Platform Notes: The hme FastEthernet Device Driver

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

Part No.: 805-4449
September 1998, Revision A

Send comments about this document to:
docfeedback@sun.com

Please
Recycle

Adobe PostScript™

Contents

Tables

CHAPTER 1

Introduction to the hme Fast Ethernet Device Driver

This book describes how to configure the hme driver for the SBus or PCI-bus based Sun™ Ultra™ workstations, Sun Enterprise™ servers, the SunSwift™ SBus Adapter, the SunFastEthernet™ Adapter 2.0, and the SunFastEthernet PCI Adapter. To configure the hme driver you can use the ndd utility to set the parameters. If you use the ndd utility, the parameter values are effective until you reboot the system. To keep the values in effect even after rebooting the system, enter the parameter values into the /etc/system file or the hme.conf file in the /kernel/drv directory.

Chapter 2 describes the hme device driver and includes topics such as operating speeds and modes, auto-negotiation, the internal transceiver, and the external transceiver. Chapter 3 describes the parameters and settings for the hme device driver. Chapter 4 describes how to set the hme device driver parameter values using the ndd utility and also in the /etc/system and hme.conf files.

With the introduction of Solaris 7, Sun Microsystems now supports a 64-bit version of the Solaris operating environment. With the new 64-bit version, there is a new directory /kernel/drv/sparcv9. In this document and others, note that when kernel/drv is mentioned, the /kernel/drv/sparcv9 directory also applies.

Related Documentation

The following related books provide an overview of the hme driver and the FEPS ASIC, give more information on the Ethernet standard, and list man pages for the hme driver.

- *Solaris 7 Sun Hardware Platform Guide:* Provides an overview of the hme driver and the Fast Ethernet Parallel Port SCSI (FEPS) ASIC

- *IEEE 802.3u Ethernet Standard:* Provides additional information about the Ethernet standard
- *Solaris on Sun Hardware Reference Manual Supplement:* Explains the following man pages:
    - The `ndd (1M)` man page summarizes information on how to use the `ndd` utility
    - The `prtconf` (1M), `system` (4), and `driver.conf` (4) man pages describe information about entering parameter values into the `/etc/system` file and the `hme.conf` file

Typographic Conventions

TABLE 1-1 describes the typographic conventions used in this book.

TABLE 1-1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, or words to be emphasized. Command-line variable; replace with a real name or value	Read Chapter 6 in the *User's Guide.* These are called *class* options. You *must* be `root` to do this. To delete a file, type `rm` *filename.*

Shell Prompts

TABLE 1-2 shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE 1-2 Shell Prompts

Shell	Prompt
C shell	*machine_name*%
C shell superuser	*machine_name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Sun Documentation on the Web

The `docs.sun.com` web site enables you to access Sun technical documentation on the Web. You can browse the `docs.sun.com` archive or search for a specific book title or subject at:

`http://docs.sun.com`

Sun Welcomes Your Comments

We are interested in improving our documentation and welcome your comments and suggestions. You can email your comments to us at.

`docfeedback@sun.com`

Please include the part number of your document in the subject line of your email.

CHAPTER 2

The hme Device Driver

The hme device driver handles the SUNW,hme device on these hardware devices:

- SunSwift SBus Adapter
- SunSwift PCI Adapter
- SunFastEthernet Adapter 2.0
- SunFastEthernet PCI Adapter
- Sun Ultra systems and Sun Enterprise servers

This chapter gives a hardware overview of the SUNW,hme device, provides information on the operating speeds and modes for the SUNW,hme device, and discusses auto-negotiation, the internal transceiver, and the external transceiver for the hme device driver. Note that the external transceiver is not present on the SunSwift SBus Adapter or the SunSwift PCI Adapter.

Hardware Overview

The SUNW,hme device provides 10BASE-TX or 100BASE-T networking interfaces using the Fast Ethernet Parallel Port SCSI (FEPS) ASIC and an internal transceiver. The driver automatically sets the link speed to 10 or 100 Mbps and conforms to the 100BASE-T IEEE 802.3u Ethernet standard. The FEPS (SBus based) or PFEX (PCI-bus based) ASIC provides the SBus or PCI interface and Media Access Control (MAC) functions. The internal transceiver, which connects to an RJ-45 connector on all of the above hardware devices, provides the physical layer functions.

