

Chapter 6

Interface Configuration and Support

6

This chapter describes how to configure and maintain the interfaces supported on the protocol translator. You will find information about enabling, shutting down, and displaying statistics about the following interfaces:

- Serial interfaces
- Ethernet interfaces
- Token Ring interfaces

You will also find information about configuring the null interface and the Point-to-Point Protocol (PPP) in this chapter.

To enable an interface, you must be in the configuration command collection mode. To enter this mode, type the EXEC command **configure** at the EXEC prompt. Once in the command collection mode, start configuring the interface by entering the **interface** command. Once an interface is configured, you can check its status by entering EXEC **show** commands at the EXEC prompt.

This chapter provides software configuration information only. For hardware technical descriptions, and for information about installing these interfaces, refer to the hardware reference for your particular product.

Summaries of the interface configuration commands and EXEC monitoring commands described in this chapter are included at the end of the chapter.

Specifying an Interface

The **interface** command is entered in configuration mode and identifies a specific network interface (for example, a serial port, Ethernet port, or a Token Ring port.) By entering this command you begin the command collection mode for the specified interface.

The **interface** command has the following syntax:

```
interface type unit
```

The argument *type* identifies the interface type and the argument *unit* identifies the connector or interface card number. Unit numbers are assigned at the factory at the time of installation, or when added to a system, and can be displayed with the **show interfaces** command.

Example:

This example begins interface configuration command collection mode for serial connector zero (interface serial 0).

```
interface serial 0
```

Use the EXEC command **show interfaces** (described later in this chapter), to determine the interface type and unit numbers.

In the interface configuration command collection mode, you enter the interface subcommands for your particular routing or bridging protocol. The interface configuration command collection mode ends when you enter a command that is not an interface subcommand, or when you type the Ctrl-Z sequence.

Adding a Descriptive Name to an Interface

To add a descriptive name to an interface, use the **description** interface subcommand.

```
description name-string  
no description
```

The argument *name-string* is text, or a description to help you remember what is attached to this interface. The **description** command is meant solely as a comment to be put in the configuration to help you remember what certain interfaces are used for. The description will appear in the output of the following commands: **show configuration**, **write terminal**, and **show interfaces**.

Example:

This example describes a 3174 controller on serial 0.

```
interface serial 0  
description 3174 Controller for test lab
```

Shutting Down and Restarting an Interface

You disable an interface using the **shutdown** interface subcommand. The full syntax for this command follows:

```
shutdown  
no shutdown
```

The **shutdown** command disables all functions on the specified interface. The command also marks the interface as unavailable. On serial interfaces, this command causes the DTR signal to be dropped. On Token Ring interfaces, this command causes the interface to be de-inserted from the ring.

To restart a disabled interface, use the **no shutdown** interface subcommand.

To check whether an interface is disabled, use the EXEC command **show interfaces** as described in the next section. An interface that has been shut down is shown as administratively down in the display from this command.

Examples:

These commands turn off the interface Ethernet 0.

```
interface ethernet 0  
shutdown
```

These commands turn the interface back on.

```
interface ethernet 0  
no shutdown
```

Clearing Interface Counters

To clear the interface counters shown with the **show interfaces** command, enter the following command at the EXEC prompt:

```
clear counters [type unit]
```

The command clears all the current interface counters from the interface unless the optional arguments *type* and *unit* are specified to clear only a specific interface type (serial, Ethernet, Token Ring, and so on) from a specific unit or card number.

Note: This command will not clear counters retrieved using SNMP, but only those seen with the EXEC **show interface** command.

Displaying Information About an Interface

The Cisco software contains commands that you can enter at the **EXEC** prompt to display different information about the interface including the version of the software and the hardware, the controller status, and some statistics about the different interfaces. These commands begin with the word “show.” (The full list of these commands can be displayed by entering the command **show ?** at the EXEC prompt.) A description of interface-specific **show** commands follows.

Displaying Controller Status

The **show controllers** command displays current internal status information for different interface cards. Enter this command at the EXEC prompt:

```
show controllers {serial | token | mci}
```

Use the following keywords to display the information about that card:

- **serial**—For the Serial Interface Card
- **token**—For the CSC-R and CSC-R16 Token Ring Interface Cards
- **mci**—For the Multiport Communications Interface Card and Serial Communications Interface Card

Sample output for the MCI controller card follows. Table 1-1 describes the fields seen.

```
MCI 0, controller type 1.1, microcode version 1.8
 128 Kbytes of main memory, 4 Kbytes cache memory
22 system TX buffers, largest buffer size 1520
Restarts: 0 line down, 0 hung output, 0 controller error
Interface 0 is Ethernet0, station address 0000.0c00.d4a6
 15 total RX buffers, 11 buffer TX queue limit, buffer size 1520
  Transmitter delay is 0 microseconds
Interface 1 is Serial0, electrical interface is V.35 DTE
 15 total RX buffers, 11 buffer TX queue limit, buffer size 1520
  Transmitter delay is 0 microseconds
  High speed synchronous serial interface
Interface 2 is Ethernet1, station address aa00.0400.3be4
 15 total RX buffers, 11 buffer TX queue limit, buffer size 1520
  Transmitter delay is 0 microseconds
Interface 3 is Serial1, electrical interface is V.35 DCE
 15 total RX buffers, 11 buffer TX queue limit, buffer size 1520
  Transmitter delay is 0 microseconds
  High speed synchronous serial interface
```

