Platform Notes: The Sun Quad FastEthernet™ Device Driver

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054 U.S.A.
650-960-1300

Part No. 816-2349-10
May 2002, Revision A

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

Please
Recycle

Adobe PostScript

Contents

Tables

Preface

This book describes how to configure the `qfe` 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.

Note that the 64-bit version of the Solaris operating environment uses the directory `/kernel/drv/sparcv9`. In this document and others, when `/kernel/drv` is mentioned, the `/kernel/drv/sparcv9` directory also applies.

How This Book Is Organized

Chapter 1 describes the hardware that uses the `qfe` driver.

Chapter 2 describes how to configure the SUNW,`qfe` device driver for your system.

Chapter 3 lists the values for each of the `qfe` driver parameters.

Chapter 4 describes how to set the parameters for the `qfe` driver.

Appendix A describes the auto-negotiation process.

Appendix B lists the SUNW,`qfe` device driver parameters.

Typographic Conventions

Typeface or Symbol	Meaning	Examples
`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, when contrasted with on-screen computer output	`% su` `Password:`
AaBbCc123	Book titles, new words or terms, words to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be superuser to do this.
	Command-line variable; replace with a real name or value	To delete a file, type `rm` *filename*.

Shell Prompts

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

Related Documentation

- The *Solaris on Sun Hardware Platform Guide* provides an overview of the `qfe` driver and the Fast Ethernet Parallel Port SCSI (FEPS) ASIC.
- The *IEEE 802.3u Ethernet Standard* provides additional information about the Ethernet standard.

Accessing Sun Documentation Online

The `docs.sun.com`SM 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 (816-2349-10) of your document in the subject line of your email.

CHAPTER 1

The qfe Device Driver

This chapter gives a hardware overview of the SUNW,qfe device, provides information on the operating speeds and modes for the SUNW,qfe device, and discusses auto-negotiation. The following sections describe features of the SUNW,qfe device.

Supported Hardware

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

- Sun Quad SBus Adapter
- Sun Quad PCI Adapter

TABLE 1-1 Transceivers Available In Sun Adapters

Adapters	Transistors
SunQuad SBus Adapter	Internal only
SunQuad PCI Adapter	Internal only

Hardware Overview

The SUNW,qfe device provides 10BASE-T 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) ASIC provides the SBus interface and Media Access Control (MAC) functions. The PFEX (PCI-bus based) ASIC provides the 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.

Operating Speeds and Modes

You can operate the link in either of the following speeds and modes with the SUNW,qfe device

- 100 BASE-T
- 10 BASE-T

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

Auto-Negotiation

The auto-negotiation protocol, as specified by the *100BASE-T IEEE 802.3u Ethernet Standard*, selects the operation mode (half-duplex or full-duplex) at boot time or when the link state changes (the link fails or tries to connect). The auto-negotiation protocol also selects the speed and the full-duplex or half-duplex mode.

Details of the way the SUNW,qfe device uses auto-negotiation are provided in Appendix A.

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 (in decreasing order):

- 100 BASE-T
- 10 BASE-T

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,`qfe` 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 of your choice.

CHAPTER 2

Installing and Configuring `qfe` Driver

This chapter includes information and instructions to install and configure the driver software used by the Sun Quad FastEthernet Device Driverdevice driver. Unless otherwise noted, all instructions apply to both the Sun Quad FastEthernet PCI adapter and the Sun Quad FastEthernet SBus adapter.

This chapter includes the following sections:

Installing the Driver Software

The Solaris CD-ROM contains the software that must be installed in order to use the Sun Quad FastEthernet Device Driverdriver.

Note – Do not use the installation CD-ROM that shipped with your adapter. The software on the Solaris CD-ROM is more current and replaces previous versions of the driver.

Before using the SUNW,`qfe` device as your network interface, you will need to create and edit configuration files, as described in the next section.

