FP header. In `ph` mode, all incoming packets go to the same place. The network driver cannot be used in `ph` mode.

The NIC usually passes HIPPI packets through the network interface and out over the network. It also accepts packets from the network and you can use the loopback option to place it in internal-loopback mode. In this mode, all packets that are sent out are internally passed back to the receive interface. All connection attempts from the network are rejected.

The NIC usually is connected to a HIPPI-SC switch (switched). To connect it to another NIC, use the direct option.

As installed, the defaults for `hippi on` are `short`, `fp`, `network`, and `switched`. You can use `hippitune(1M)` to change the defaults. But to change the operating mode (for example, to `short`, `long`, `fp`, `ph`, `network`, `loopback`, `switched`, or `direct`), you must deconfigure the NIC by using the `ifconfig down`. See `ifconfig(1M)`.

off [dump]

Immediately stop the HIPPI RunCode and place the system into a state in which it can neither accept nor transmit packets. All pending reads and writes are completed with EINTR. All CDI calls complete with ENODEV errors until you issue either a `hippi on` or `hippi restart` command.

The `dump` option causes a dump file to be generated. The dump file contains the current state of the driver and RunCode. Customer support can use the dump file to diagnose a problem.

restart [dump]

Stop the RunCode. A read or write that is actively passing data is completed with EINTR, and the packet is truncated. Reads and writes that are waiting to use the HIPPI device are not affected. Firmware is loaded and started. Processing continues with the operation after the failed operation.

The dump option causes a dump file to be generated. The dump file contains the current state of the driver and RunCode.

status

Query current system status and report whether the system is on or off.

If the system is on, other flags (for instance, `accept`, `reject`, and `long`) indicate if the system is accepting or rejecting connection requests and if the system allows transmission of long packets. `IS_LOOPBACK` is set when the NIC discovers that it is connected to a loopback cable. `IS_DIRECT` is set when the NIC discovers that it is directly connected to another NIC. `LINK_ON` and `LINK_OFF` reflect the state of the optical link. `RUNCODE_ON` and `RUNCODE_OFF` reflect the operation of the RunCode. Other statistics are:

SRC connections The number of connections generated.

SRC packets The number of packets sent.

SRC failures The number of errors encountered during an attempt to transmit packets. No breakdown of errors on transmission is provided. In particular, connection timeouts, connection rejects, and sequence errors are all counted by this one multipurpose counter.

DST packets The number of packets received.

DST rcv on bad ulp The number of received packets that are destined for a non-active ULP.

DST hippi-le drop The number of packets dropped due to lack of resources in the IP stack.

DST data errors The number of packets received with data errors (either parity or LLRC).

DST sequence err The number of packets received with HIPPI sequence errors.

DST sdic lost The number of times the interconnect signal dropped.

accept

Set the system to a mode in which it accepts incoming connection requests. This is the default mode. Use this option to resume accepting connections after you have issued the reject option to reject them. The device must be in the on state for this command to work.

reject

Set the system into a mode in which it rejects future incoming connection requests. This command does not affect established connections. The device must be in the on state for this command to work.

version

Report the driver version number and RunCode version number of each NIC in the system.

cards

Report the number of NICs in the host system.

SEE ALSO

hippid(1M), **hippitune**(1M), **hippidisp**(1M), **hippistat**(1M)

HIPPI/P 1.0 Installation and User's Guide

HIPPI/P 1.0 Character Device Interface User's Guide and Reference Manual

NAME

hippiarp – HIPPI ARP (address resolution) display and control

SYNOPSIS

hippiarp *hostname*

hippiarp –a [*unit*]

hippiarp –h [*unit*]

hippiarp –c [*unit*]

hippiarp –s *hostname Adapter-ULA logical-address* [*unit*] [*temp*] [pub] [dnd]

hippiarp –d *hostname*

hippiarp –D *logical-address* [*unit*]

hippiarp –l *logical-address* [*unit*]

hippiarp –i [*unit*]

AVAILABILITY

SUNWhip

DESCRIPTION

When entered with only its *hostname* option, `/etc/opt/SUNWconn/bin/hippiarp` displays the Internet-to-HIPPI adress translation table entry used by the Address Resolution Protocol for HIPPI (see RFC 1374) for the specified host.