Table 1-1

Show Controllers Field Descriptions

| Field | Description |
|----------------------|--|
| MCI (number) | The unit number of the MCI card |
| controller type | The version number of the MCI card |
| microcode version | The version number of the MCI card's internal software (in read-only memory) |
| main memory | The amount of main and cache memory on the cache memory card |
| cache memory | |
| system TX | Number of buffers that hold packets to be transmitted |
| Restarts | Count of restarts due to the following conditions: |
| line down | Communication line down |
| hung output | Output unable to transmit |
| controller error | Internal error |
| interface..is | Names of interfaces, by number |
| electrical interface | Line interface type for serial connections |
| RX buffers | Number of buffers for received packets |
| TX queue limit | Maximum number of buffers in transmit queue |
| Transmitter delay | Delay between outgoing frames |
| Station address | The hardware address of the interface |

Displaying Interface Statistics

The Cisco software also provides the **show interfaces** command which displays statistics for the network interfaces on the network server. Enter this command at the EXEC prompt:

show interfaces [*type unit*]

Specify the optional arguments *type* and *unit* to display statistics for a particular network interface. The argument *type* can be one of the following: **ethernet**, **serial**, or **tokenring**. Use the argument *unit* to specify the interface unit number.

You will use the **show interfaces** command frequently while configuring and monitoring your modules.

For further explanations and examples about a specific interface, refer to the following sections in this chapter: “Monitoring the Serial Interface,” “Monitoring the Ethernet Interface,” and “Monitoring the Token Ring Interface.”

Serial Interface Support

Support for the serial interface is supplied on one of Cisco Systems' serial network interface cards:

- The Multiport Communications Interface (MCI) card provides up to two high-speed synchronous serial port connectors on a single card that support RS-232, V.35, and RS-449 connections, and X.21 connections using the RS-449 connector.
- The Serial-Port Communication Interface (SCI) card provides up to four high-speed serial ports on a single card that support RS-232, V.35, and RS-449 connections, and X.21 connections using the RS-449 connector.

The high-speed synchronous serial interface is also supported on the IGS network server.

Specifying a Serial Interface

To specify a serial interface, use this configuration command:

```
interface serial unit
```

Specify the serial interface connector number with the argument *unit*.

Follow this command with the interface subcommands for your particular protocol or application as described in the chapters in Part Five.

The SCI and MCI cards can query the appliques to determine their types. However, they do so only at system startup, so the appliques must be attached when the system is started. Issue a **show controllers serial** or **show controllers mci** command to determine how the serial card (either MCI or SCI) has identified them. The command will also show the capabilities of the serial card and report controller-related failures.

Example:

This command begins configuration on interface serial 0.

```
interface serial 0
```

Serial Encapsulation Methods

The serial interfaces support the following kinds of serial encapsulations:

- High-Level Data Link Control (HDLC)
- HDLC Distant Host (HDH)
- Frame Relay
- Point-to-Point Protocol (PPP)

- Switched Multi-megabit Data Services (SMDS)
- X.25-based encapsulations

The HDLC and PPP encapsulation methods are described in this chapter. The Frame Relay encapsulation method is described in the “Frame Relay Configuration and Management” chapter, the SMDS encapsulation in the “SMDS Configuration and Management” chapter, and the X.25 and LAPB encapsulation methods in the “X.25 Configuration and Management” chapter.

The encapsulation method is changed by using the interface configuration subcommand **encapsulation** followed by a keyword that defines the encapsulation method.

encapsulation *encapsulation-type*

The *encapsulation-type* argument is a keyword that identifies one of the following serial encapsulation types that Cisco Systems’ software supports:

- **bfex25**—Blacker Front End Encryption X.25 operation
- **ddnx25-dce**—DDN X.25 DCE operation
- **ddnx25**—DDN X.25 DTE operation
- **frame-relay**—Frame Relay
- **hdh**—HDH Protocol
- **hdlc**—Cisco Systems HDLC Protocol
- **lapb-dce**—X.25 LAPB DCE operation
- **lapb**—X.25 LAPB DTE operation
- **multi-lapb-dce**—X.25 LAPB multiprotocol DCE operation
- **multi-lapb**—X.25 LAPB multiprotocol DTE operation
- **ppp**—Point-to-Point Protocol (PPP)
- **smds**—SMDS service
- **x25-dce**—X.25 DCE operation
- **x25**—X.25 DTE operation

HDLC Serial Encapsulation Method

Cisco provides HDLC serial encapsulation for serial lines. This encapsulation method provides the synchronous framing and error detection functions of HDLC without windowing or retransmission. Although HDLC is the default serial encapsulation method, it can be re-installed using the **hdlc** keyword with the **encapsulation** command as follows:

encapsulation hdlc

Maintaining the Serial Interface

Use the command **clear interface** to reset the hardware logic on an interface. Enter this command at the EXEC prompt:

```
clear interface serial unit
```

The argument *unit* specifies the serial part number. In this case, the argument *type* is **serial**.