Configuring the Hostname File

▼ To Configure the Hostname File

1. **At the command line, use the** `grep` **command to search the** `/etc/path_to_inst` **file for** `qfe` **devices.**

   For a Sun Quad FastEthernet PCI adapter:

```
# grep qfe /etc/path_to_inst
"/pci@1f,2000/pci@2/SUNW,qfe@0,1" 0 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@1,1" 1 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@2,1" 2 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@3,1" 3 "qfe"
```

   In the example above, the four SUNW,qfe@x,1 instances are from a Sun Quad FastEthernet Device DriverPCI adapter installed in slot 2.

   For a Sun Quad FastEthernet SBus adapter:

```
# grep qfe /etc/path_to_inst
"/sbus@1f,0 /SUNW,qfe@1,8c10000" 1 "qfe"
"/sbus@1f,0 /SUNW,qfe@1,8c00000" 0 "qfe"
"/sbus@1f,0 /SUNW,qfe@1,8c30000" 3 "qfe"
"/sbus@1f,0 /SUNW,qfe@1,8c20000" 2 "qfe"
```

   In the example above, the four SUNW,qfe@1 instances are from a Sun Quad FastEthernet Device DriverSbus adapter installed in slot 1.

2. Create an `/etc/hostname.qfe`*num* file, where *num* is the instance number of each interface you plan to use.

   If you want to use the network interface from the example in Step 1, you will need to create four files:

File Name	Instance Number	Adapter Ethernet Network Interface
`/etc/hostname.qfe0`	0	0

File Name	Instance Number	Adapter Ethernet Network Interface
/etc/hostname.qfe1	1	1
/etc/hostname.qfe2	2	2
/etc/hostname.qfe3	3	3

- Do not create /etc/hostname.qfe*num* files for Sun Quad FastEthernet Device Driver network interfaces you plan to leave unused.
- The /etc/hostname.qfe*num* file must contain the host name for the appropriate network interface.
- The host name should have an IP address that will need to be entered in the /etc/hosts file.
- The host name should be different from the host name of any other interface, for example: /etc/hostname.qfe0 and /etc/hostname.qfe2 cannot share the same host name.

Using the instance examples in Step 1, the following example shows the four /etc/hostname.qfe*num* files required for a system called zardoz that has a Sun Quad FastEthernet Device Driver (zardoz-11, zardoz-12, zardoz-13, and zardoz-14).

```
# cat /etc/hostname.qfe0
zardoz
# cat /etc/hostname.qfe0
zardoz-11
# cat /etc/hostname.qfe1
zardoz-12
# cat /etc/hostname.qfe2
zardoz-13
# cat /etc/hostname.qfe3
zardoz-14
```

3. **Create an appropriate entry in the `/etc/hosts` file for each active `qfe` network interface.**

   Using the previous example, you will have:

```
# cat /etc/hosts
#
# Internet host table
#
127.0.0.1     localhost
129.144.10.57 zardoz    loghost
129.144.11.83 zardoz-11
129.144.12.41 zardoz-12
129.144.13.67 zardoz-13
129.144.14.30 zardoz-14
```

Note – The Internet Protocol, version 6 (IPv6), expands the capabilities of IPv4, which is the current version and the default. The Sun Quad FastEthernet Device Driverdevice driver included in this release of the Solaris operating environment supports both IPv4 and IPv6. IPv4 uses the /etc/hosts configuration file, but IPv6 uses a different configuration file. To transition to, manage, and implement IPv6, refer to the Solaris System Administration documentation.

4. **Reboot your system.**

Booting From the Network

To use a Sun Quad FastEthernet Device Driver interface as the boot device, perform the following tasks:

▼ To Boot From the Network

1. **At the** `ok` **prompt type:**

```
ok show-devs
```

The `show-devs` command lists the system devices. You should see the full path name of the `qfe` devices, similar to the following examples:

For Sun Quad FastEthernet PCI adapter:

```
/pci@1f,2000/pci@2/SUNW,qfe@0,1
/pci@1f,2000/pci@2/SUNW,qfe@1,1
/pci@1f,2000/pci@2/SUNW,qfe@2,1
/pci@1f,2000/pci@2/SUNW,qfe@3,1
```

For Sun Quad FastEthernet SBus adapter:

```
/sbus@1f,0 /SUNW,qfe@1,8c3000
/sbus@1f,0 /SUNW,qfe@1,8c2000
/sbus@1f,0 /SUNW,qfe@1,8c1000
/sbus@1f,0 /SUNW,qfe@1,8c0000
```

Note – Select only one of these `qfe` devices for booting.