`hippiarp` is an extended `arp(1M)` utility that performs the same functions as `arp` except for the –`f`, –`u`, and –`trail` options. `hippiarp` provides additional functions that are specific to HIPPI, and it provides ARP address translation information for hosts that do not support ARP over HIPPI.

When a destination does not support ULA (Universal LAN Address, also known as the IEEE Universal MAC Address), the ULA is entered as 0:0:0:0:0:0. The utility creates a locally administered ULA that uses the logical address as the low-order 12 bits of the ULA. IP packets directed to the host are sent with zero as the ULA in both source and destination fields. The driver always accepts packets with a zero as the ULA.

HIPPI logical addresses are 12-bit numbers that are used by the switch to route the packet. Addresses in the range 0xF90 through 0xFFF, inclusive, are reserved (see HIPPI-SC) and may not be set by this utility. When sending an IP packet to a destination host, the driver will set the CAMP-ON and logical routing bits in the I-field for this packet.

The adapter may be connected to a switch (switched mode) or directly connected to another adapter (direct mode).

OPTIONS

The following options are supported:

hostname

Specify *hostname* by name or number, using Internet dot notation.

unit

unit is expressed as `hip`*n*, where *n* is an integer 0 to 3, inclusive. To view a list of HIPPI devices installed on the system, you can execute `hippi version`. See **hippi**(1M). In single-adapter configurations, *unit* is always optional. In multi-adapter configurations, *unit* is required with each flag used except `-h` and `-a`. The default for `-h` and `-a` is to display all adapters.

–a **[***unit***]**

Display all current ARP entries in the kernel table. If you do not specify a unit, the utility displays a line for each unit in the system.

–h **[***unit***]**

Display the ULA, logical address, and status information for the specified HIPPI unit. If you do not specify a unit, the utility displays a line for each unit in the system.

–c **[***unit***]**

Clear the ULA-to-logical-address-mapping table for the specified unit of non-reserved and non-permanent entries. You can delete permanent entries by using the `-d` option. This option requires super user privileges and returns an EBUSY error if the table is being updated.

–s *hostname ULA switch-address* **[***unit***]**

Create an ARP entry for the specified host with the specified ULA, the HIPPI logical-switch address (`–switch-address` and, optionally, the specified network unit. You must provide the *unit* option if more than one HIPPI unit exists in the system. The ULA is given as six hexadecimal characters separated by colons or dashes. The HIPPI switch address is given as three hexadecimal characters (for example, 0x3ef). If an ARP entry already exists for the specified host, the existing entry is updated with the new information. The entry is permanent unless you specify the `–temp` flag. This command requires super user privileges.

–d *hostname*

Delete the ARP entry if one exists for the specified host. This command requires super user privileges.

`–D` *logical-address –unit*

Delete the ARP entry if one exists for the specified logical address, as long as no IP address is assigned. This command requires super user privileges.

`–l` *logical-address* **[***unit***]**

Set the logical address of the adapter switch. The switch address is coded as described above. If the adapter discovers itself at a different logical address, the discovered address is used. This command requires super user privileges.

`–i` **[***unit***]**

Invalidate the logical address of the adapter. This command requires super user privileges.

SEE ALSO

arp(1M), ifconfig(1M), **`hippi`**(1M)

NAME

hippid – HIPPI support daemon

SYNOPSIS

hippd

hippd [–h *host_name*]

hippd [–k]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippid` is a system daemon. The process forks and the parent dies. `hippid` provides a process context for the following driver functions:

1. Provides a user context for the ARP Agent to broadcast ARP requests to known hosts. Each host that supports ARP resolution over HIPPI directs its ARP request messages to the HIPPI-SC logical address, `0xfe1`. The ARP agent receives the ARP request messages and forwards them to all known hosts. `hippid` is a replacement mechanism for Ethernet broadcast.
2. Provides a user context for the IP broadcast agent to broadcast IP packets to known hosts. Each host that supports IP broadcast over HIPPI directs its broadcast IP datagrams to the HIPPI-SC logical address, `0xfe1`. The broadcast agent receives the IP datagrams and forwards them to all known hosts. `hippid` is a replacement mechanism for Ethernet broadcast.
3. Provides a user context for self-discovery activity. The driver determines its own logical address (if any) and the logical addresses of possible remote HIPPI NICs.
4. Provides a user context for NIC watchdog processing. The driver uses a watchdog mechanism to make sure that the NIC is running properly. When a failure is discovered, a dump *file set* is generated by the dump daemon. The daemon makes an ioctl() call that sleeps in the kernel. To stop the daemon, execute the command `hippd -k`. Starting and stopping the daemon requires super-user privileges.

