Chapter 1
Configuring the Interfaces

1

This chapter contains information on enabling, shutting down, and displaying messages pertinent to interface configuration and operation. Also discussed in this chapter are the procedures and command descriptions for configuring and maintaining the following interface components of the router:

- Serial interfaces
- Ethernet interfaces
- Token Ring interfaces
- Fiber Distributed Data Interface (FDDI)
- High-Speed Serial Interface (HSSI)
- UltraNet interface

You will also find information about configuring the serial Dial Backup feature, and how to configure a 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; the **configure** command will place the system into the configuration command collection mode. 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 appropriate Cisco Hardware Installation and Reference publication.

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

Enter the **interface** global configuration command in configuration mode to identify a specific network interface (for example, a serial port, Ethernet port, or a Token Ring port). By entering this command you begin the configuration *subcommand* collection mode for the specified interface.

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

interface *type unit*

The argument *type* identifies the interface type; 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 subcommand collection mode for serial connector 0 (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 subcommand collection mode, you enter the interface subcommands for your particular routing or bridging protocol. The interface configuration subcommand 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 descriptive text 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 for. The description will appear in the output of the following commands: **show configuration**, **write terminal**, and **show interfaces**. The **no** version removes the *name-string*.

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, as well as prevents the transmission of all the packets. The command also marks the interface as unavailable, which is communicated to other network servers through all dynamic routing protocols. The interface will not be mentioned in any routing updates. On serial interfaces, this command causes the DTR signal to be dropped. On FDDI interfaces, this command causes the optical bypass switch, if present, to go into bypass mode. On Token Ring interfaces, this command causes the interface to de-insert 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:

This example turns off the interface Ethernet 0:

```
interface ethernet 0
shutdown
```

This example turns 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 [*interface-name*]

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 interfaces** 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 follow.

Displaying the System Configuration

The **show version** command displays the configuration of the system hardware, the software version, the names and sources of configuration files, and the boot images. Enter this command at the EXEC prompt:

show version

The following shows sample output from this command:

```
GS Software, Version 9.0(1)
Copyright (c) 1986-1992 by cisco Systems, Inc.
Compiled Fri 14-Feb-92 12:37

System Bootstrap, Version 4.3

thor uptime is 2 days, 10 hours, 0 minutes
System restarted by reload
System image file is unknown, booted via tftp from 131.108.13.111
Host configuration file is "thor-boots", booted via tftp from
131.108.13.111
Network configuration file is "network-confg", booted via tftp from
131.108.13.111

CSC3 (68020) processor with 4096K bytes of memory.
X.25 software.
Bridging software.
1 MCI controller (2 Ethernet, 2 Serial).
2 Ethernet/IEEE 802.3 interface.
2 Serial network interface.
32K bytes of non-volatile configuration memory.
Configuration register is 0x0
```

In the output, the first line is the bootstrap version string. The second through fourth lines list information about the system software; the version number is on the second line. Always specify the complete version number when reporting a possible software problem. In the sample output, the version number is 9.0, initial release.

The fifth line shows the system name and *uptime*, or the amount of time the system has been up and running. The sixth line provides a log of how the system was last booted, both as a result of normal system startup and of system error. For example, this line may be displayed to indicate a bus error that is generally the result of an attempt to access a nonexistent address:

```
System restarted by bus error at PC0XC4CA address 0X210C0C0
```

If the software was booted over the network, the seventh line shows the Internet address of the boot host. If the software was loaded from onboard ROM, this line reads `running default software`. The eighth and ninth lines identify the names and sources of the host and network configuration files.

The remaining lines of output show the hardware configuration and any non-standard software options. The configuration register contents are displayed in hexadecimal notation.

Note: The output of the **show version** EXEC command can also provide certain messages, such as bus error messages. If such error messages appear, report the complete text of this message to your Cisco technical support specialist.

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**|**cbus**|**fddi**}

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

- **serial**—For the Serial Interface Card

Note: Although the **show controllers serial** command continues to be supported, it works a bit differently on the IGS and CSC-based systems. On the IGS, the command requires explicitly specifying an interface number (such as **show controllers serial 0**). On systems with CSC/3 and CSC/4 controllers, using this command shows information about the CSC-S cards (if present).