In addition to the RJ-45 connector, a Media Independent Interface (MII) connector, which is an Ultra DB 40 connector, is also provided on Sun systems. The MII connects to an external transceiver that may use any physical media, such as copper or fiber as specified in the 100BASE-TX standard. When an external transceiver is connected to the MII, the driver selects the external transceiver and disables the internal transceiver. The external transceiver may also support the 100BASE-T4 standard, which allows the link to operate in 100 Mbps speed using four pairs of category 3 or better cable.

Operating Speeds and Modes

You can operate the link in any of the following speeds and modes with the `SUNW, hme` device:

- 100 Mbps, full-duplex
- 100 BASE-T4 (with external transceiver only)
- 100 Mbps, half-duplex
- 10 Mbps, full-duplex
- 10 Mbps, half-duplex

The 100BASE-T standard, *IEEE 802.3u Ethernet Standard*, describes these speeds and modes.

Auto-Negotiation

The *auto-negotiation* protocol, as specified by the 100BASE-T standard, selects the operation mode (half-duplex or full-duplex) at boot time or when the link state changes (the link goes down or tried to come up). The auto-negotiation protocol also selects the speed and the full duplex or half duplex mode.

The auto-negotiation protocol does the following:

- Identifies all link partner-supported modes of operation
- Advertises its capabilities to the link partner
- Selects the highest common denominator mode of operation based on the following priorities:
  - 100 BASE-T4
  - 100 Mbps, full-duplex
  - 100 Mbps, half-duplex
  - 10 Mbps, full-duplex
  - 10 Mbps, half-duplex

The link partner is the networking device (system, Ethernet hub, or Ethernet switch) at the other end of the link or cable. If the `SUNW,hme` device is connected to a remote system or interface that is not capable of auto-negotiation, the system automatically selects the correct speed and half-duplex mode.

If adapters or systems are connected to a link partner and the auto-negotiation protocol fails to operate successfully, you can configure the device so it does not use this protocol. This forces the driver to set up the link in the mode and speed that you choose instead of using the auto-negotiation protocol.

Internal Transceiver

The internal transceiver is a feature supported by the driver and is capable of all the operating speeds and modes (except the 100BASE-T4 mode) listed in the section "Operating Speeds and Modes," earlier in this chapter. When the internal transceiver is used, the default is auto-negotation by the `hme` driver, which automatically selects the speed and mode of the link. The internal transceiver performs auto-negotiation with the remote end of the link (link partner) to select a common mode of operation.

The internal transceiver also supports a forced mode of operation. This is where the user selects the speed and mode using the `ndd` utility, the `/etc/system` file, or the `hme.conf` file. The `ndd` utility makes calls to the `hme` driver to choose the speed and mode.

External Transceiver

When an external transceiver (not present on the SunSwift SBus Adapter) is connected to the MII interface, the driver selects the external transceiver for networking operations.

- If the external transceiver supports auto-negotiation, the driver uses the auto-negotiation feature to select the link speed and mode.
- If the external transceiver does not support auto-negotiation, the driver selects the highest priority mode supported by the transceiver.

You can also manually select the speed and mode of the link. For example, two transceivers might not support the same mode and speed. Therefore, you must select the *highest* mode and speed that both transceivers support using the `ndd` utility. See the list of operating speeds and modes in the section, "Operating Speeds and Modes," in this chapter.

CHAPTER 3

Parameter Definitions

This chapter describes the parameters and settings for the hme device driver.

Driver Parameter Values and Definitions

The following sections describe the hme driver parameters, which are listed in TABLE 3-1.

