

Configuring the Quad FastEthernet Device Driver Parameters

The `qfe` device driver controls the Sun Quad FastEthernet SBus adapter's `SUNW,qfe` Ethernet devices. You can manually configure the `qfe` device driver parameters to customize each `SUNW,qfe` device in your system. This appendix provides an overview of the internal transceiver used in the adapter, lists the available `qfe` device driver parameters, and describes how you can configure these parameters.

Internal Transceiver

The four `SUNW,qfe` channels provide 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 ASIC provides the SBus interface and Media Access Control (MAC) functions. The internal transceiver, which connects to an RJ-45 connector, provides the physical layer functions.

The internal transceiver is capable of all the operating speeds and modes listed in the section “Auto-Negotiation” on page 23. 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. You can select the speed and mode using the `ndd` utility, by editing the `/etc/system` file, or creating a `qfe.conf` file.

Driver Parameter Values and Definitions

This section describes the parameters and settings for the `qfe` device driver. TABLE C-1 lists these parameters.

TABLE C-1 `qfe` Driver Parameter, Status, and Descriptions

Parameter	Status	Description
<code>transceiver_inuse</code>	Read only	Defines the current status
<code>link_status</code>	Read only	Defines the current status
<code>link_speed</code>	Read only	Defines the current status
<code>link_mode</code>	Read only	Defines the current status
<code>ipg1</code>	Read and write	Inter-packet gap parameter
<code>ipg2</code>	Read and write	Inter-packet gap parameter
<code>use_int_xcvr</code>	Read and write	Operational mode parameter
<code>pace_size</code>	Read and write	Operational mode parameter
<code>adv_autoneg_cap</code>	Read and write	Operational mode parameter
<code>adv_100fdx_cap</code>	Read and write	Operational mode parameter
<code>adv_100hdx_cap</code>	Read and write	Operational mode parameter
<code>adv_10fdx_cap</code>	Read and write	Operational mode parameter
<code>adv_10hdx_cap</code>	Read and write	Operational mode parameter
<code>autoneg_cap</code>	Read only	Local transceiver auto negotiation capability
<code>100fdx_cap</code>	Read only	Local transceiver capability of the hardware
<code>100hdx_cap</code>	Read only	Local transceiver capability of the hardware
<code>10fdx_cap</code>	Read only	Local transceiver capability of the hardware
<code>10hdx_cap</code>	Read only	Local transceiver capability of the hardware
<code>lp_autoneg_cap</code>	Read only	Link partner auto negotiation capability
<code>lp_100fdx_cap</code>	Read only	Link partner capability
<code>lp_100hdx_cap</code>	Read only	Link partner capability

TABLE C-1 qfe Driver Parameter, Status, and Descriptions (Continued)

Parameter	Status	Description
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 C-2 explain the operational mode of the interface. These parameters define the current status.

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

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

Inter-Packet Gap Parameters

The Fast Ethernet Parallel Port SCSI (FEPS) ASIC supports the 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 C-3 lists the default values and allowable values for the inter-packet gap (IPG) parameters `ipg1` and `ipg2`.

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

Parameter	Values (Byte-time)	Description
<code>ipg1</code>	0, 255	<code>ipg1</code> = 8 (default at initialization)
<code>ipg2</code>	0, 255	<code>ipg2</code> = 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 Fast Ethernet Parallel Port SCSI (FEPS) ASIC supports 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 add the additional delay by setting 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 C-4 defines the `lance_mode` and `ipg0` parameters.

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

Parameter	Values	Description
<code>lance_mode</code>	0	<code>lance_mode</code> disabled
	1	<code>lance_mode</code> enabled (default)
<code>ipg0</code>	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 C-5 describes the operational mode parameters and their default values.

TABLE C-5 Operational Mode Parameters

Parameter	Values	Description
<code>adv_autoneg_cap</code>	0	Local transceiver capability advertised by the hardware = Forced mode
	1	= Auto-negotiation (default)
<code>adv_100fdx_cap</code> ¹	0	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 environments)
	1	= 100Mbit/sec full-duplex capable (default in the Solaris 2.6 software environment)
<code>adv_100hdx_cap</code> ¹	0	Local transceiver capability advertised by the hardware; read/write parameter = Not 100Mbit/sec half-duplex capable
	1	= 100Mbit/sec half-duplex capable (default)
<code>adv_10fdx_cap</code> ¹	0	Local transceiver capability advertised by the hardware; read/write parameter = Not 10Mbit/sec full-duplex capable (default)
	1	= 10Mbit/sec full-duplex capable
<code>adv_10hdx_cap</code> ¹	0	Local transceiver capability advertised by the hardware; read/write parameter = Not 10Mbit/sec half-duplex capable
	1	= 10Mbit/sec half-duplex capable (default)

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

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

The `pace_size` parameter (see TABLE C-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 C-6 Back-to-back Packet Transmission Capability

Parameter	Values	Description
<code>pace_size</code>	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 C-7 describes the read-only transceiver capabilities. These parameters define the capabilities of the hardware. The internal transceiver can support all of these capabilities.