- **token**—For the CSC-R, CSC-1R, CSC-2R and CSC-R16 (or CSC-R16M) Token Ring Interface Cards
- **mci**—For the Multiport Communications Interface Card
- **cbus**—For the cBus Controller Card
- **fddi**—For the FDDI Controller 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
Card type and number	Unit type and number card (varies depending on card)
controller type	Version number of the card
microcode version	Version number of the card's internal software (in read-only memory)
main memory cache memory	Amount of main and cache memory on the card
system TX	Number of buffers that hold packets to be transmitted buffers
Restarts line down hung output controller error	Count of restarts due to the following conditions: Communication line down Output unable to transmit 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

Note: The interface type is only queried at startup. If the hardware changes *subsequent* to initial startup, then the wrong type is reported. This has *no* adverse effect on the operation of the software. For instance, if a DCE cable is connected to a dual-mode V.35 applique after the unit has been booted, then the display presented for **show interfaces** incorrectly reports attachment to a DTE device although the software recognizes the DCE interface and berhaves accordingly.

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**, **tokenring**, **fddi**, **hssi**, or **ultranet**. 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,” “Monitoring the Token Ring Interface,” “Monitoring the FDDI,” “Monitoring the HSSI,” and “Monitoring the UltraNet 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.
- 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.

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 routing or bridging interface subcommands for your particular protocol or application, as described later in this manual.

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 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)
- Synchronous Data Link Control (SDLC)
- Switched Multimegabit Data Services (SMDS)
- Cisco Serial Tunnel (STUN)
- X.25-based encapsulations

With the exception of the PPP and HDLC encapsulations, these serial encapsulation methods are described in the chapter "Configuring Packet-Switched Software." The HDLC serial encapsulation method is described in this section. PPP serial encapsulation is described later in this chapter, in "Configuring the Point-to-Point Protocol."

Change the encapsulation method 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**—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)
- **sdlc-primary**—IBM serial SNA

- **smds**—SMDS service
- **stun**—STUN protocol function
- **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 reinstalled 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 *interface-name*

The arguments *type* and *unit* specify a particular interface type and its unit 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

In general, use the command **show interfaces** to display information about the serial interface. Enter this command at the EXEC prompt:

show interfaces serial [*type unit*]

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

Sample output of this command for an HDLC synchronous serial interface 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 Interfaces Serial 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.

Field	Description
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 bits 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 that are discarded because they are smaller than the medium's minimum packet size.
giants	The number of packets that 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. 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.

Field	Description
underruns	Number of times that the transmitter has been running faster than the router can handle. This may never be reported on some interfaces.
output errors	The sum of all errors that 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	Number of messages retransmitted due to an Ethernet collision. This usually is the result of an overextended 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. On a serial line, this can be caused by a malfunctioning modem that 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 or shut 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 (CSC-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 (CSC-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 routing or bridging interface subcommands for your particular protocol or application as described later in this manual.

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 *interface-name*

The arguments *type* and *unit* specify a particular interface type and its unit number. In this case, the argument *type* is **ethernet**.

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

Monitoring the Ethernet Interface

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

show interfaces [*type unit*]

Where *type* is the **ethernet** keyword and *unit* is the interface unit number. If you do not provide values for the parameters *type* and *unit*, 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 Interfaces Ethernet Field Descriptions

Field	Description
Ethernet ... is up ...is administratively down	Tells whether the interface hardware is currently active 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 believe the interface is usable (are keepalives successful?).
Hardware	Specifies the hardware type (for example, MCI Ethernet, SCI, cBus Ethernet) and address.
Internet address	Lists the Internet address followed by subnet mask and then the 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.
ARP type:	Type of Address Resolution Protocol assigned.
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	Number of hours, minutes, and seconds since the last packet was successfully transmitted by the interface. Useful for knowing when a dead interface failed.

Field	Description
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 bits and packets transmitted per second in the last five minutes.
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 that are discarded because they are smaller than the medium's minimum packet size. For instance, any Ethernet packet that is less than 64 bytes is considered a runt.
giants	The number of packets that are discarded because they exceed the medium's maximum packet size. For example, any Ethernet packet that is greater than 1,518 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.

Field	Description
overrun	The number of times the 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 be reported on some interfaces.
output errors	The sum of all errors that 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 overextended 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. On a serial line, this can be caused by a malfunctioning modem that 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 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
undebug 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
undebug 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 single-port 4-Mbps Cisco Systems Token Ring card (CSC-R) provides interconnection of Cisco network servers to IEEE 802.5 and IBM-compatible Token Ring media.
- The three 4/16 Mbps Cisco Systems Token Ring cards provide interconnection of Cisco network servers to IEEE 802.5 and IBM-compatible Token Ring media at speeds of 16 or 4 megabits per second. Cisco's 4/16 Mbps cards are the CSC-R16 (or CSC-R16M), CSC-1R, and CSC-2R (dual Token Ring card).

The Cisco Token Ring interface supports both routing (Level 3 switching) and source-route bridging (Level 2 switching). The use of routing and bridging is on a per-protocol basis. For example, IP traffic could be routed while SNA traffic is bridged. The routing support interacts correctly with source-route bridges.