---

2. **At the** `ok` **prompt type:**

```
ok boot full_path_name_of_the_qfe_device
```

For example:

```
ok boot /sbus@1f,0 /SUNW,qfe@1,8c3000
```

Optional Post-Installation Procedures

To customize the performance of the Sun Quad FastEthernet Device Driverdevice, perform the tasks in the following sections.

Setting Driver Parameters

The `qfe` device driver, which is loaded from the Solaris CD-ROM, controls the `SUNW,qfe` Ethernet devices. The device driver selects the link speed using the auto-negotiation protocol with the link partner. (See "The Auto-Negotiation Protocol" on page A35.)

You can manually set the `qfe` device driver parameters to customize each SUNW,`qfe` device in your system in one of three ways:

- Set a parameter on a per-device basis by creating the `qfe.conf` file in the `/kernel/drv` directory.
- Use the `ndd` utility to *temporarily* change a parameter. This change is lost when you reboot the system.
- Set the `qfe` driver parameters generally for all SUNW,`qfe` devices in the system by entering the parameter variables in the `/etc/system` file.

See "Parameter Setting Options" on page 425 for more information.

Note – In the future, the `/etc/system` file will not be available. It is not compatible with dynamic reconfiguration.

▼ To Force Network Speed Between 10 Mbps and 100 Mbps

1. **At the `ok` prompt, use the `show-devs` command to list the system devices.**

   You should see the full path names of the `qfe` devices, similar to the following examples:

   For Sun Quad FastEthernet PCI adapter:

```
/pci@1f,2000/pci@2/SUNW,qfe@0,1
/pci@1f,2000/pci@2/SUNW,qfe@1,1
/pci@1f,2000/pci@2/SUNW,qfe@2,1
/pci@1f,2000/pci@2/SUNW,qfe@3,1
```

   For Sun Quad FastEthernet SBus adapter:

```
/sbus@1f,0/SUNW,qfe@1,8c30000
/sbus@1f,0/SUNW,qfe@1,8c20000
/sbus@1f,0/SUNW,qfe@1,8c10000
/sbus@1f,0/SUNW,qfe@1,8c00000
```

2. **Type:**

```
ok nvedit
```

3. **Type the following, pressing the Return key at the end of line `0`:**

```
0: probe-all install-console banner
1: apply transfer-speed=10 full_ path_name_of_a_qfe_device
```

Note – If you already have commands in NVRAM, append these lines to the end of the file.

---

4. **Press Control-C after typing *full_ path_name_of_a_`qfe`_device*.**

   Perform Steps 2 to 4 to set the network speed for each `qfe` network interface.

Note – In the preceding example, the speed is forced to 10 Mbps. To force the speed to 100 Mbps, replace 10 with 100.

---

5. **At the `ok` prompt type:**

```
ok nvstore
ok setenv use-nvramrc? true
```

6. **Reboot your system.**

   See "Setting Forced Mode" on page 429 for more information on forcing network speed.

local-mac-address Property

Each of the network interfaces of the Sun Quad FastEthernet Device Driver has been assigned a unique Media Access Control (MAC) address, which represents the 48-bit Ethernet address for that network interface. The OpenBoot™ firmware reports this MAC address via the `local-mac-address` property in the device nodes corresponding to the network interfaces.

A system is not obligated to use this assigned MAC address if it has a systemwide MAC address. In such cases, the systemwide MAC address applies to all network interfaces on the system.

The device driver, or any other adapter utility, can use the network device's MAC address (`local-mac-address`) while configuring it. A network interface's MAC address can be used when booting over the network.