TABLE 3-1 hme Driver Parameter, Status, and Descriptions

Parameter	Status	Description
transceiver_inuse	Read only	Defines the current status
link_status	Read only	Defines the current status
link_speed	Read only	Defines the current status
link_mode	Read only	Defines the current status
ipg1	Read and write	Inter-packet gap parameter
ipg2	Read and write	Inter-packet gap parameter
use_int_xcvr	Read and write	Operational mode parameter
pace_size	Read and write	Operational mode parameter
adv_autoneg_cap	Read and write	Operational mode parameter
adv_100T4_cap	Read and write	Operational mode parameter
adv_100fdx_cap	Read and write	Operational mode parameter
adv_100hdx_cap	Read and write	Operational mode parameter
adv_10fdx_cap	Read and write	Operational mode parameter

TABLE 3-1 `hme` Driver Parameter, Status, and Descriptions *(Continued)*

Parameter	Status	Description
`adv_10hdx_cap`	Read and write	Operational mode parameter
`autoneg_cap`	Read only	Local transceiver auto negotiation capability
`100T4_cap`	Read only	Local transceiver capability of the hardware
`100fdx_cap`	Read only	Local transceiver capability of the hardware
`100hdx_cap`	Read only	Local transceiver capability of the hardware
`10fdx_cap`	Read only	Local transceiver capability of the hardware
`10hdx_cap`	Read only	Local transceiver capability of the hardware
`lp_autoneg_cap`	Read only	Link partner auto negotiation capability
`lp_100T4_cap`	Read only	Link partner capability
`lp_100fdx_cap`	Read only	Link partner capability
`lp_100hdx_cap`	Read only	Link partner capability
`lp_10fdx_cap`	Read only	Link partner capability
`lp_10hdx_cap`	Read only	Link partner capability
`instance`	Read and write	Device instance
`lance_mode`	Read and write	Additional delay before transmitting a packet
`ipg0`	Read and write	Additional delay before transmitting a packet

Defining the Current Status

The read-only parameters described in TABLE 3-2 explain the operational mode of the interface. These parameters define the current status.

TABLE 3-2 Read-Only Parameters Defining the Current Status

Parameter	Values	Description
`transceiver_inuse`	0 1	= Internal transceiver = External transceiver
`link_status`	0 1	Current link status = Link down = Link up
`link_speed`	0 1	Valid only if the link is up = 10 Mbps = 100 Mbps
`link_mode`	0 1	Valid only if the link is up = Half duplex = Full duplex

Inter-Packet Gap Parameters

The Fast Ethernet Parallel Port SCSI (FEPS) ASIC supports programmable Inter-Packet Gap (IPG) parameters `ipg1` and `ipg2`. The total IPG is the sum of `ipg1` and `ipg2`. The total IPG is 9.6 microseconds when the link speed set by the auto-negotiation protocol is 10 Mbps. When the link speed is 100 Mbps, the total IPG is 0.96 microseconds.

TABLE 3-3 lists the default values and allowable values for the inter-packet gap (IPG) parameters, `ipg1` and `ipg2`.

TABLE 3-3 Read-Write Inter-Packet Gap Parameter Values and Descriptions

Parameter	Values (Byte-time)	Description
`ipg1`	0, 255	`ipg1` = 8 (default at initialization)
`ipg2`	0, 255	`ipg2` = 4 (default at initialization)

By default, the driver sets `ipg1` to 8-byte time and `ipg2` to 4-byte time, which are the standard values. (Byte time is the time it takes to transmit one byte on the link, with a link speed of either 100 Mbps or 10 Mbps.)

If your network has systems that use longer IPG (the sum of ipg1 and ipg2) and if those machines seem to be slow in accessing the network, increase the values of ipg1 and ipg2 to match the longer IPGs of other machines.

Defining an Additional Delay Before Transmitting a Packet Using lance_mode and ipg0

Two ASICs:

- Fast Ethernet Parallel Port SCSI (FEPS) for SBus)
- PCI Fast Ethernet and Expansion (PFEX for PCI)

support a programmable mode called lance_mode. The ipg0 parameter is associated with lance_mode.

After a packet is received with lance_mode enabled (default) an additional delay is added by setting the ipg0 parameter before transmitting the packet. This delay, set by the ipg0 parameter, is in addition to the delay set by the ipg1 and ipg2 parameters. The additional delay set by ipg0 helps to reduce collisions. Systems that have lance_mode enabled might not have enough time on the network.

If lance_mode is disabled, the value of ipg0 is ignored and no additional delay is set. Only the delays set by ipg1 and ipg2 are used. Disable lance_mode if other systems keep sending a large number of back-to-back packets.