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 bridging interface subcommands for your particular protocol or applications as described later in this manual.

Example:

This command begins configuration on the first Token Ring interface.

```
interface tokenring 0
```

Note: If the system receives an indication of a cabling problem from a Token Ring interface, it puts that interface into a "reset" state, and does not attempt to restart the interface. Once you have replugged the cable into the MAU, type the command **clear interface tokenring** *unit*, where *unit* is the interface number. This command will restart the interface. The system functions in this manner because periodic attempts to restart the Token Ring interface have a drastic impact on the stability of protocol routing tables.

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 is:

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
```

Configuring Early Token Release

The CSC-R16 (or CSC-R16M), CSC-2R, and CSC-1R cards all support *early token release*, a method whereby these interfaces can release the token back onto the ring immediately after transmitting, rather than waiting for the frame to return. This can help to increase the total bandwidth of the Token Ring. The following interface subcommands control the use of this feature:

early-token-release
no early-token-release

By default, early token release is not enabled on the interface. The **no early-token-release** command disables this feature.

Example:

For example, to enable the use of early token release on your tokenring 1 interface, issue the following configuration commands:

```
interface tokenring 1
early-token-release
```

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 *interface-name*

The arguments *type* and *unit* specify a particular interface type and its unit number. In this case, the argument *type* is **tokenring**.

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

Monitoring the Token Ring Interface

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

show interfaces [*type unit*]

The argument *type* is the keyword **tokenring** and *unit* is the interface unit number. If you do not provide values for the parameters *type* and *unit*, the command will display statistics for all the network interfaces. 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 Interfaces Token Ring 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.
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.

Field	Description
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 bits 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 that are discarded because they are smaller than the medium's minimum packet size.
giants	The number of packets that 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.
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.

Field	Description
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 router's receiver can handle. This may never be reported on some interfaces.
output errors	The sum of all errors that 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 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
undebug token-ring

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

Enter the **undebug token-ring** and **undebug token-event** commands 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: • 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.

FDDI Support

FDDI is an ANSI-defined standard for timed 100-Mbps token-passing over fiber-optic cable. Support for the Fiber Distributed Data Interface (FDDI) is supplied on Cisco Systems' FDDI interface (CSC-FCI) card. Cisco's implementation of FDDI complies with Version 6.1 of the X3T9.5 FDDI specification, offering a Class A dual-attach interface that supports the fault-recovery methods of the Dual-Attach Stations (DASs).

An FDDI network consists of two counter token-passing fiber-optic rings. On most networks, the primary ring is used for data communication and the secondary ring is used as a hot standby. The FDDI standard sets a 200 kilometers total fiber length for multimode fiber, and 10 kilometers for single-mode fiber, both of which are supported by Cisco's FDDI interface controller. (The maximum circumference of the FDDI network is only half the specified kilometers because of the *wrapping*, or the looping back of the signal, that occurs during fault isolation.) The FDDI standard allows a maximum of 500 stations with a maximum distance between active stations of two kilometers. The FDDI frame can contain a minimum of 76 bytes and a maximum of 4500 bytes.

Cisco also provides support for some of the FDDI MIB variables as described in RFC 1285, "FDDI Management Information Base," January 1992 by Jeffrey D. Case of the University of Tennessee and SNMP Research, Inc. One such variable that Cisco supports is snmpFd-diSMTCFState.

Specifying an FDDI

To specify an FDDI, use the following configuration command:

interface fddi *unit*

Specify the interface number with the argument *unit*.

Follow this command with the routing or bridging interface subcommands for your particular protocol or application as described later in this manual.

FDDI Encapsulation Methods

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

Monitoring the FDDI

The EXEC command **show interfaces** displays information about the FDDI interface, and its use is recommended when configuring the interface.

What follows is a sample of a partial display of FDDI-specific data from the **show interfaces** command output. Table 1-6 describes the fields displayed.