Note: Under normal circumstances, you do not need to clear the hardware logic on interfaces.

Monitoring the Serial Interface

Use the command **show interfaces serial** to display information about the serial interface and the state of source bridging. Enter this command at the EXEC prompt:

```
show interfaces serial [unit]
```

The argument *unit* is the interface unit number. If you do not provide values for the *unit* argument, the command will display statistics for all the network interfaces.

Sample output of this command for Cisco's synchronous serial interfaces is provided below: Table 1-2 describes the fields seen.

```
Serial 0 is up, line protocol is up
  Hardware is MCI Serial
  Internet address is 150.136.190.203, subnet mask is 255.255.255.0
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec, rely 255/255, load 1/255
  Encapsulation HDLC, loopback not set, keepalive set (10 sec)
  Last input 0:00:07, output 0:00:00, output hang never
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  Five minute input rate 0 bits/sec, 0 packets/sec
  Five minute output rate 0 bits/sec, 0 packets/sec
    16263 packets input, 1347238 bytes, 0 no buffer
    Received 13983 broadcasts, 0 runts, 0 giants
    2 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 2 abort
    22146 packets output, 2383680 bytes, 0 underruns
    0 output errors, 0 collisions, 2 interface resets, 0 restarts
    1 carrier transitions
```


Table 1-2 Show Serial Interface Field Descriptions

| Field | Description |
|--|---|
| Serial ... is {up down} ...is administratively down | Tells whether the interface hardware is currently active (whether carrier detect is present) and if it has been taken down by an administrator. |
| line protocol is {up down administratively down} | Tells whether the software processes that handle the line protocol think the line is usable (are keepalives successful?). |
| Hardware is | Specifies the hardware type. |
| Internet address is | Specifies the Internet address and subnet mask, followed by packet size. |
| MTU | Maximum Transmission Unit of the interface. |
| BW | Bandwidth of the interface in kilobits per second. |
| DLY | Delay of the interface in microseconds. |
| rely | Reliability of the interface as a fraction of 255 (255/255 is 100% reliability), calculated as an exponential average over five minutes. |
| load | Load on the interface as a fraction of 255 (255/255 is completely saturated), calculated as an exponential average over five minutes. |
| Encapsulation | Encapsulation method assigned to interface. |
| loopback | Tells whether loopback is set or not. |
| keepalive | Tells whether keepalives are set or not. |
| Last input | The number of hours, minutes, and seconds since the last packet was successfully received by an interface. Useful for knowing when a dead interface failed. |
| output hang | The number of hours, minutes, and seconds (or never) since the interface was last reset because of a transmission that took too long. When the number of hours in any of the "last" fields exceeds 24 hours, the number of days and hours is printed. If that field overflows, asterisks are printed. |
| Output queue, Input Queue, drops | Number of packets in output and input queues. Each number is followed by a slash, the maximum size of the queue, and the number of packets dropped due to a full queue. |
| Five minute input rate, Five minute output rate | The average number of bytes and packets transmitted per second in the last five minutes. |
| packets input | The total number of error-free packets received by the system. |
| broadcasts | The total number of broadcast or multicast packets received by the interface. |

| Field | Description |
|----------------|--|
| runts | The number of packets which are discarded because they are smaller than the medium's minimum packet size. |
| giants | The number of packets which are discarded because they exceed the medium's maximum packet size. |
| input error | The total number of no buffer, runts, giants, CRCs, frame, overrun, ignored, and abort counts. Other input-related errors can also increment the count, so that this sum may not balance with the other counts. |
| CRC | The Cyclic Redundancy Checksum generated by the originating station or far-end device does not match the checksum calculated from the data received. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link. |
| frame | The number of packets received incorrectly having a CRC error and a noninteger number of octets. On a serial line, this is usually the result of noise or other transmission problems. |
| overrun | The number of times the serial receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the receiver's ability to handle the data. |
| ignored | The number of received packets ignored by the interface because the interface hardware ran low on internal buffers. These buffers are different than the system buffers mentioned previously in the buffer description. Broadcast storms and bursts of noise can cause the ignored count to be increased. |
| abort | An illegal sequence of one bits on a serial interface. This usually indicates a clocking problem between the serial interface and the data link equipment. |
| packets output | Total number of messages transmitted by the system. |
| bytes output | Total number of bytes, including data and MAC encapsulation, transmitted by the system. |
| underruns | Number of times that the transmitter has been running faster than the server can handle. This may never happen (be reported) on some interfaces. |
| output errors | The sum of all errors which prevented the final transmission of datagrams out of the interface being examined. Note that this may not balance with the sum of the enumerated output errors, as some datagrams may have more than one error, and others may have errors that do not fall into any of the specifically tabulated categories. |

| Field | Description |
|---------------------|---|
| interface resets | The number of times an interface has been completely reset. This can happen if packets queued for transmission were not sent within several seconds' time. On a serial line, this can be caused by a malfunctioning modem which is not supplying the transmit clock signal, or by a cable problem. If the system notices that the carrier detect line of a serial interface is up, but the line protocol is down, it periodically resets the interface in an effort to restart it. Interface resets can also occur when an interface is looped back down. |
| restarts | The number of times the controller was restarted because of errors. |
| carrier transitions | The number of times the carrier detect signal of a serial interface has changed state. Indicates modem or line problems if the carrier detect line is changing state often. |

Debugging the Serial Interface

Use the commands **debug serial-interface** and **debug serial-packet** to debug serial interface events. The EXEC commands are as follows:

```
debug serial-interface
debug serial-packet
```

Use **debug serial-packet** for detailed debugging information, and **debug serial-interface** for more general information.