The `mac-address` property of the network device specifies the network address (systemwide or `local-mac-address`) used for booting the system. To start using the MAC addresses assigned to the network interfaces of the Sun Quad FastEthernet Device Driver, set the NVRAM configuration variable `local-mac-address?` to `true`.

```
ok setenv local-mac-address? true
```

To Avoid Losing TCP/IP Parameter Changes

Add the parameter change to a run control script in the `/etc/rc2.d` directory, similar to the following example

```
#!/sbin/sh
# Local kernel modifications
#
case "$1" in
'start')
    echo "Setting local kernel parameters...\c"
    ndd -set /dev/tcp tcp_rexmit_interval_max 60000
    echo ""
    ;;
'stop')
    echo "No kernel parameters changed."
    ;;
*)
    echo "Usage: $0 {start|stop}"
    ;;
esac
exit 0
```

CHAPTER 3

Parameter Definitions

This chapter describes the parameters and settings for the `qfe` device driver.

Driver Parameter Values and Definitions

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

TABLE 3-1 `qfe` Driver Parameters, Status, and Descriptions

Parameter	Status	Description	Details
transciever_inuse	Read only	Defines the current status	
`link_status`	Read only	Defines the current status	"Defining the Current Status" on page 317
`link_speed`	Read only	Defines the current status	
`link_mode`	Read only	Defines the current status	

TABLE 3-1 `qfe` Driver Parameters, Status, and Descriptions *(Continued)*

Parameter	Status	Description	Details
`ipg1`	Read and write	Inter-packet gap parameter	"Inter-Packet Gap Parameters" on page 317
`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	"Defining the Number of Back-to-Back Packets to Transmit" on page 321
`adv_autoneg_cap`	Read and write	Operational mode parameter	"Reporting Transceiver Capabilities" on page 321
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	
adv_10hdx_cap	Read and write	Operational mode parameter	
`autoneg_cap`	Read only	Local transceiver auto negotiation capability	"Defining the Current Status" on page 317
`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	"Reporting the Link Partner Capabilities" on page 322
`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	

TABLE 3-1 `qfe` Driver Parameters, Status, and Descriptions *(Continued)*

Parameter	Status	Description	Details
instance	Read and write	Device instance	
`lance_mode`	Read and write	Additional delay before transmitting a packet	"Defining an Additional Delay Before Transmitting a Packet Using lance_mode and ipg0" on page 318
`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. Click on this link to return to "Driver Parameter Values and Definitions" on page 315

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

Parameter	Description	Values
`link_status`	Current link status	0 = Link down 1 = Link up
`link_speed`	Valid only if the link is up	0 = 10 Mbps 1 = 100 Mbps
`link_mode`	Valid only if the link is up	0 = Half duplex 1 = 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 IPG parameters, `ipg1` and `ipg2`. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

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

The following two ASICs support a programmable mode called `lance_mode`. The `ipg0` parameter is associated with `lance_mode`:

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

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. 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 8000 ns. If the link speed is 100 Mbps, and you set `ipg0` to 30 nibble-times, multiply 30 by 40 ns to get 1200 ns.

TABLE 3-4 defines the `lance_mode` and `ipg0 parameters`. Click on this link to return to "Driver Parameter Values and Definitions" on page 315 .

TABLE 3-4 Parameters Defining `lance_mode` and `ipg0`

Parameter	Values	Description
`lance_mode`	0	`lance_mode` disabled
	1	`lance_mode` enabled (default)
`ipg0`	0-31 — See the following Note	Additional IPG before transmitting a packet (after receiving a packet)

Note – The default value for `ipg0` 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. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

TABLE 3-5 Operational Mode Parameters

Parameter	Description	Values
`adv_autoneg_cap`	Local transceiver capability advertised by the hardware	0 = Forced mode 1 = Auto-negotiation (default)
`adv_100fdx_cap`	Local transceiver capability advertised by the hardware; read/write parameter	0 = Not 100 Mbit/sec full-duplex capable 1 = 100 Mbit/sec full-duplex capable (default)

TABLE 3-5 Operational Mode Parameters *(Continued)*

Parameter	Description	Values
`adv_100hdx_cap`	Local transceiver capability advertised by the hardware; read/write parameter	0 = Not 100 Mbit/sec half-duplex capable 1 = 100 Mbit/sec half-duplex capable (default)
`adv_10fdx_cap`	Local transceiver capability advertised by the hardware; read/write parameter	0 = Not 10 Mbit/sec full-duplex capable 1 = 10 Mbit/sec full-duplex capable (default)
`adv_10hdx_cap`	Local transceiver capability advertised by the hardware; read/write parameter	0 = Not 10 Mbit/sec half-duplex capable 1 = 10 Mbit/sec half-duplex capable (default)