You can set the additional delay with the ipg0 parameter from 0 to 31, which is the nibble time delay. Note that nibble time is the time it takes to transfer four bits on the link. If the link speed is 10 Mbps, nibble time is equal to 400 ns. If the link speed is 100 Mbps, nibble time is equal to 40 ns.

For example, if the link speed is 10 Mbps, and you set ipg0 to 20 nibble times, multiply 20 by 400 ns to get 800 ns. If the link speed is 100 Mbps, and you set ipg0 to 30 nibble-times, multiply 30 by 40 ns to get 120 ns.

TABLE 3-2 defines the lance_mode and ipg0 parameters.

TABLE 3-4 Parameters Defining lance_mode and ipg0

Parameter	Values	Description
lance_mode	0 1	lance_mode disabledlance_mode enabled (default)
ipg0	0-31[1]	Additional IPG before transmitting a packet (after receiving a packet)

[1] The default value is 16 nibble-times, which is 6.4 microseconds for 10 Mbps and 0.64 microseconds for 100 Mbps.

Operational Mode Parameters

TABLE 3-5 describes the operational mode parameters and their default values.

TABLE 3-5 Operational Mode Parameters

Parameter	Values	Description
`adv_autoneg_cap`	0 1	Local transceiver capability advertised by the hardware = Forced mode = Auto-negotiation (default)
`adv_100T4_cap`[1]	0 1	Local transceiver capability advertised by the hardware; read/write parameter = Not 100BASE-T4 capable = 100BASE-T4 capable
`adv_100fdx_cap`[1]	0 1	Local transceiver capability advertised by the hardware; read/write parameter = Not 100Mbit/sec full-duplex capable (default in the Solaris 2.5 and 2.5.1 software enviroments) = 100Mbit/sec full-duplex capable (default in the Solaris 2.6 software environment)
`adv_100hdx_cap`[1]	0 1	Local transceiver capability advertised by the hardware; read/write parameter = Not 100Mbit/sec half-duplex capable = 100Mbit/sec half-duplex capable (default)
`adv_10fdx_cap`[1]	0 1	Local transceiver capability advertised by the hardware; read/write parameter = Not 10Mbit/sec full-duplex capable (default) = 10Mbit/sec full-duplex capable
`adv_10hdx_cap`[1]	0 1	Local transceiver capability advertised by the hardware; read/write parameter = Not 10Mbit/sec half-duplex capable = 10Mbit/sec half-duplex capable (default)
`use_int_xcvr`	0 1	Local transceiver capability selected for networking by the user = External transceiver is used if connected (default) = Internal transceiver is used, even if the external transceiver is connected

[1] The priority (in descending order) for these parameters is: `adv_100fdx_cap,adv_100T4_cap,adv_100hdx_cap, adv_10fdx_cap,` and `adv_10hdx_cap.`

Selecting the Internal or External Transceiver

Use the use_int_xcvr parameter to change the default. For example, you can select the internal transceiver, even though the driver is capable of selecting the external transceiver. The driver, not the hardware, determines the internal and external transceiver.

If the external transceiver, which is connected to the MII interface, is present, the driver selects the external transceiver using the default use_int_xcvr 0. If the external transceiver is not present, the driver selects the internal transceiver. When this parameter is set to use_int_xcvr 1, the driver selects the internal transceiver, even if the external transceiver is connected.

Defining the Number of Back-to-Back Packets to Transmit

The pace_size parameter (see TABLE 3-6) defines the maximum number of back-to-back packets you can transmit at one time. If the value is zero, there is no limit to the number of back-to-back packets that can be transmitted.

TABLE 3-6 Back-to-Back Packet Transmission Capability

Parameter	Values	Description
pace_size	1 to 255	= Number of back-to-back packets transmitted at one time
	0	= No limit to the number of back-to-back packets that can be transmitted (default)

Reporting Transceiver Capabilities

TABLE 3-7 describes the read-only transceiver capabilities (either the internal transceiver or the external transceiver), whichever is selected.