OPTIONS

The following options are supported:

–k	Kill the daemon.
–h *host_name*	Set the specified host name into the driver. This name is used by the startup scripts, since the host name has not been set. After the system is booted, `-h` is not needed.

SEE ALSO

hippiarp(1M), **hippi**(1M), **hippidisp**(1M), **hippidmpd**(1M)

NAME

hippidb – HIPPI driver debug trace display and control

SYNOPSIS

hippidb [–t *trace-level*] [–d *trace-level*] [–v *validataion-level*]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippidb` displays and controls the debug trace levels of the HIPPI driver. When executed with no options, `hippidb` displays the current status of type t tracing, type d tracing, and packet validation levels. `hippidb` with any of its options can be executed only by super user.

OPTIONS

The following options are supported:

–t *trace-level*

Trace the general operation of the driver at the specified level. The higher the trace level, the more noticeable the performance reduction. Levels are:

- 0 - Disable tracing (the default)
- 1 - Enable general tracing
- 2 - Enable extensive tracing
- 3 - Same as 2

–d *trace-level*

Trace error paths at the specified level. This option does not affect performance. Levels are:

- 0 - Disable tracing
- 1 - Enable tracing (the default)
- 2 - Stop the RunCode when RunCode discovers an error

At level 2, a dump is extracted and the RunCode is not automatically restarted. `hippi on` will restart the RunCode. See **hippi**(1M).

–v *validation-level* Validate the structure of received HIPPI packets at the specified level. This option is used on control testing in the network driver. You can use the **hippitd**(1M) utility to extract the trace buffer from the driver and format it into a text file. Validation levels are:

- 0 - Minimal validation (the default)
- 1 - Enable extensive validation (may not be available on all systems)

If the return status is -1, the validation code is not turned on in the driver and you cannot change this value.

SEE ALSO

hippi(1M), **hippiarp**(1M), **hippistat**(1M), **hippitb**(1M)

NAME

hippidispp – HIPPI ARP (address resolution) display and control

SYNOPSIS

hippidisp –D *unit* [*general-options*] [*device-options*]

hippidisp –f *filename* [*general-options*] [*device-options*]

hippidisp –f *filename* [*general-options*] [*program-options*]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippidisp` is a diagnostic utility that displays information retrieved from a NIC, either directly by this utility or previously by the dump daemon, `hippidmpd(1M)`. `hippidisp` displays the internal structure of RunCode program files. Much of the information displayed relates to the internal operation of the driver and RunCode. This man page does not attempt to describe the various reports in detail.

When executed with no options, `hippidisp` prints out a usage message.

The `-D` [*unit*] form of `hippidisp` extracts and displays information about the specified HIPPI device. To view a list of HIPPI devices installed on the system, you can execute `hippi version`. See `hippi(1M)`.

The `-f` *filename* form of the command processes the specified file, then displays the the desired records. Files contain RunCode images or NIC dump images.

OPTIONS

This command supports three types of options, which are described below. General options control the general operation of the utility. Device options, also called NIC Dump options, either directly access a NIC or display a NIC dump file. Program options display a RunCode file. When this command is used with its `-f` option, device options and program options are mutually exclusive.

General Options

–H	Display record headers.
–p	Display the generally used *partial* information from the records.
–F	Display all of the information from the records.
–x	Display the entire record in hex.

Device (NIC Dump) Options

–a	Display all NIC dump records.
–d	Display driver records. These records contain data structures that are used by the driver to manage the NIC.