Use the **undebug serial-interface** and **undebug serial-packets** to turn off messaging from these **debug** commands.

Ethernet Interface Support

Support for the Ethernet interface is supplied on one of Cisco Systems' Ethernet network interface cards:

- The Multiport Communications Interface (MCI) card provides up to two Ethernet connectors compatible with Ethernet Versions 1 and 2, and the IEEE 802.3 protocol.
- The Multiport Ethernet Controller (MEC) interface card provides two, four, or six high-speed Ethernet connectors compatible with Ethernet Versions 1 and 2, and the IEEE 802.3 protocol.

The Ethernet interface is also supported on the IGS network server.

Specifying an Ethernet Interface

To specify an Ethernet interface, use this configuration command:

```
interface ethernet unit
```

Specify the Ethernet interface connector number with the argument *unit*.

Follow this command with the interface subcommands for your particular protocol or application as described in the chapters in Part Five.

Example:

This command begins configuration on interface Ethernet 1.

```
interface ethernet 1
```

Ethernet Encapsulation Methods

The Ethernet interface supports a number of encapsulation methods. These methods are assigned by using the interface subcommand **encapsulation** followed by a keyword that defines the encapsulation method. The particular encapsulation method used depends on the protocol. Currently, there are three common Ethernet encapsulation methods:

- The standard Ethernet Version 2.0 encapsulation that uses a 16-bit protocol type code.
- The IEEE 802.3 encapsulation, where the type code becomes the frame length for the IEEE 802.2 LLC encapsulation (destination and source Service Access Points, and a control byte).
- The SNAP method, as specified in RFC 1042, which allows Ethernet protocols to run on IEEE 802.2 media.

The syntax of the **encapsulation** command follows:

```
encapsulation encapsulation-type
```

The *encapsulation-type* is one of the following three keywords:

- **arpa**—Standard Ethernet Version 2.0 encapsulation
- **iso1**—IEEE 802.3 encapsulation
- **snap**—IEEE 802.2 Ethernet media

Example:

These commands enable standard Ethernet Version 2.0 encapsulation on interface Ethernet 0.

```
interface ethernet 0  
encapsulation arpa
```

Maintaining the Ethernet Interface

Use the command **clear interface** to reset the hardware logic on an interface. Enter this command at the EXEC prompt:

```
clear interface ethernet unit
```

The argument *unit* specifies the Ethernet port number.

Note: Under normal circumstances, you do not need to clear the hardware logic on interfaces.

Monitoring the Ethernet Interface

Use the command **show interfaces ethernet** to display information about the Ethernet interface. Enter this command at the EXEC prompt:

```
show interfaces ethernet [unit]
```

The argument *unit* is the interface unit number. If you do not provide values for the *unit* argument, the command will display statistics for all the network interfaces.

Sample output of this command is provided on the following page. Table 1-3 describes the fields seen.

```
Ethernet 0 is up, line protocol is up
  Hardware is MCI Ethernet, address is aa00.0400.0134 (bia
0000.0c00.4369)
  Internet address is 131.108.1.1, subnet mask is 255.255.255.0
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, PROBE, ARP Timeout 4:00:00
  Last input 0:00:00, output 0:00:00, output hang never
  Output queue 0/40, 0 drops; input queue 0/75, 2 drops
  Five minute input rate 61000 bits/sec, 4 packets/sec
  Five minute output rate 1000 bits/sec, 2 packets/sec
    2295197 packets input, 305539992 bytes, 0 no buffer
    Received 1925500 broadcasts, 0 runts, 0 giants
    3 input errors, 3 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
    3594664 packets output, 436549843 bytes, 0 underruns
    8 output errors, 1790 collisions, 10 interface resets, 0 restarts
```