TABLE 3-7 Read-Only Transceiver Capabilities

Parameter	Values	Description
`autoneg_cap`	0 1	Local transceiver capability of the hardware = Not capable of auto-negotiation = Auto negotiation capable
`100T4_cap`	0 1	Local external transceiver capability of the hardware = Not 100BASE-T4 capable = 100BASE-T4 capable
`100fdx_cap`	0 1	Local transceiver capability of the hardware; initialized at startup = Not 100Mbit/sec full-duplex capable = 100Mbit/sec full-duplex capable
`100hdx_cap`	0 1	Local transceiver capability of the hardware; initialized at startup = Not 100Mbit/sec half-duplex capable = 100Mbit/sec half-duplex capable
`10fdx_cap`	0 1	Local transceiver capability of the hardware; initialized at startup = Not 10Mbit/sec full-duplex capable = 10Mbit/sec full-duplex capable
`10hdx_cap`	0 1	Local transceiver capability of the hardware; initialized at startup = Not 10Mbit/sec half-duplex capable = 10Mbit/sec half-duplex capable

The parameters in TABLE 3-7 define the capabilities of the hardware. The internal transceiver can support all of these capabilities. The capabilities of the external transceiver are dependent on the device. If the external transceiver is not capable of auto-negotiation but has the capability of all speeds and modes (100 Mbps, 10 Mbps, half-duplex, and full-duplex), you must force the operational speed and mode of the external transceiver.

Reporting the Link Partner Capabilities

TABLE 3-8 describes the read-only link partner capabilities.

TABLE 3-8 Read-Only Link Partner Capabilities

Parameter	Values	Description
`lp_autoneg_cap`	0 1	= No auto-negotiation = Auto-negotiation
`lp_100T4_cap`	0 1	= No 100BASE-T4 = 100BASE-T4
`lp_100fdx_cap`	0 1	= No100Mbit/sec full-duplex transmission = 100Mbit/sec full-duplex
`lp_100hdx_cap`	0 1	= No 100Mbit/sec half-duplex transmission = 100Mbit/sec half-duplex
`lp_10fdx_cap`	0 1	= No 10Mbit/sec full-duplex transmission = 10Mbit/sec full-duplex
`lp_10hdx_cap`	0 1	= No 10Mbit/sec half-duplex transmission = 10Mbit/sec half-duplex

If the link partner is not capable of auto-negotiation (when `lp_autoneg_cap` is 0) the information described in TABLE 3-8 is not relevant and the parameter value = 0.

If the link partner is capable of auto-negotation (when `lp_autoneg_cap` is 1) then the speed and mode information is displayed when you use auto-negotiation and get the link partner capabilities.

CHAPTER **4**

Setting Parameters

This chapter describes how to configure the hme driver parameters using the ndd utility in the /etc/system file, or in the hme.conf file. Use the ndd utility to configure parameters that are valid until you reboot the system.

To configure the hme driver parameters for all devices in the system so that the parameter values are always in effect (even after rebooting the system), enter the parameter values in the /etc/system file. When the system is rebooted, it reads the /etc/system file and sets the parameter values in that file.

To set the parameters for a particular device in the system, set the parameters in the hme.conf file in the /kernel/drv directory. The parameters set in the hme.conf file have precedence over the parameters set in the /etc/system file and override the parameters set in the /etc/system file. Setting hme.conf parameter values are always in effect (even after rebooting the system).

Parameter Options

You can set the hme device driver parameters in three ways (ndd, /etc/system, and hme.conf), depending on your needs. To set parameters that are valid until you reboot the system, use the ndd utility. Using ndd is a good way to test parameter settings.

To set parameters so they remain in effect after you reboot the system:

- Add the parameter values to /etc/system when you want to configure parameters for all devices in the system.
- Create the hme.conf file and add parameter values to hme.conf when you need to set a particular parameter for a device in the system.

If you want to test parameter settings, use the `ndd` utility described in Chapter 3. With `ndd`, the parameters are effective until you reboot the system. To make the parameter settings permanent, enter the values in `/etc/system` or `hme.conf` as described in this chapter.

Setting Parameters Using `ndd`

Use the `ndd` utility to configure parameters that are valid until you reboot the system. The `ndd` utility supports any networking driver, which implements the Data Link Provider Interface (DLPI).

The following sections describe how you can use the `hme` driver and the `ndd` utility to modify (with the `-set` option) or display (without the `-set` option) the parameters for each `SUNW,hme` device.