TABLE C-7 Read-Only Transceiver Capabilities

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

Reporting the Link Partner Capabilities

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

TABLE C-8 Read-Only Link Partner Capabilities

Parameter	Values	Description
lp_autoneg_cap	0	= No auto-negotiation
	1	= Auto-negotiation
lp_100fdx_cap	0	= No 100Mbit/sec full-duplex transmission
	1	= 100Mbit/sec full-duplex
lp_100hdx_cap	0	= No 100Mbit/sec half-duplex transmission
	1	= 100Mbit/sec half-duplex
lp_10fdx_cap	0	= No 10Mbit/sec full-duplex transmission
	1	= 10Mbit/sec full-duplex
lp_10hdx_cap	0	= No 10Mbit/sec half-duplex transmission
	1	= 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 C-8 is not relevant and the parameter value = 0.

If the link partner is capable of auto-negotiation (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.

Setting `qfe` Driver Parameters

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.

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

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

Setting Parameters Using the `ndd` Utility

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 a `qfe` device, you must specify the device instance for the utility since there will be at least four `SUNW,qfe` devices.

▼ 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.

```
# grep qfe /etc/path_to_inst
"/sbus@1f,0/SUNW,qfe@1,8c10000" 2 "qfe"
"/sbus@1f,0/SUNW,qfe@1,8c00000" 1 "qfe"
"/sbus@1f,0/SUNW,qfe@1,8c30000" 4 "qfe"
"/sbus@1f,0/SUNW,qfe@1,8c20000" 3 "qfe"
```

In the example above, the four `SUNW,qfe@x,1` instances are from a Sun Quad FastEthernet SBus adapter installed in slot 1. For clarity, the instance numbers are bold.

2. Use the instance number to select the device.

```
# 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 and to display parameter values.

- **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
```

- **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

- **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 entering the parameter name, the `ndd` utility prompts you for the parameter value (see TABLE C-1 through TABLE C-8).

- To list all the parameters supported by the `qfe` driver, type `ndd /dev/qfe \?`. (See TABLE C-1 through TABLE C-8 for parameter descriptions.)

```
# 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)
#
```

FIGURE C-1 Example of Listing All Parameters Supported by the `qfe` Driver

Setting Forced Mode

This section 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_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 (see the footnote from TABLE C-5 on page 35).

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” on page 40.

Auto-Negotiation Mode

This section describes how to select at least one of the four local transceiver capabilities and set the mode to auto-negotiation.

▼ To Set the Mode to Auto-Negotiation

1. **Select at least one of the four capabilities** (`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” on page 40.

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 C-9 lists the variables you can set in the `/etc/system` file.

TABLE C-9 Setting Variables in the `/etc/system` File

Parameter	Variable
<code>ipg1</code>	<code>qfe_ipg1</code>
<code>ipg2</code>	<code>qfe_ipg2</code>
<code>use_int_xcvr</code>	<code>qfe_use_int_xcvr</code>
<code>pace_size</code>	<code>qfe_pace_size</code>
<code>adv_autoneg_cap</code>	<code>qfe_adv_autoneg_cap</code>
<code>adv_100fdx_cap</code>	<code>qfe_adv_100fdx_cap</code>
<code>adv_100hdx_cap</code>	<code>qfe_adv_100hdx_cap</code>
<code>adv_10fdx_cap</code>	<code>qfe_adv_10fdx_cap</code>
<code>adv_10hdx_cap</code>	<code>qfe_adv_10hdx_cap</code>
<code>lance_mode</code>	<code>qfe_lance_mode</code>
<code>ipg0</code>	<code>qfe_ipg0</code>

These parameter values, described in “Driver Parameter Values and Definitions” on page 32, are applicable to all `SUNW,qfe` devices on the system. See TABLE C-1 through TABLE C-8 for the descriptions of these parameters.

Here's an example of setting parameters in a `/etc/system` file:

▼ Setting the `ipg1` and `ipg2` Parameters in the `/etc/system` File

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

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

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

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 a per-device basis by creating a `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 a `qfe.conf` file 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 32.

The man pages for `prtconf (1M)`, `system (4)` and `driver.conf (4)` include additional details. The next section shows an example of setting parameters in a `qfe.conf` file.

Setting ipg Driver Parameters Using a qfe.conf File

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. Become superuser.
4. 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 Chapter 3.

```
name="qfe" 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 qfe.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.**