```
Fddi 0 is up, line protocol is up
Hardware type is cBus Fddi, hardware address is AA00.0400.6510
Internet address is 131.111.21.6, subnet mask is 255.255.255.0
MTU 4470 bytes, BW 100000 Kbit, DLY 100 usec, rely 255/255, load 1/255
Encapsulation is SNAP, loopback is not set, keepalive is not set (10
sec.)
ARP type: SNAP
Phy-A state is active, neighbor is B, cmt signal bits 08/20C, status ALS
Brk 1, Con 1, Tra 0, Nxt 11, Sig 10, Join 1, Vfy 1, Act 1
Phy-B state is active, neighbor is A, cmt signal bits 20C/08, status ILS
Brk 1, Con 1, Tra 0, Nxt 11, Sig 10, Join 1, Vfy 1, Act 1
CFM is wrap A, token rotation 5000 usec, ring operational 0:01:42
Upstream neighbor 0800.2008.C52E, downstream neighbor 0800.2008.C52E
Last input 0:00:24, output 0:00:15, output hang never
  Output queue 0/25, 0 drops; input queue 0/75, 0 drops
  Five minute input rate 0 bits/sec, 0 packets/sec
  Five minute output rate 1 bits/sec, 0 packets/sec
  2725 packets input, 166225 bytes, 0 no buffer
  Received 2725 broadcasts, 0 runts, 0 giants
    2 input errors, 2 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
   5184 packets output, 316224 bytes
    0 output errors, 0 collisions, 1 interface resets, 0 restarts
   4 transitions
```

Table 1-6 Show Interfaces FDDI Field Descriptions

Field	Description
Fddi is {up |down} ...is administratively down	Tells whether the interface hardware is currently active and can transmit and receive, and if it has been taken down by an administrator.
line protocol is {up | down | administratively down}	Tells whether the interface hardware is currently active and can transmit and receive, or if it has been taken down by an administrator.
Hardware	Provides the hardware type followed by the hardware 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.
loopback	Tells whether loopback is set or not.
keepalive	Tells whether keepalives are set or not.
ARP type:	Type of Address Resolution Protocol assigned.

Field	Description
Phy-{A | B}	Lists the state the Physical A or Physical B connection is in; one of: off, active, trace, connect, next, signal, join, verify, or break.
neighbor	State of the neighbor: • A—Indicates that the CMT process has established a connection with its neighbor. The bits received during the CMT signaling process indicate that the neighbor is a Physical A type dual attachment station or concentrator that attaches to the primary ring IN and the secondary ring OUT when attaching to the dual ring. • S—Indicates that the CMT process has established a connection with its neighbor and that the bits received during the CMT signaling process indicate that the neighbor is one Physical type in a single attached station (SAS). • B—Indicates that the CMT process has established a connection with its neighbor and that the bits received during the CMT signaling process indicate that our neighbor is a Physical B dual attached station or concentrator that attaches to the secondary ring IN and the primary ring OUT when attaching to the dual ring. • M—Indicates that the CMT process has established a connection with its neighbor and that the bits received during the CMT signaling process indicate that the router's neighbor is a Physical M type concentrator that serves as a Master to a connected station or concentrator. • unk—Indicates that the Cisco network server has not completed the CMT process, and as a result, does not know about its neighbor. See the section "Setting Bit Control" for an explanation of the bit patterns.
cmt signal bits	Shows the transmitted/received CMT bits. The transmitted bits are 0x008 for a Physical A type and 0x20C for Physical B type. The number after the slash (/) is the received signal bits. If the connection is not active, the received bits are zero (0); see the line beginning Phy-B in the above display.

Field	Description
status	The status value displayed is the actual status on the fiber. The FDDI standard defines the following values: • LSU—Line State Unknown, the criteria for entering or remaining in any other line state have not been met. • NLS—Noise Line State is entered upon the occurrence of 16 potential noise events without satisfying the criteria for entry into another line state. • MLS—Master Line State is entered upon the reception of eight or nine consecutive HQ or QH symbol pairs. • ILS—Idle Line State is entered upon receipt of four or five idle symbols. • HLS—Halt Line State is entered upon the receipt of 16 or 17 consecutive H symbols. • QLS—Quiet Line State is entered upon the receipt of 16 or 17 consecutive Q symbols or when carrier detect goes low. • ALS—Active Line State is entered upon receipt of a JK symbol pair when carrier detect is high. • OVUF—Elasticity buffer Overflow/Underflow. The normal states for a connected Physical type is ILS or ALS. If the report displays the QLS status, this indicates that the fiber is disconnected from Physical B, or that it is not connected to another Physical type, or that the other station is not running.
Off	Indicates the CMT is not running on the Physical Sublayer. The state will be off if the interface has been shutdown or if the **cmt disconnect** command has been issued for Physical A or Physical B.
Brk	The Break State is the entry point in the start of a PCM connection.
Tra	The Trace State localizes a stuck beacon condition.
Con	The Connect State is used to synchronize the ends of the connection for the signaling sequence.
Nxt	The Next State separates the signaling performed in the Signal State, and transmits Protocol Data Units (PDUs) while MAC Local Loop is performed.
Sig	The Signal State is entered from the Next State when a bit is ready to be transmitted.
Join	The Join State is the first of three states in a unique sequence of transmitted symbol streams received as line states—the Halt Line State, Master Line State, and Idle Line State, or HLS-MLS-ILS—that leads to an active connection.
Vfy	The Verify State is the second state in the path to the Active State, and will not be reached by a connection that is not synchronized.

Field	Description