Identifying Device Instances

Before you use the `ndd` utility to get or set a parameter for the `hme` device, you must specify the device instance for the utility if there is more than one `SUNW,hme` device.

Note – If there is only one `SUNW,hme` device, the device is automatically chosen by the `ndd` utility.

▼ To Specify the Device Instance for the `ndd` Utility

1. **Check the** `/etc/path_to_inst` **file to identify the instance associated with a particular device.**
2. **Use that instance number to select the device as follows:**

```
% ndd -set /dev/hme instance instance#
```

The device remains selected until you change the selection.

Non-Interactive and Interactive Modes

You can use the ndd utility in two modes:

- Non-interactive
- Interactive

In non-interactive mode, you invoke the utility to execute a specific command. Once the command is executed, you exit the utility. In interactive mode, you can use the utility to get or set more than one parameter value. (Refer to the ndd (1M) man page for more information.)

Using the ndd Utility in Non-Interactive Mode

This section describes how to modify a parameter value and how to display a parameter value.

- **To modify a parameter value, use the -set option.**

If you invoke the ndd utility with the -set option, the utility passes *value,* which must be specified down to the named /dev/hme driver instance, and assigns it to the parameter:

```
% ndd -set /dev/hme parameter value
```

- **To display the value of a parameter, specify the parameter name (and omit the value).**

When you omit the -set option, a query operation is assumed and the utility queries the named driver instance, retrieves the value associated with the specified parameter, and prints it:

```
% ndd /dev/hme parameter
```

Using the ndd Utility in Interactive Mode

- **To modify a parameter value in interactive mode, specify ndd hme, as shown below.**

The ndd utility then prompts you for the name of the parameter:

```
% ndd /dev/hme
name to get/set? (Enter the parameter name or ? to view all parameters)
```

After entering the parameter name, the ndd utility prompts you for the parameter value (see Table 4-1 through Table 4-8).

- **To list all the parameters supported by the** hme **driver, type** ndd /dev/hme \?**.**

  (See Table 4-1 through Table 4-8 for parameter descriptions.)

CODE EXAMPLE 4-1 Example of Listing All Parameters Supported by the hme Driver

```
example# ndd /dev/hme \?
?                             (read only)
transceiver_inuse             (read only)
link_status                   (read only)
link_speed                    (read only)
link_mode                     (read only)
ipg1                          (read and write)
ipg2                          (read and write)
use_int_xcvr                  (read and write)
pace_size                     (read and write)
adv_autoneg_cap               (read and write)
adv_100T4_cap                 (read and write)
adv_100fdx_cap                (read and write)
adv_100hdx_cap                (read and write)
adv_10fdx_cap                 (read and write)
adv_10hdx_cap                 (read and write)
autoneg_cap                   (read only)
100T4_cap                     (read only)
100fdx_cap                    (read only)
100hdx_cap                    (read only)
10fdx_cap                     (read only)
10hdx_cap                     (read only)
lp_autoneg_cap                (read only)
lp_100T4_cap                  (read only)
lp_100fdx_cap                 (read only)
lp_100hdx_cap                 (read only)
lp_10fdx_cap                  (read only)
lp_10hdx_cap                  (read only)
instance                      (read and write)
lance_mode                    (read and write)
ipg0                          (read and write)
example#
```

Example of Listing All Parameters Supported by the hme Driver

Setting Forced Mode

The procedure that follows describes how to set forced mode (not capable of auto-negotiation).

▼ To Select One Local Transceiver Capability and Setting Forced Mode

1. **Select one of the following capabilities:** `adv_100T4_cap, adv_100fdx_cap`**,** `adv_100hdx_cap`**,** `adv_10fdx_cap`**, or** `adv_10hdx_cap`**, and set its value to** `1`**.**

   If you select more than one of the local transceiver capabilities, the driver selects the one that is highest in the priority order.

2. **Set the local transceiver capabilities advertised by the hardware to forced mode = 0, which is not capable of auto-negotiation:** `adv_autoneg_cap 0`

   Use the `ndd` utility as described in "Using the ndd Utility in Interactive Mode in this chapter.