`-r`	Display all ring records. The rings are the principle interface between the driver and the NIC. You can have up to 256 receive rings, a send ring, an event ring, and a command ring. A list of descriptors, if any, is printed for each ring. The receive ring number corresponds to the 8-bit ULP number in the incoming packet.
`-s`	Display statistics for the driver and NIC.
`-n`	Display the NIC registers.
`-l`	Display the NIC SRAM contents.
`-e` **[***unit***]**	Display the NIC EEPROM contents. The NIC must be halted (`hippi off`) for the EEPROM to be displayed. See **`hippi`**(1M).
`-m`	Display the manufacturing information area of the EEPROM. This area shows the part number and revision for various components of the board (for example, the ULA address, board serial number, and manufacturing data). The NIC must be halted (`hippi off`) for the EEPROM information to be displayed. See **`hippi`**(1M).
`-t`	Display the driver trace buffer. This form of `hippidisp` uses the same format as `hippitb(1M)`.
`-N`	Display the NIC trace buffer. This form of `hippidisp` uses the same format as `hippitb(1M)` with its `-n` option.

Program File Options

`-A`	Display all of the program file sections.
`-L`	Display all of the `LINE` records (that is, the objects that have several line number records). The source-level debugger uses the source code line number and corresponding SRAM address information.
`-S`	Display all of the symbol table records.
`-T`	Display all of the text sections (that is, `TXT1`, `TXT2`, and `TEXT` for Phase-1 text, Phase-2 text, and the RunCode text, respectively).

–P	Display all of the program counter records.
–V	Verify the checksum if it follows a text segment.

SEE ALSO

hippi(1M), **hippidmpd**(1M), **hippistat**(1M), **hippitb**(1M), **hippitune**(1M)

NAME

hippidmpd – HIPPI dump daemon

SYNOPSIS

hippidmpd [–a] [–d *dump_dir*]

hippidmpd [–k]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippidmpd` is a system daemon. The process forks and the parent dies. When the user requests generation of a dump file by issuing a `hippi off dump` or `hippi restart dump` command, and when the watchdog discovers that the NIC is not operating properly, the `hippidmpd` daemon extracts the relevant information and produces a dump file as described below. See **`hippi`**(1M).

Starting and stopping the daemon requires super-user privileges

By default, the dump files are placed in `/var/hippi`.

Wnen the first dump file is generated, a *Bounds* file is created. The Bounds file is an ASCII file that contains one line for each defined HIPPI card that has been dumped into the target directory. The fields in the lines are decimal numbers separated by a space. The line is terminated by a newline character. Each line contains the following fields:

- `card number` - The number of the card that is being dumped
- `set number` - The set number of the most recently created set
- `current files` - The number of sets for the card
- `max sets` - The maximum number of sets allowed for the card (the default is 5)

As dump requests are received, the daemon produces the requested file. When the maximum number of files is reached, the lowest-numbered file for the card is deleted to make room for the new file.

By default, the maximum number of files is five. You can change that number by editing the Bounds file. You can delete any file, including the Bounds file, at any time.

OPTIONS

The following options are supported:

–a
: Produce an ASCII dump file instead of the default binary file. Use this option carefully, as the ASCII dump file can be quite large, and it does not contain as much information.

–d *dump_directory*	Place dump files in the specified directory. The directory must already exist and root must be able to create and update files there.
–k	Kill the daemon process.

SEE ALSO **hippi**(1M), **hippidisp**(1M)

NAME

hippidnld – HIPPI driver RunCode download utility

SYNOPSIS

hippidnld [–d]

hippidnld [–c]

hippidnld [–D *unit*]

hippidnld [–l *file*]

hippidnld [–e *file*]

hippidnld [–r *file*]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippidnld` manages the RunCode download to the NIC. The RunCode can be located in the EEPROM on the NIC or as a cached image in the memory space of the driver.

`hippidnld` lets you use an alternative version of the RunCode for one session, or program it into the EEPROM for regular use. When the NIC is reset, it loads the cached image from the driver, if one is available. Otherwise, it loads an image from the EEPROM.

If no RunCode is available, `hippidnld` fails.

OPTIONS

The following options are supported:

–D *unit* — In multi-card configurations, use the specified card on which `hippidnld` is to perform download operations. *unit* is expressed as `hip` *n*, where *n* is an integer 0 to 3, inclusive. To view a list HIPPI devices installed on the system, you can execute `hippi version`. See **`hippi`**(1M).

–d — Delete the RunCode in the driver cache.

–l *file* — Download the hex-format RunCode to the driver cache. You can use this option while the NIC is operational. The new RunCode goes into effect when the NIC is restarted.