Table 1-3 Show Ethernet Interface Field Descriptions

| Field | Description |
|--|---|
| Ethernet ... is up ...is administratively down | Tells whether the interface hardware is currently active and if it's been taken down by an administrator. |
| line protocol is {up down administratively down} | Tells whether the software processes that handle the line protocol believe the interface is usable (are keepalives successful?). |
| Hardware | Specifies the hardware type (for example, MCI Ethernet, cBus Ethernet) and address. |
| Internet address | Lists the Internet address followed by subnet mask. |
| MTU | Maximum Transmission Unit of the interface. |
| BW | Bandwidth of the interface in kilobits per second. |
| DLY | Delay of the interface in microseconds. |
| rely | Reliability of the interface as a fraction of 255 (255/255 is 100% reliability), calculated as an exponential average over five minutes. |
| load | Load on the interface as a fraction of 255 (255/255 is completely saturated), calculated as an exponential average over five minutes. |
| Encapsulation | Encapsulation method assigned to interface. |
| ARP type: | Type of Address Resolution Protocol assigned. |
| loopback | Tells whether loopback was set or not. |
| output | Number of hours, minutes, and seconds (or never) since the last packet was successfully transmitted by the interface. Useful for knowing when a dead interface failed. |
| output hang | The number of hours, minutes, and seconds (or never) since the interface was last reset because of a transmission that took too long. When the number of hours in any of the "last" fields exceeds 24 hours, the number of days and hours is printed. If that field overflows, asterisks are printed. |
| Last Clearing | The time at which the counters that measure cumulative statistics (such as number of bytes transmitted and received) shown in this report were last reset to zero. Note that variables that might affect routing (for example, load and reliability) are not cleared when the counters are cleared. |
| Output queue, Input queue, drops | Number of packets in output and input queues. Each number is followed by a slash, the maximum size of the queue, and the number of packets dropped due to a full queue. |
| Five minute input rate, Five minute output rate | The average number of bytes and packets transmitted per second in the last five minutes. |

| Field | Description |
|-----------------------|--|
| packets input | The total number of error-free packets received by the system. |
| Received ..broadcasts | The total number of broadcast or multicast packets received by the interface. |
| runts | The number of packets which are discarded because they are smaller than the medium's minimum packet size. For instance, any Ethernet packet which is less than 64 bytes is considered a runt. |
| giants | The number of packets which are discarded because they exceed the medium's maximum packet size. For example, any Ethernet packet which is greater than 1518 bytes is considered a giant. |
| input error | Includes runts, giants, no buffer, CRC, frame, overrun, and ignored counts. Other input-related errors can also cause the input errors count to be increased, and some datagrams may have more than one error; therefore, this sum may not balance with the sum of enumerated input error counts. |
| CRC | The Cyclic Redundancy Checksum generated by the originating LAN station or far-end device does not match the checksum calculated from the data received. On a LAN, this usually indicates noise or transmission problems on the LAN interface or the LAN bus itself. A high number of CRCs is usually the result of collisions or a station transmitting bad data. |
| frame | The number of packets received incorrectly having a CRC error and a noninteger number of octets. On a LAN, this is usually the result of collisions or a malfunctioning Ethernet device. |
| overrun | The number of times the serial receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the receiver's ability to handle the data. |
| ignored | The number of received packets ignored by the interface because the interface hardware ran low on internal buffers. These buffers are different than the system buffers mentioned previously in the buffer description. Broadcast storms and bursts of noise can cause the ignored count to be increased. |
| packets output | Total number of messages transmitted by the system. |
| bytes | Total number of bytes, including data and MAC encapsulation, transmitted by the system. |
| underruns | Number of times that the transmitter has been running faster than the router can handle. This may never happen (be reported) on some interfaces. |

| Field | Description |
|------------------|--|
| output errors | The sum of all errors which prevented the final transmission of datagrams out of the interface being examined. Note that this may not balance with the sum of the enumerated output errors, as some datagrams may have more than one error, and others may have errors that do not fall into any of the specifically tabulated categories. |
| collisions | The number of messages retransmitted due to an Ethernet collision. This is usually the result of an over-extended LAN (Ethernet or transceiver cable too long, more than two repeaters between stations, or too many cascaded multiport transceivers). A packet that collides is counted only once in output packets. |
| interface resets | The number of times an interface has been completely reset. This can happen if packets queued for transmission were not sent within several seconds time. Interface resets can also occur when an interface is looped back or shut down. |
| restarts | The number of times a Type 2 Ethernet controller was restarted because of errors. |

Debugging the Ethernet Interface

Use the command **debug broadcast** to debug MAC broadcast packets. Enter this command at the EXEC prompt.

debug broadcast

Use the **undebug broadcast** command to turn off messaging.

Use the command **debug packet** to enable a log of packets that the network is unable to classify. Examples of this are packets with unknown link type, or IP packets with an unrecognized protocol field. Enter this command at the EXEC prompt.

debug packet

Use the **undebug packet** command to turn off messaging.

Token Ring Interface Support

Support for the Token Ring interface is supplied on one of Cisco Systems' Token Ring network interface cards:

- The Cisco Systems Token Ring Card (CSC-R) provides interconnection of Cisco network servers to IEEE 802.5 and IBM-compatible Token Ring media.
- The Cisco Systems 4/16 Mbps Token Ring Card (CSC-R16) provides interconnection of Cisco network servers to IEEE 802.5 and IBM-compatible Token Ring media at speeds of 16 or 4 megabits per second.

The Cisco Token Ring interface supports both routing (Level 3 switching) and source-route bridging (Level 2 switching).

Specifying a Token Ring Interface

To configure a Token Ring interface, use this configuration command:

```
interface tokenring unit
```

Specify *the card number* with the argument *unit*.