Auto-Negotiation Mode

The procedure that follows describes how to select at least one of the five local transceiver capabilities and set the mode to auto-negotiation.

▼ To Set the Mode to Auto-Negotiation

1. **Select** *at least one* **of the five capabilities (**`adv_100T4_cap`**,** `adv_100fdx_cap`**,** `adv_100hdx_cap`**,** `adv_10fdx_cap`**,** `adv_10hdx_cap`**) that you want to advertise to the remote system, and set its value to 1.**

2. **Set the local transceiver capabilities advertised by the hardware to 1, the auto-negotiation setting:** `adv_autoneg_cap 1`

   Use the `ndd` utility as described in "Using the ndd Utility in Interactive Mode in this chapter.

Configuring TCP/IP for Maximum Performance

This section describes how to benchmark the TCP/IP throughput and how to set the TCP/IP hiwater marks for best performance.

● **To benchmark the TCP/IP throughput, use the** `ndd` **command to configure the values of some of the TCP parameters for maximum performance.**

- **To set the TCP hiwater marks for maximum performance, type the following** ndd **commands:**

```
# ndd -set /dev/tcp tcp_xmit_hiwat 65535
# ndd -set /dev/tcp tcp_recv_hiwat 65535
# ndd -set /dev/tcp tcp_cwnd_max 65534
```

Setting Parameters in the /etc/system File

To configure the hme driver parameters for all SUNW,hme devices in the system so that the parameter variables are always effective (even after rebooting the system), enter the parameter variables in the /etc/system file. When you reboot the system, the system reads the /etc/system file and sets these parameter variables in the hme module in the operating system kernel.

TABLE 4-1 lists the variables you need to set in the /etc/system file.

TABLE 4-1 Setting Variables in the /etc/system File

Parameter	Variable
ipg1	hme_ipg1
ipg2	hme_ipg2
use_int_xcvr	hme_use_int_xcvr
pace_size	hme_pace_size
adv_autoneg_cap	hme_adv_autoneg_cap
adv_100T4_cap	hme_adv_100T4_cap
adv_100fdx_cap	hme_adv_100fdx_cap
adv_100hdx_cap	hme_adv_100hdx_cap
adv_10fdx_cap	hme_adv_10fdx_cap
adv_10hdx_cap	hme_adv_10hdx_cap
lance_mode	hme_lance_mode
ipg0	hme_ipg0

These parameter values, described in Chapter 3, are applicable to all `SUNW,hme` devices on the system. See TABLE 3-2 through TABLE 3-8 for parameter descriptions. An example follows.

▼ To Set the `ipg1` to 10 and `ipg2` to 5 When Rebooting

1. **Become superuser.**
2. **Add the following lines to the** `/etc/system` **file:**

```
set hme:hme_ipg1 = 10
set hme:hme_ipg2 = 5
```

3. **Save the** `/etc/system` **file.**
4. **Save all files and exit all programs, exit the windowing system.**
5. **Reboot the system by typing** `init 6` **at the** # **prompt.**

   The system is halted and then rebooted.

Setting Parameters Using the `hme.conf` File

You can also specify the properties described in the section, “Setting Parameters in the /etc/system File,” in this chapter on a per-device basis by creating the `hme.conf` file in the `/kernel/drv` directory. The properties set in the `hme.conf` file will override the parameters set in the `/etc/system` file. Use `hme.conf` when you need to set a particular parameter for a device in the system. The parameters you set are read and write parameters that are listed in Chapter 3.

The man pages for `prtconf` (1M), `system` (4) and `driver.conf` (4) include additional details. An example follows:

▼ To Set `ipg1` to 20 and `ipg2` in SBus Slot 0xe

1. **Invoke the** `prtconf -v` **command and pipe the output to the** `more` **command (**`prtconf -v | more` **) or redirect the output of the command to a file name (**`prtconf -v >` *filename***) and print the redirected file.**

2. **Find the section in the** `prtconf -v` **output for** `SUNW,hme,instance #0`**, or** `SUNW,hme,instance #1`**, and so on.**