Act	The Active State indicates that the CMT process has established communications to its physical neighbor. The transition states are defined in the X3T9.5 specification, and you are referred to the specification for details about these states.
CFM is ...	Contains information about the current state of the MAC connection. The Configuration Management (CFM) state may be one of the following: • isolated—The MAC is not attached to any Physical type. • wrap A—The MAC is attached to Physical A. Data is received on Physical A and transmitted on Physical A. • wrap B—The MAC is attached to Physical B. Data is received on Physical B and transmitted on Physical B. • thru A—The MAC is attached to Physical A and B. Data is received on Physical A and transmitted on Physical B. This is the normal mode for a dual attachment station (DAS) with one MAC. The ring has been operational for 1 minute and 42 seconds.
token rotation	The token rotation value is the default or configured rotation value as determined by the **fddi token rotation-time** command. This value is used by all stations on the ring. The Cisco default is 5000 microseconds.
ring operational	When the ring is operational, the displayed value will be the negotiated token rotation time of all stations on the ring. Operational times are displayed by the number of hours:minutes:seconds the ring has been up. If the ring is not operational, the message ring not operational is displayed.
Upstream | downstream neighbor	Displays the canonical MAC address of outgoing upstream and downstream neighbors. If the interface is not up, then these values will be zero (0).
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 bits 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.
broadcasts	The total number of broadcast or multicast packets received by the interface.
runts	The number of packets that are discarded because they are smaller than the medium's minimum packet size.
giants	The number of packets that 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 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 FDDI, frame errors might result from a failing fiber or from hardware malfunctions.
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.
output errors	The sum of all errors that 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	Because an FDDI ring cannot have collisions, this statistic is always zero.
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.

Field	Description
restarts	Should always be zero for FDDI interfaces.
transitions	The number of times the ring made a transition from ring operational to ring nonoperational, or vice versa. A large number of transitions indicates a problem with the ring or the interface.

The received CMT bits are sometimes helpful if the Cisco network server is having problems establishing a connection to the remote Physical connection.

In the previous display, Physical A (`Phy-A`) has completed CMT with its neighbor. The state is active and the display indicates a Physical B type neighbor. The neighbor is determined from the received signal bits, as follows:

Bit Positions	9	8	7	6	5	4	3	2	1	0
Value Received	1	0	0	0	0	0	1	1	0	0

S1770

The received value equals 0x20C. Bit positions 1 and 2 (0 1) indicate a Physical B type connection.

The transition states displayed indicate that the CMT process is running and actively trying to establish a connection to the remote physical connection. The CMT process requires state transition with different signals being transmitted and received before moving on to the next state. The ten bits of CMT information are transmitted and received in the Signal State. Each state displays the number of times it was entered. In the above example, the Next State (`Nxt`) was entered 11 times.

Note: The display line showing transition states is not generated if the FDDI interface has been shut down or if the **cmt disconnect** command has been issued.

The CFM state is `wrap A` in the sample output because the Cisco network server has not completed CMT with its neighbor to connect to Physical B.

The display (or nondisplay) of the Cisco upstream and downstream neighbor does not affect the ability to route data. The determination of the upstream and downstream neighbors is dependent upon all stations on the ring running the same version of Station Management (SMT). Since the Cisco upstream neighbor is also its downstream neighbor in the sample supplied previously, there are only two stations in the ring, the Cisco network server and the router at address 0800.2008.C52E.

Debugging the FDDI

Use the following EXEC commands to monitor the state of the FDDI interface. (Note that each command has a corresponding **undebug** command that turns off the messages displayed by these commands.)

debug fddi-smt-packets

debug fddi-cmt-events

The **debug fddi-smt-packets** command enables logging of FDDI station management (SMT) packets, whereas the **debug fddi-cmt-events** command enables logging of FDDI Connection Management (CMT) transactions. See the X3T9.5 specification for more information about these packets and transactions.

FDDI Special Commands and Configurations

Using special FDDI interface subcommands, you can set the token rotation time, set the transmission valid timer, control the transmission time, set the bit control, and start and stopping FDDI. This section also describes FDDI dual homing—a built-in configuration capability of the Cisco FDDI software.

Setting Token Rotation Time

Use the **fddi token-rotation-time** interface subcommand to control ring scheduling during normal operation, and to detect and recover from serious ring error situations.

fddi token-rotation-time *microseconds*

The argument *microseconds* determines the token rotation time (TRT). The default value is 5000 microseconds. The FDDI standard restricts the allowed time to be greater than 4000 microseconds and less than 165000 microseconds.

As defined in the X3T9.5 specification, the value remaining in the TRT is loaded into the token holding timer (THT). Combining the values of these two timers provides the means to determine the amount of bandwidth available for subsequent transmissions.

Example:

These commands set the rotation time to 24000 microseconds.