–e *file* — Load a full RunCode image from a hex-formatted file into the EEPROM and preserve the existing serial number and ULA (Universal LAN Address) values. This option does not alter the driver cache. The target NIC must be halted prior to this operation.

–r *file*	Download only a RunCode image from a hex-formatted file into the EEPROM manufacturing and header information. It does not alter the driver cache. The target NIC must be halted prior to this operation.
–c	Clear only the RunCode form the EEPROM. This option preserves manufacturing and tuning data.

SEE ALSO **hippi**(1M), **hippistat**(1M)

NAME

hippistat – HIPPI hardware statistics

SYNOPSIS

hippistat [–D *unit*]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippistat` hardware statistics for a HIPPI device. To obtain network statistics, use `netstat(1M)`.

OPTIONS

The following options are supported:

–D *unit* Display statistics about the specified HIPPI card. *unit* is expressed as `hip`*n*, where *n* is an interger 0 to 3, inclusive. To view a list of HIPPI devices installed on the system, you can execute `hippi version`. See **`hippi`**(1M). If `-D` is not used, `hippistat` displays information about `hip0`.

SEE ALSO

`hippi`(1M), netstat(1M)

NAME

hippitb – HIPPI driver debug trace display

SYNOPSIS

hippitb [–n [–D *unit*]]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippitb` displays a formatted version of the driver and RunCode debug trace buffers. It supports driver maintenance. The format of the report depends on the version of the driver and is not detailed here.

OPTIONS

The following options are supported:

`–n` Display a formatted version of the NIC RunCode debug trace buffer.

`–D` *unit* Display information about the specified HIPPI device, expressed as `hip` *n*, where *n* is an interger 0 to 3, inclusive. To view a list of HIPPI devices installed on the system, you can execute `hippi version`. See `hippi(1M)`. The unit is required only on multiple-adaptor configurations. A single driver trace buffer is shared by all NICs. `-D` is used only with `-n`.

SEE ALSO

hippi(1M), hippiarp(1M), hippidb(1M), hippistat(1M)

NAME

hippitune – HIPPI driver RunCode download utility

SYNOPSIS

hippitune [–l] [–p] [–e] [–c *retry_count*] [–t *retry_timer*] [–o *campon_timeout*] [–s *stat_timer*] [–i *interrupt_timer*] [–x *tx_idle*] [–r *rx_idle*] [–w *dma_write_state*] [–d *dma_read_state*] [–h *pci_state_req*] [–D *unit*]

AVAILABILITY

SUNWhip

DESCRIPTION

`/etc/opt/SUNWconn/bin/hippitune` provides access to the registers in a HIPPI device that are used for tuning performance and controlling device operation.

Default values for the registers are cached in the driver so they can be set each time the NIC RunCode is started. Specifically, they are stored in the EEPROM on the NIC so that the driver cache can be set at system boot.

If you change the counters and timers, the RunCode operation changes immediately. Changes to the state registers become effective when RunCode is restarted. Updates to the EEPROM (`-e`) do not become effective until the next system boot. The EEPROM can be accessed only while RunCode is off.

You can combine the options in a single command. If the system has more than NIC, you must specify the NIC interface (*unit*). See `-D`, below.

`hippitune` can display the current tuning values (`-p`), and the tuning values stored in the EEPROM (`-p` with `-e`). Only super user can change the currect values and EEPROM values.

Time values are specified in 0.97u-sec units.

OPTIONS

The following options are supported:

–c *retry_count*

Retry a rejected connection the specified number of times before aborting when the HIPPI-SC Campon bit is not set in the I-field. *retry_count* is an integer of zero or more.

–t *retry_timer*

When `-c` is specified with an integer greater than one, wait the specified number of seconds between each retry. *retry_timer* is an integer.

–o *campon_timeout*

When the HIPPI-SC Campon bit is set in the I-field, make the adapter wait the specified number of seconds for the connection to be accepted. If the

connection has not been accepted after this amount of time, consider the connection to be rejected. *campon_timeout* is an integer.

`–s` *stat_timer*

Place a new snapshot of operating statistics in host memory each *stat_timer* seconds, where *stat_timer* is an integer. If *stat_timer* is set to zero, the statistics are not automatically copied to host memory.