   The output for `SUNW,hme,instance #0` for a Sun Ultra 1 Creator Series system follows:

```
SUNW,hme, instance #0
            Driver software properties:
                name <pm_norm_pwr> length <4>
                    value <0x00000001>.
                name <pm_timestamp> length <4>
                    value <0x30743b26>.
            Register Specifications:
                Bus Type=0xe, Address=0x8c00000, Size=108
                Bus Type=0xe, Address=0x8c02000, Size=2000
                Bus Type=0xe, Address=0x8c04000, Size=2000
                Bus Type=0xe, Address=0x8c06000, Size=2000
                Bus Type=0xe, Address=0x8c07000, Size=20
```

3. **Become superuser.**

4. **Create the** `hme.conf` **file in the** `/kernel/drv` **directory using a text editor and add lines similar to the following to the file:**

   a. **Specify** `name=”hme”` **and** `class=”sbus`.”

   b. **Use the** `reg` **property to specify the device,** `0xe` **in this case. Use the value following** `Bus Type` **in the** `prtconf -v` **output.**

   c. **Type the addresses followed by the specified sizes. Precede each size with** `0x` **and leading zeros, as indicated in the following screen.**

   d. **Set** `ipg1` **and** `ipg2`**. Type a semicolon (;) after the last value.**

   These parameters are set to 20 and 10, respectively, in this example. The `ipg` parameters are defined in Chapter 3.

```
name=”hme” class=”sbus”
reg=0xe,0x8c00000,0x00000108,0xe,0x8c02000,0x00002000,0xe,
0x8c04000,0x00002000,0xe,0x8c06000,0x00002000,0xe,0x8c07000,
0x00000020
ipg1=20 ipg2=10;
```

5. **Save the** `hme.conf` **file.**

6. **Save and close all files and exit all programs; exit the windowing system.**

7. **Halt and reboot the system by typing the** `init 6` **command at the** # **prompt.**

Setting Driver Parameters for PCI-Bus hme Interfaces Using `hme.conf`

▼ To Configure Driver Parameters With PCI-bus Based Systems

1. **Obtain the hardware path name for the device in the device tree. Typically this path name and the associated instance number will be present in the** `/etc/path_to_inst` **file. For example, on a Sun Ultra 30 UPA/PCI system in which one SunSwift-PCI card is plugged in, the** `/etc/path_to_inst` **file will have the following two entries (in addition to entries for other devices):**

   Obtain the hardware path name for the device in the device tree. Typically this path name and the associated instance number will be present in the `/etc/path_to_inst` file. For example, on a Sun Ultra 30 UPA/PCI system in which one SunSwift-PCI card is plugged in, the `/etc/path_to_inst` file will have the following two entries (in addition to entries for other devices):

```
“/pci@1f,4000/network@1,1” 0 “hme”
“/pci@1f,4000/pci@4/SUNW,hme@0,1” 1 “hme”
```

- The first entry corresponds to the `hme` device on the motherboard. The second entry corresponds to the hme device on the SunSwift-PCI card.
- In the previous lines:
  - The first part within the double quotes specifies the hardware node name in the device tree.
  - The second number is the instance number.
  - The last part in double quotes is the driver name.
- In the device path name, the last component after the last / character and before the @ character is the device name.
- The path name before the last component is the parent name.

- The comma-separated numbers after the @ character at the end represent the device and function numbers, which are together referred to as unit-address.

To identify a PCI device unambiguously in the hme.conf file, use the name, parent name, and the unit-address for the device. Refer to the pci(4) man page for more information about the PCI device specification.

In the first line of the previous example:

- Name = network
- Parent = /pci@1f,4000
- Unit-address = 1,1

In the second line in the previous example:

- Name = SUNW,hme
- Parent = /pci@1f,4000/pci@4
- Unit-address = 0,1

2. **Set the i**pg1 **and** ipg2 **parameters for the above two devices in the** /kernel/drv/hme.conf **file:**

```
name = “SUNW,hme” parent = “/pci@1f,4000” unit-address = “1,1” ipg1=10 ipg2=5;
name = “SUNW,hme” parent = “/pci@1f,4000/pci@4” unit-address = “0,1” ipg1=20
ipg2=10;
```

Note that for the motherboard device, the SUNW,hme device is used even though the path name uses the name network. The SUNW,hme device name is the value of the compatible property for this device.