```
interface fddi 0
fddi token-rotation-time 24000
```

Setting the Transmission Valid Timer

Use the **fddi valid-transmission-time** interface subcommand to recover from a transient ring error.

fddi valid-transmission-time *microseconds*

The argument *microseconds* sets the transmission valid timer (TVX) interval. The default valid transmission timer value is 2500 microseconds.

Example:

These commands change the transmission timer interval to 3000 microseconds.

```
interface fddi 0
fddi valid-transmission-time 3000
```

Controlling Transmission Time

Use the **fddi tl-min-time** interface subcommand to control the FDDI TL_MIN time (the minimum time to transmit a Physical Sublayer, or PHY line state before advancing to the next Physical Connection Management, or PCM state, as defined by the X3T9.5 specification).

fddi tl-min-time *microseconds*

The specification defines the argument *microseconds* to be a value of 30. This value is used during the Connection Management (CMT) phase to ensure that signals are maintained for at least the value of TL_MIN so the remote station can acquire the signal.

Note: Interoperability tests have shown that some implementations of the FDDI standard need more than 30 microseconds to sense a signal.

Example:

These commands change the TL_MIN time from 30 microseconds to 100 microseconds.

```
interface fddi 0
fddi tl-min-time 100
```

Setting Bit Control

Use the **fddi cmt-signal-bits** interface subcommand to control the information transmitted during the CMT signaling phase.

fddi cmt-signal-bits *signal-bits* **phy-a** | **phy-b**

The argument *signal-bits* is written as a hexadecimal number preceded by “0x”; for example, “0x208.” The keywords **phy-a** and **phy-b** select the Physical Sublayer (Physical A or Physical B station), for control of each fiber.

The FDDI standard defines nine bits of signaling information (*signal-bits*) that must be transmitted:

- **bit 0**—Escape bit. Reserved for future assignment by the FDDI standards committee.
- **bits 1 and 2**—Physical type, as defined in Table 1-7.

Table 1-7 FDDI Physical Type Bit Specifications

bit 2	bit 1	Physical Type
0	0	Physical A
1	0	Physical B
0	1	Physical S
1	1	Physical M

Bits 1 and 2 are transmitted (signaled), and each physical type determines if it is allowed to connect to a type at the other end.

- **bit 3**—Physical compatibility. Set if topology rules include the connection of a physical-to-physical type at the end of the connection.
- **bits 4 and 5**—Link Confidence test duration; set as defined in Table 1-8.

Table 1-8 FDDI Link Confidence Test Duration Bit Specification

bit 5	bit 4	Test Duration
0	0	Short test (default 50 milliseconds)
1	0	Medium test (default 500 milliseconds)
0	1	Long test (default 5 seconds)
1	1	Extended test (default 50 seconds)

- **bit 6**—Media Access Control (MAC) available for link confidence test.
- **bit 7**—Link confidence test failed.
- The setting of bit 7 indicates that the link confidence was failed by the Cisco end of the connection.
- **bit 8**—MAC for local loop.
- **bit 9**—MAC on physical output.

The default signal bits for the **phy-a** and **phy-b** keywords are as follows:

- **phy-a** is set to 0x008 (hexadecimal) or 00 0000 1000 (binary). Bits 1 and 2 are set to 00 to select Physical A. Bit 3 is set to 1 to indicate “accept any connection.”

- **phy-b** is set to 0x20c (hexadecimal) or 10 0000 1100 (binary). Bits 1 and 2 are set to 10 to select Physical B. Bit 3 is set to 1 to indicate “accept any connection.” Bit 9 is set to 1 to select MAC on output. The normal data flow on FDDI is input on Physical A and output on Physical B.

If the **phy-a** or **phy-b** keyword is not specified, then the signal bits apply to both physical connections.

Note: Use of the **fddi cmt-signal-bits** subcommand is *not* recommended. This subcommand has been helpful in resolving CMT implementation issues.

Starting and Stopping the FDDI

In normal operation, the FDDI interface is operational once the interface is connected and configured, and is turned off using the **shutdown** interface subcommand described in the section “Shutting Down and Restarting an Interface,” earlier in this chapter. The privileged EXEC commands **cmt connect** and **cmt disconnect** allow the operator to start and stop the processes that perform the Connection Management (CMT) function, and particularly, allows the ring on one fiber to be stopped.

The EXEC commands **cmt connect** and **cmt disconnect** are not needed in the normal operation of FDDI; these commands are mainly used in interoperability tests, and are entered at the EXEC prompt:

cmt connect [*interface-name* [**phy-a|phy-b**]]
cmt disconnect [*interface-name* [**phy-a|phy-b**]]

The optional argument *interface* specifies the particular FDDI interface. The optional keywords **phy-a** and **phy-b** specify a Physical Sublayer (A or B).

Examples—Starting FDDI:

The following commands demonstrate use of the **cmt connect** commands for starting the CMT processes on the FDDI ring.

This command starts all FDDI interfaces:

```
cmt connect
```

This command starts both fibers on the FDDI interface unit 0 (zero):

```
cmt connect fddi 0
```

This command starts only Physical Sublayer A on the FDDI interface unit 0 (zero):

```
cmt connect fddi 0 phy-a
```

Examples—Stopping FDDI:

The following commands demonstrates using the **cmt disconnect** command for stopping the CMT processes on the FDDI ring.

This command stops all FDDI interfaces:

```
cmt disconnect
```

This command stops FDDI interfaces unit 0:

```
cmt disconnect fddi 0
```

This command stops only Physical Sublayer A on the FDDI interface unit 0 (zero). This command causes the FDDI media to go into a wrapped state, so that the ring will be broken.

```
cmt disconnect fddi 0 phy-a
```

FDDI Dual-Homing Configuration

Configuration of the FDDI interface is not required for dual homing. The FDDI interface recognizes that it is attached to two M ports on the concentrators, and automatically supports dual homing.

Controlling Buffering of FDDI SMT Messages

Although most configuration for FDDI interfaces involves specific interface settings, a Cisco router's ability to buffer FDDI Station Management (SMT) messages is controlled *globally*. The following brief description outlines how you can control the buffering of SMT messages that are queued for processing.

Setting SMT Message Queue Size

Use the global configuration command **smt-queue-threshold** to set the maximum number of unprocessed Station Management (SMT) frames that will be held for processing. The command syntax is:

smt-queue-threshold *number*
no smt-queue-threshold

The argument *number* specifies the number of buffers used to store unprocessed SMT message that are to be queued for processing. Acceptable values are positive integers.

The default value for *number* is equal to the number of FDDI interfaces installed in the router.

This command helps ensure that Cisco routers keep track of FDDI *upstream* and *downstream* neighbors, particularly when a router includes more that one FDDI interface.

In FDDI, upstream and downstream neighbors are determined by transmitting and receiving SMT Neighbor Information Frames (NIFs).

The router can appear to lose track of neighbors when it receives an SMT frame and the queue currently contains an unprocessed frame. This occurs because the router discards incoming SMT frames if the queue is full. Discarding SMT NIF frames can cause the router to lose its upstream or downstream neighbor.

Note: Use this command carefully because the SMT buffer is charged to the inbound interface (input hold queue) until the frame is completely processed by the system. Setting this value to a high limit can impact buffer usage and the ability of the router to receive routable packets or routing updates.

The command **no smt-queue-threshold** restores the queue to the default.

Example:

The following example specifies that the SMT queue can hold ten messages. As SMT frames are processed by the system, the queue is decremented by one.

```
smt-queue-threshold 10
```

High-Speed Serial Interface (HSSI) Support

The Cisco High-Speed Serial Interface (HSSI) consists of two cards:

- CSC-HSCI controller card, which is cBus-resident
- CSC-HSA, which is a back-panel applique

The card provides a single full duplex synchronous serial interface capable of transmitting and receiving data at up to 52 megabits per second. The HSSI is a de facto industry standard providing connectivity to T3 (DS-3), E3, SMDS (at a DS-3 route), and other high-speed wide-area services through a DSU or Line Termination Unit.

The high-speed, full duplex synchronous serial interface is supported only on Cisco's modular network server products.

Specifying the HSSI

To specify the HSSI, use this configuration command:

interface hssi *unit*

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

Follow this command with the routing or bridging interface subcommands for your particular protocol or application as described in later in this manual.

The cBus cards can query the appliques to determine their types. However, they do so only at system start-up, so the appliques must be attached when the system is started. Issue a **show controllers cbus** command to determine how the HSSI card has identified them. The command will also show the capabilities of the card and report controller-related failures.

Example:

This command begins configuration on interface HSSI 0.

```
interface hssi 0
```

HSSI Encapsulation Methods

The HSSI supports the serial encapsulation methods described in the section "Serial Encapsulation Methods," earlier in this chapter.

Maintaining the HSSI

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

clear interface *interface-name*

The arguments *type* and *unit* specify a particular interface type and its unit number. In this case, the argument *type* is **hssi**.

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

Monitoring the HSSI

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

show interfaces [*type unit*]

Where *type* is the keyword **hssi** and *unit* is the interface unit number. If you do not provide values for the parameters *type* and *unit*, the command will display statistics for all the network interfaces.

Sample output of this command for Cisco's HSSI is provided below. Table 1-9 describes the fields seen.