Follow this command with the interface subcommands for your particular protocol or application as described in the chapters in Part Five.

Example:

This command begins configuration on the first Token Ring interface.

```
interface tokenring 0
```

Configuring Ring Speed for IGS/TR

The Token Ring interface on the IGS/TR can run at either 4 or 16 Mbps. This speed is software selectable. The IGS/TR does not default to any particular ring speed. This speed must be provided the first time the IGS/TR is put to use.



Caution: Configuring a ring speed that is wrong or incompatible with the connected Token Ring will cause the ring to beacon, which effectively takes the ring down and makes it nonoperational.

Use the **ring-speed** interface subcommand to set ring speed for an IGS/TR Token Ring interface. The command syntax follows:

```
ring-speed speed
```

The argument *speed* can be either **4** or **16**. When specified as **4**, ring speed is set for 4 Mbps operation; when specified to **16**, ring speed is set for 16 Mbps operation. The default is **16**.

Example:

The following commands set an IGS/TR Token Ring interface ring speed to 4 Mbps.

```
interface tokenring 0
ring-speed 4
```

Token Ring Encapsulation Methods

Cisco's Token Ring interface, by default, uses the SNAP encapsulation format defined in RFC 1042. It is not necessary to define an encapsulation method for this interface.

Maintaining the Token Ring Interface

Use the command **clear interface** to reset the hardware logic on an interface. Enter this command at the EXEC prompt:

clear interface token ring *unit*

The arguments *unit* specifies the Token Ring card number.

Note: Under normal circumstances, you do not need to clear the hardware logic on interfaces.

To maintain the Routing Information Field (RIF) cache for protocol translators with Token Ring interfaces, use the **clear rif-cache** command. The command syntax is:

clear rif-cache

This command clears all entries from the RIF cache. It applies only to Token Ring interfaces.

Monitoring the Token Ring Interface

Use the command **show interface** to display information about the Token Ring interface and the state of source bridging. Enter this command at the EXEC prompt:

show interfaces tokenring [*unit*]

The argument *unit* is the interface unit number.

Sample output of this command is provided below. Table 1-4 describes the fields seen.

```

TokenRing 0 is up, line protocol is up
  Hardware is 16/4 Token Ring, address is 5500.2000.dc27 (bia
0000.3000.072b)
  Internet address is 150.136.230.203, subnet mask is 255.255.255.0
  MTU 8136 bytes, BW 16000 Kbit, DLY 630 usec, rely 255/255, load 1/255
  Encapsulation SNAP, loopback not set, keepalive set (10 sec)
  ARP type: SNAP, ARP Timeout 4:00:00
  Ring speed: 16 Mbps
  Single ring node, Source Route Bridge capable
  Group Address: 0x00000000, Functional Address: 0x60840000
  Last input 0:00:01, output 0:00:01, output hang never
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  Five minute input rate 0 bits/sec, 0 packets/sec
  Five minute output rate 0 bits/sec, 0 packets/sec
    16339 packets input, 1496515 bytes, 0 no buffer
    Received 9895 broadcasts, 0 runts, 0 giants
    0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
    32648 packets output, 9738303 bytes, 0 underruns
    0 output errors, 0 collisions, 2 interface resets, 0 restarts
    5 transitions

```

Table 1-4 Show Token Ring Interface Field Descriptions

| Field | Description |
|--|--|
| Token Ring is up down | The interface is currently active and inserted into ring (up) or inactive and not inserted (down). |
| Token Ring is Reset | Hardware error has occurred. |
| Token Ring is Initializing | Hardware is up, in the process of inserting the ring. |
| Token Ring is Administratively Down | Hardware has been taken down by an administrator. |
| line protocol is {up down administratively down} | Tells whether the software processes that handle the line protocol believe the interface is usable (are keepalives successful?). |
| Hardware | Specifies the hardware type (Token Ring or 16/4 Token Ring) and provides the address. |
| Internet address | Lists the Internet address followed by subnet mask. |
| MTU | Maximum Transmission Unit of the interface. |
| BW | Bandwidth of the interface in kilobits per second. |
| DLY | Delay of the interface in microseconds. |
| rely | Reliability of the interface as a fraction of 255 (255/255 is 100% reliability), calculated as an exponential average over five minutes. |
| load | Load on the interface as a fraction of 255 (255/255 is completely saturated), calculated as an exponential average over five minutes. |
| Encapsulation | Encapsulation method assigned to interface. |