Operational Mode Priorities

TABLE 3-6 lists the priority of the operational mode parameters. When the Sun `qfe` FastEthernet Device Driver negotiates with a partner on the network, it implements the operational mode in the order shown in TABLE 3-6. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

TABLE 3-6 Operational Mode Priorities

Priority	Parameter
First	adv_100fdx_cap
Second	adv_100hdx_cap
Third	adv_10fdx_cap
Fourth	adv_10hdx_cap

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

The `pace_size` parameter (see TABLE 3-7) 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. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

TABLE 3-7 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 3-8 describes the read-only transceiver capabilities (either the internal transceiver or the external transceiver), whichever is selected. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

TABLE 3-8 Read-Only Transceiver Capabilities

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

TABLE 3-8 Read-Only Transceiver Capabilities

Parameter	Description	Values
`100hdx_cap`	Local transceiver capability of the hardware; initialized at startup	0 = Not 100 Mbit/sec half-duplex capable 1 = 100 Mbit/sec half-duplex capable
`10fdx_cap`	Local transceiver capability of the hardware; initialized at startup	0 = Not 10 Mbit/sec full-duplex capable 1 = 10 Mbit/sec full-duplex capable
`10hdx_cap`	Local transceiver capability of the hardware; initialized at startup	0 = Not 10 Mbit/sec half-duplex capable 1 = 10 Mbit/sec half-duplex capable

The parameters in TABLE 3-8 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 3-9 describes the read-only link partner capabilities. Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

TABLE 3-9 Read-Only Link Partner Capabilities

Parameter	Values
`lp_autoneg_cap`	0 = No auto-negotiation 1 = Auto-negotiation
lp_100T4_cap	0 = No 100BASE-T4 1 = 100BASE-T4
`lp_100fdx_cap`	0 = No 100 Mbit/sec full-duplex transmission 1 = 100Mbit/sec full-duplex

TABLE 3-9 Read-Only Link Partner Capabilities *(Continued)*

Parameter	Values
`lp_100hdx_cap`	0 = No 100 Mbit/sec half-duplex transmission 1 = 100 Mbit/sec half-duplex
`lp_10fdx_cap`	0 = No 10 Mbit/sec full-duplex transmission 1 = 10 Mbit/sec full-duplex
`lp_10hdx_cap`	0 = No 10 Mbit/sec half-duplex transmission 1 = 10 Mbit/sec half-duplex

If the link partner is not capable of auto-negotiation (when `lp_autoneg_cap` is 0) the information described in TABLE 3-9 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

Click on this link to return to "Driver Parameter Values and Definitions" on page 315.

CHAPTER 4

Setting Parameters

This chapter describes three methods to configure the `qfe` driver parameters. This chapter contains the following sections:

Parameter Setting Options

You can set the `qfe` device driver parameters in three ways (`ndd`, `/etc/system`, and `qfe.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. If you want to test parameter settings, use the `ndd` utility described in "Setting Parameters Using ndd" on page 426. With `ndd`, the parameters are effective until you reboot the system.

To configure the `qfe` 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.. "Setting Parameters in the `/etc/system` File" on page 430 describes this option.

To set the parameters for a particular device in the system, set the parameters in the `qfe.conf` file in the `/kernel/drv` directory. The parameters set in the `qfe.conf` file have precedence over the parameters set in the `/etc/system` file and override the parameters set in the `/etc/system` file. The parameters values set in `qfe.conf` are always in effect (even after rebooting the system)."Setting Parameters Using the qfe.conf File" on page 431 describes this option.

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 qfe driver and the ndd utility to modify (with the -set option) or display (without the -set option) the parameters for each SUNW,qfe device.