`–i` *interrupt_timer*

Separate back-to-back interrupts by the specified number of seconds. This option lets you prevent the adapter from generating interrupts faster than the host system can handle them. Use this option carefully; short times tend to flood the host with interrupts and long times tend to reduce responsiveness of the device. *interrupt_timer* is expressed in integers.

`–x` *tx_idle*

Use the specified timeout period for idle connections. If a transmit connection has not passed any data for a period of *tx_idle* seconds, where *tx_idle* is expressed as an integer, the connection is aborted.

`–r` *rx_idle*

Use the specified receive timeout period for idle connections. If a receive connection has not passed any data for a period of *rx_idle* seconds, where *rx_idle* is expressed as an integer, the connection is aborted.

`–w` *dma_write_state*

See the Roadrunner specification for bit settings.

`–d` *dma_read_state*

See the Roadrunner specification for bit settings.

`–h` *pci_state_reg*

See the Roadrunner specification for bit settings.

`–l –e`

When used together these options place the current HIPPI flags into the EEPROM.

–p

When used without `-e`, `hippitune -p` displays the current values of the tuning parameters contained within the driver. When used with `-e`, it displays the current values of the tuning parameters in the EEPROM.

–e

Display and modify values in the EEPROM. The NIC must be turned off. This option is a modifier for the other options. See EXAMPLES, below.

–D *unit*

Access information about the specified HIPPI device, expressed as `hip`*n*, where *n* is an integer 0 to 3, inclusive. To view a list of HIPPI devices installed on the system, you can execute `hippi version`. See `hippi(1M)`. If you have more than one NIC configured, you must specify this option.

EXAMPLES

EXAMPLE 1 Changing `retry_count` In the Driver Cache

The following command changes `retry_count` in the driver cache.

```
# hippitune -c 0x1234
```

EXAMPLE 2 Changing the EEPROM Values

The following command writes the `retry_count` into the EEPROM.

```
# hippitune -c 0x1234 -e
```

EXAMPLE 3 Displaying EEPROM Values

The following command displays the values currently written into the EEPROM.

```
# hippitune -p -e
```

SEE ALSO

hippi(1M)

NAME

sink – HIPPI character device receiver example

SYNOPSIS

sink [–2] [–D *unit*] [–l *size*] [–n *number-of-reads*] [–u *ULP*] [–c *checking-level*] [–s] [–v]

AVAILABILITY

SUNWhip

DESCRIPTION

The /etc/opt/SUNWconn/bin/sink program provides sample code for testing and using a HIPPI character device. The code includes most of the ioctl() settings for receiving data. (For more information about these ioctl() settings, see the character device interface user's guide and reference document.) You can use this sample code with blast(1M) to analyze system performance.

OPTIONS

All of the Arguments are optional and default values are provided.

–2

Cause sink to run in double-threaded mode. In this mode, two processes are bound to the same ULP at the same time.

–s

Bink sink to the ULP using a shared bind. The default is exclusive bind.

–D *unit*

Provide sample code for testing the specified HIPPI device, expressed as hip*n*, where *n* is an integer 0 to 3, inclusive. To view a list of HIPPI devices installed on the system, you can execute hippi version. See hippi(1M). If this option is not specified, sink acts for the first HIPPI card displayed by hippi version.

–l *size*

Use the specified size, expressed in bytes, for the buffer passed to each read() call. sink accepts packets of any length. *size* indicates the number of bytes received at one time, independent of packet size. The default is 4104 bytes, as determined by adding eight bytes for the HIPPI-FP header to the 4096-byte default buffer size of the blast(1M) sample program.

–n *number-of-reads*

Perform the specified number of reads before exiting. sink continues to execute until it successfully completes this number of reads or encounters an error while in checking mode. The default is to continue reading indefinitely.

–u *ULP*

Use the specified upper layer protocol identifier for the framing protocol header. *ULP* must match the upper layer protocol identifier specified by `blast(1M)`. The default is 0x82.

–c *checking-level*

Use the specified checking level for `sink`. If the level is 1, only the first data error is flagged; if 2, all data errors are flagged. The default is no checking.

–v

Generate verbose output.

SEE ALSO

blast(1M), hippi(1M)