| Field | Description |
|--|--|
| loopback | Tells whether loopback is set or not. |
| keepalive | Tells whether keepalives are set or not. |
| ARP type: | Type of Address Resolution Protocol assigned. |
| Ring speed: | Speed of Token Ring — 4 or 16 Mbps. |
| {Single ring multiring node} | Indicates whether a node is enabled to collect and use source routing information (RIF) for routable Token Ring protocols. |
| Group Address: | The interface's group address, if any. The group address is a multicast address; any number of interfaces on the ring may share the same group address. Each interface may have at most one group address. |
| Last input | The number of hours, minutes, and seconds since the last packet was successfully received by an interface. Useful for knowing when a dead interface failed. |
| output hang | The number of hours, minutes, and seconds (or never) since the interface was last reset because of a transmission that took too long. When the number of hours in any of the "last" fields exceeds 24 hours, the number of days and hours is printed. If that field overflows, asterisks are printed. |
| Output queue, Input Queue, drops | Number of packets in output and input queues. Each number is followed by a slash, the maximum size of the queue, and the number of packets dropped due to a full queue. |
| Five minute input rate, Five minute output rate | The average number of bytes and packets transmitted per second in the last five minutes. |
| packets input | The total number of error-free packets received by the system. |
| broadcasts | The total number of broadcast or multicast packets received by the interface. |
| runts | The number of packets which are discarded because they are smaller than the medium's minimum packet size. |
| giants | The number of packets which are discarded because they exceed the medium's maximum packet size. |
| CRC | The Cyclic Redundancy Checksum generated by the originating LAN station or far-end device does not match the checksum calculated from the data received. On a LAN, this usually indicates noise or transmission problems on the LAN interface or the LAN bus itself. A high number of CRCs is usually the result of a station transmitting bad data. |
| frame | The number of packets received incorrectly having a CRC error and a noninteger number of octets. |

| Field | Description |
|------------------|--|
| overrun | The number of times the serial receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the receiver's ability to handle the data. |
| ignored | The number of received packets ignored by the interface because the interface hardware ran low on internal buffers. These buffers are different than the system buffers mentioned previously in the buffer description. Broadcast storms and bursts of noise can cause the ignored count to be increased. |
| packets output | Total number of messages transmitted by the system. |
| bytes output | Total number of bytes, including data and MAC encapsulation, transmitted by the system. |
| underruns | Number of times that the far-end transmitter has been running faster than the near-end server's receiver can handle. This may never happen (be reported) on some interfaces. |
| output errors | The sum of all errors which prevented the final transmission of datagrams out of the interface being examined. Note that this may not balance with the sum of the enumerated output errors, as some datagrams may have more than one error, and others may have errors that do not fall into any of the specifically tabulated categories. |
| collisions | Since a Token Ring cannot have collisions, this statistic is nonzero only if an unusual event occurred when frames were being queued or dequeued by the system software. |
| interface resets | The number of times an interface has been reset. The interface may be reset by the administrator or automatically when an internal error occurs. |
| restarts | Should always be zero for Token Ring interfaces. |
| transitions | The number of times the ring made a transition from up to down, or vice versa. A large number of transitions indicates a problem with the ring or the interface. |

Debugging the Token Ring Interface

Use the EXEC commands described in this section to troubleshoot the Token Ring interface.

Use the **debug rif** command to enable logging of route information.

debug rif

The command enables logging of information about the route information fields (RIF) in Token Ring packets.

Use the **debug token-event** command to enable logging of Token Ring events.

debug token-event

This command provides a display of low-volume output.

Use the EXEC command **debug token-ring** to display messages about the Token Ring interface activity. This command reports several lines of information for each packet sent or received and is intended for low traffic, detailed debugging.

debug token-ring

The Token Ring interface records detailed information regarding the current state of the ring. These messages are only displayed when **debug token-event** is enabled.

Enter the **undebug** command with the appropriate keyword to turn off the messages.

The last ring status message is displayed in the EXEC command **show interfaces** display for a Token Ring interface. Table 1-5 describes the messages displayed by this command.

Table 1-5 Debug Token Ring Messages

| Message | Description |
|------------------|---|
| Signal Loss | The controller detected loss of signal on the interface. Several situations can cause this to happen, but the most likely is that another station has just inserted, causing a disruption in service that is reported as signal loss. |
| Hard Error | This error indicates a significant problem that is preventing transmission of data. There may be a break in the physical cabling or an inserted interface may have died. This message is displayed when the interface is either transmitting or receiving beacon frames. |
| Soft Error | The interface has detected an aberration on the ring and is transmitting a Report Error MAC frame. These frames are used to report the following types of errors: <ul style="list-style-type: none">• Line Error (code violation, token code violation, CRC violation)• Burst Error• MAC AC Set Error• Lost Frame Error• Frame Copied• Receiver Congestion• Token Error These errors are described more fully in the IEEE 802.5 standard. |
| Ring Beacon | The interface is transmitting beacon frames onto the ring. Something is wrong with the ring. |
| Wire Fault | The interface has detected an open or short circuit in the lobe data path. The data path starts at the edge of the chipset, and includes the Token Ring transition cable and any other cabling connection on the Multistation Access Unit. |
| HW Removal | The interface has detected an internal hardware error and has removed itself from the ring. |
| Remote Removal | The interface received a Remove Ring Station MAC frame from another station on the ring. The interface has removed itself from the ring. |
| Counter Overflow | Indicates an internal counter is close to reaching its maximum value. The Token Ring monitor firmware automatically reads and clears this condition. |
| Only Station | The interface has detected that it is the only interface connected and inserted on the ring. |
| Ring Recovery | The interface is either transmitting or receiving Claim Token MAC frames. This condition is cleared when an Active Monitor has been determined and it transmits a Ring Purge MAC frame. |

Configuring the Point-to-Point Protocol

The Point-to-Point Protocol (PPP), described in RFC 1134, is designed as a method of encapsulating Internet Protocol (IP) datagrams and other Network Layer protocol information over point-to-point links. The document “Point-to-Point Initial Configuration Options” defines the set of options that are negotiated during startup.