Identifying Device Instances

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

Note – If there is only one SUNW,qfe 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.**

   You should see the full path names of the qfe devices, similar to the following examples:

   For Sun Quad FastEthernet PCI adapter:

```
/pci@1f,2000/pci@2/SUNW,qfe@0,1
/pci@1f,2000/pci@2/SUNW,qfe@1,1
/pci@1f,2000/pci@2/SUNW,qfe@2,1
/pci@1f,2000/pci@2/SUNW,qfe@3,1
```

   For Sun Quad FastEthernet SBus adapter:

```
/sbus@1f,0/SUNW,qfe@1,8c30000
/sbus@1f,0/SUNW,qfe@1,8c20000
/sbus@1f,0/SUNW,qfe@1,8c10000
/sbus@1f,0/SUNW,qfe@1,8c00000
```

2. **Use that instance number to select the device as follows:**

```
% ndd -set /dev/qfe 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.

1. **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/qfe` driver instance, and assigns it to the parameter:

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

2. **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/qfe parameter
```

Using the ndd Utility in Interactive Mode

1. **To modify a parameter value in interactive mode, specify** `ndd /dev/qfe`**, as shown below.**

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

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

   After you enter the parameter name, the `ndd` utility prompts you for the parameter value (see TABLE 3-2 through TABLE 3-9).

2. **To list all the parameters supported by the** `qfe` **driver, type** `ndd /dev/qfe \?`**.**

   (See TABLE 3-2 through TABLE 3-9 for parameter descriptions.)

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

```
example# ndd /dev/qfe \?
?                               (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_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)
100fdx_cap                      (read only)
100hdx_cap                      (read only)
10fdx_cap                       (read only)
10hdx_cap                       (read only)
lp_autoneg_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#
```

Setting Forced Mode

The following procedure describes how to set forced mode. When the SUNW,qfe device is in forced mode it is not capable of auto-negotiation.

▼ To Select One Local Transceiver Capability and Setting Forced Mode

1. **Select one of the transceiver capabilities 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. The permitted values are as follows:

   - `adv_100fdx_cap`
   - `adv_100hdx_cap`
   - `adv_10fdx_cap`
   - `adv_10hdx_cap`

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 Non-Interactive Mode" on page 427 in this chapter.

Setting Auto-Negotiation Mode

▼ To Set the Mode to Auto-Negotiation

1. **Select** *at least one* **of the five transceiver capabilities that you want to advertise to the remote system, and set its value to** `1`**.**

   - `adv_100fdx_cap`
   - `adv_100hdx_cap`
   - `adv_10fdx_cap`
   - `adv_10hdx_cap`

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" on page 428 in this chapter.

Setting Parameters in the /etc/system File

To configure the qfe driver parameters for all SUNW,qfe 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 qfe 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	qfe_ipg1
ipg2	qfe_ipg2
use_int_xcvr	qfe_use_int_xcvr
pace_size	qfe_pace_size
adv_autoneg_cap	qfe_adv_autoneg_cap
adv_100fdx_cap	qfe_adv_100fdx_cap
adv_100hdx_cap	qfe_adv_100hdx_cap
adv_10fdx_cap	qfe_adv_10fdx_cap
adv_10hdx_cap	qfe_adv_10hdx_cap
lance_mode	qfe_lance_mode
ipg0	qfe_ipg0

These parameter values, described in Chapter 3, are applicable to all SUNW,qfe devices on the system. See TABLE 3-2 through TABLE 3-9 for parameter descriptions. An example follows.

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

1. **As superuser, add the following lines to the** `/etc/system` **file:**

```
set qfe:qfe_ipg1 = 10
set qfe:qfe_ipg2 = 5
```

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

   The system is halted and then rebooted.

Setting Parameters Using the qfe.conf File

You can also specify the properties described in the section, "Setting Parameters in the `/etc/system` File" on page 430," in this chapter on a per-device basis by creating the `qfe.conf` file in the `/kernel/drv` directory. The properties set in the `qfe.conf` file will override the parameters set in the `/etc/system` file. Use `qfe.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 "Driver Parameter Values and Definitions" on page 315.

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 to 10 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,qfe,instance #0`**, or** `SUNW,qfe,instance #1`**, and so on.**

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

```
SUNW,qfe, 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. **As superuser, create the** `qfe.conf` **file in the** `/kernel/drv` **directory using a text editor and add lines similar to the following to the file:**

   a. **Specify** `name="qfe"` **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 "Driver Parameter Values and Definitions" on page 315..

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

4. **Save the** `qfe.conf` **file.**

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

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

Setting Driver Parameters for PCI-Bus qfe Interfaces Using qfe.conf

▼ To Configure Driver Parameters on a Sun Quad FastEthernet PCI Adapter

1. **Obtain the hardware path name for the `qfe` devices 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 Sun Quad FastEthernet Device Driver PCI card is installed, the `/etc/path_to_inst` file will have the following entries (in addition to entries for other devices):

```
"/pci@1f,2000/network@1,1" 0 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@0,1" 4 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@1,1" 5 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@2,1" 6 "qfe"
"/pci@1f,2000/pci@2/SUNW,qfe@3,1" 7 "qfe"
```

   - The first entry corresponds to the `qfe` device on the motherboard. The second entry corresponds to the `qfe` device on the Sun Quad FastEthernet Device Driver PCI card.
   - In the following 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 qfe.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,qfe`
- Parent = `/pci@1f,4000/pci@4`
- Unit-address = `0,1`

2. **Set the `ipg1` and `ipg2` parameters for the above two devices in the `/kernel/drv/qfe.conf` file:**

```
name="SUNW,qfe" parent="/pci@1f,2000/pci@2" unit-address="0,1" ipg1=20 ipg2=10;
name="SUNW,qfe" parent="/pci@1f,2000/pci@2" unit-address="1,1" ipg1=20 ipg2=10;
name="SUNW,qfe" parent="/pci@1f,2000/pci@2" unit-address="2,1" ipg1=20 ipg2=10;
name="SUNW,qfe" parent="/pci@1f,2000/pci@2" unit-address="3,1" ipg1=20 ipg2=10;
```

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

3. **Save the `/kernel/drv/qfe.conf` file.**
4. **Save and close all programs, and exit the windowing system.**
5. **As superuser, halt and reboot the system by typing `init 6`.**

APPENDIX **A**

Auto-Negotiation

Click on this link to return to "Auto-Negotiation" on page 2.

The Auto-Negotiation Protocol

Auto-negotiation is a key feature of the Sun `qfe` FastEthernet driver. The auto-negotiation protocol, as specified by the *100BASE-T IEEE 802.3u Ethernet Standard*, selects the operation mode (half-duplex or full-duplex), and the auto-sensing protocol selects the speed (10 Mbps or 100 Mbps) for the adapter.

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 (in decreasing order):
  - 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 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 of your choice. For more information on this topice, see "Parameter Setting Options" on page 25

Boot Process on the Network

The auto-negotiation protocol does the following when the system is booted:

- 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 (in decreasing order):

Priority	Line Speed and Mode
First	100 BASE T4
Second	100 Mbps, full-duplex
Third	100 Mbps, half-duplex
Fourth	10 Mbps, full-duplex
Fifth	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.

Correcting Errors in Negotiating

If the SUNW,`qfe` device is connected to a remote system or interface that is not capable of auto-negotiation, your system automatically selects the correct speed and half-duplex mode.

If the SUNW,`qfe` device is connected to a link partner with which 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 of your choice. For more information on this topice, see “Parameter Setting Options” on page 25

Internal (Local) Transceiver

The internal transceiver is 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” on page 2,”. When the internal transceiver is used, the default is auto-negotation by the `qfe` 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 `qfe.conf` file. The `ndd` utility makes calls to the `qfe` 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" on page 2.

Click on this link to return to "Auto-Negotiation" on page 2

APPENDIX **B**

Driver Parameters

This appendix presents the complete list of SUNW,qfe device driver parameters..

Driver Parameter Definitions

TABLE B-1 lists the qfe driver parameters in the order they are encountered in the /kernel/drv/qfe.conf file..

TABLE B-1 qfe Driver Parameters, 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 B-1 `qfe` Driver Parameters, 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

Index

T

U