Cisco Systems’ current implementation of PPP supports no configuration options. The software sends no options and any proposed options are rejected.

Of the possible upper layer protocols, only IP is supported at this time. Thus, the only upper-level protocol that can be sent or received over a point-to-point link using PPP encapsulation is IP. Refer to RFC 1134 for definitions of the codes and protocol states.

The Point-to-Point Protocol is enabled on an interface using the **encapsulation** interface subcommand followed by the **ppp** keyword.

encapsulation ppp

Example:

These commands enable PPP encapsulation on serial interface zero.

```
interface serial 0
encapsulation ppp
```

Configuring the Null Interface

Cisco provides support for a null interface. This pseudo-interface functions similarly to the null devices available on most operating systems. This interface is always up and can never forward or receive traffic; encapsulation always fails.

The null interface provides an alternative method of filtering traffic. The overhead involved with using access lists can be avoided by directing undesired network traffic to the null interface.

To specify the null interface, specify “null 0” (or “null0”) as the interface name and unit. The null interface may be used in any command that has an interface type as a parameter.

Example:

This command configures a null interface for IP route 127.0.0.0.

```
ip route 127.0.0.0 null 0
```

Interface Support Subcommand Summary

Following are alphabetically arranged summaries of the interface subcommands for interface support.

[no] description *name-string*

Adds a descriptive name to an interface. The argument *name-string* is a comment to be put in the configuration.

encapsulation *encapsulation-type*

Assigns encapsulation method. The *encapsulation-type* argument is a keyword that identifies one of the following serial encapsulation types that Cisco Systems' software supports:

- **arpa**—Ethernet version 2.0 encapsulation
- **bfex25**—Blacker Front End Encryption X.25 operation
- **ddnx25-dce**—DDN X.25 DCE operation
- **ddnx25**—DDN X.25 DTE operation
- **frame-relay**—Frame Relay
- **hdh**—HDH Protocol
- **hdlc**—HDLC Protocol
- **iso1**—IEEE 802.3 encapsulation
- **lapb-dce**—X.25 LAPB DCE operation
- **lapb**—X.25 LAPB DTE operation
- **multi-lapb-dce**—X.25 LAPB multiprotocol DCE operation
- **multi-lapb**—X.25 LAPB multiprotocol DTE operation
- **ppp**—Point-to-Point Protocol (PPP)
- **snap**—IEEE 802.2 Ethernet media
- **smds**—SMDS service
- **x25-dce**—X.25 DCE operation
- **x25**—X.25 DTE operation

interface *type unit*

Specifies a serial interface. The argument *type* specifies the interface type—**serial**, **ethernet**, or **tokenring**—and the argument *unit* specifies the interface number or card number.

ring-speed *speed*

Sets operational ring speed for interface.

The argument *speed* can be either **4** or **16**. When specified as **4**, ring speed is set for 4 Mbps operation; when specified to **16**, ring speed is set for 16 Mbps operation. The default is **16**.

[no] shutdown

Disables and enables an interface.

Interface Support EXEC Command Summary

Following is an alphabetically arranged summary of the EXEC interface support commands.

clear counters [*type unit*]

Resets all interface counters listed in **show interface** statistics. The arguments *type* and *unit* specify the interface type and unit or card number (such as, ethernet 0, serial 0, or tokenring 0).

clear interface *type unit*

Resets the hardware logic on an interface. The arguments *type* and *unit* specify the interface type and unit or card number (such as, ethernet 0, serial 0, or tokenring 0).

clear rif-cache

Maintains the Routing Information Field (RIF) cache for protocol translators with a Token Ring interface. This command clears all entries from the RIF cache. It applies only to protocol translators with Token Ring interfaces.

[un]debug broadcast

Enables you to log all Level 2 (MAC) broadcast packets received. This information is useful for finding the source of a broadcast storm.

[un]debug packet

Enables logging of packets that the network server is unable to classify. Examples of this are packets with an unknown Ethernet link type, or IP packets with an unrecognized protocol field.

[un]debug rif

Enables logging of route information about the route information fields (RIF) in Token Ring packets.

[un]debug serial-interface

Enables general logging of serial-interface events for network servers equipped with serial network interfaces.

[un]debug serial-packet

Enables detailed logging of serial-interface events for network servers equipped with serial network interfaces.

[un]debug token-event

Enables logging of Token Ring events and provides a display of low-volume output.

[un]debug token-ring

Enables logging of Token Ring interface activity. This command reports several lines of information for each packet sent or received and is intended for low traffic, detailed debugging.

show controllers {serial | token | mci}

Displays current internal status information for different interface cards.

show interfaces [type unit]

Displays statistics for the network interfaces on the network server. The optional argument *type* can be one of the following: **ethernet**, **serial**, or **tokenring**. The argument *unit* specifies the interface unit or card number.