```
HSSI 0 is up, line protocol is up
  Hardware is cBus HSSI
  Internet address is 150.136.67.190, subnet mask is 255.255.255.0
```

```
     MTU 4470 bytes, BW 45045 Kbit, DLY 20000 usec, rely 255/255, load 1/255
     Encapsulation HDLC, loopback not set, keepalive set (10 sec)
     Last input 0:00:03, 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
        0 packets input, 0 bytes, 0 no buffer
        Received 0 broadcasts, 0 runts, 0 giants
                 0 parity, 0 rx disabled
        0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
        17 packets output, 994 bytes, 0 underruns
        0 output errors, 0 applique, 4 interface resets, 0 restarts
        2 carrier transitions
```

Table 1-9 Show Interfaces HSSI Field Descriptions

Field	Description
HSSI 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 thinks the line is usable (are keepalives successful?
Hardware	Specifies the hardware type.
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.
loopback	Tells whether loopback is set, and type of loopback test.
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.

Field	Description
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 bits 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 that are discarded because they are smaller than the medium's minimum packet size.
giants	The number of packets that are discarded because they exceed the medium's maximum packet size.
parity	Report of the parity errors on the HSSI.
rx disabled	Indicates inability to get a buffer when accessing a packet.
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. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link. CRC errors are also reported when a far-end abort occurs, and when the idle flag pattern is corrupted. This makes it possible to get CRC errors even when there is no data traffic.
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.

Field	Description
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 router's receiver can handle. This may never happen (be reported) on some interfaces.
congestion drop	The number of messages discarded because the output queue on an interface grew too long. This can happen on a slow, congested serial link.
output errors	The sum of all errors that 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.
applique	Indicates an unrecoverable error has occurred on the HSA applique. The system then invokes an interface reset.
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 that 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 or shut 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 HSSI

Use the command **debug serial-interface** to debug HSSI events. Enter this command at the EXEC prompt:

debug serial-interface

Enter the **undebug serial-interface** to turn off messaging.

UltraNet Interface Support

The UltraNet Interface consists of two cards: the CSC-HSCI, which is a cBus resident card, and the CSC-ULA, which is a back-panel applique. The UltraNet Interface provides connectivity to the UltraNet product offered by Ultra Network Technologies.

Specifying an UltraNet Interface

To specify an UltraNet interface, use this configuration command:

interface ultranet *unit*

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

Follow this command with the appropriate routing or bridging interface subcommands for your particular protocol or application as described in later in this manual.

Example:

This command begins configuration on UltraNet interface 0:

```
interface ultranet 0
```

UltraNet Encapsulation Method

The UltraNet interface supports the UltraNet encapsulation only. Therefore, there is no encapsulation command for the interface. It will default to UltraNet encapsulation.

Maintaining the UltraNet Interface

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

clear interface *interface-name*

The arguments *type* and *unit* specify a particular interface type and its unit number. In this case, the argument *type* is **ultranet**.

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

Monitoring the UltraNet Interface

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

show interfaces [*type unit*]

The argument *type* is the keyword **ultranet** and *unit* is the interface unit number. If you do not provide values for the parameters *type* and *unit*, 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 1-10 describes the fields seen.

```
UltraNet 0 is up, line protocol is up
  Hardware is cBus UltraNet, address is 8/32
  Internet address is 150.136.68.190, subnet mask is 255.255.255.0
  MTU 3500 bytes, BW 125000 Kbit, DLY 1000 usec, rely 255/255, load 1/255
  Encapsulation ULTRANET, loopback not set, keepalive set (10 sec)
  ARP type: ULTRA
  Last input 0:00:09, output 0:00:09, 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
     6 packets input, 236 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants
              0 parity, 0 rx disabled
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     6 packets output, 236 bytes, 0 underruns
     0 output errors, 0 applique, 1 interface resets, 0 restarts
```

Table 1-10 Show Interfaces UltraNet Field Descriptions

Field	Description
UltraNet is {up |down} ...is administratively down	Tells whether the interface is currently active and if it has been taken down by an administrator.
line protocol is {up | down | administratively down}	Tells whether the processes that handle the line protocol are active.
Hardware	Specifies the hardware type.
Internet Address	Specifies 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.

Field	Description
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 bits 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.
parity	Report of the parity errors on the UltraNet interface.
rx disabled	Indicates inability to get a buffer when accessing a packet.

Field	Description
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. 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 router can handle. This may never happen (be reported) on some interfaces.
output errors	The sum of all errors that 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.
applique	Indicates an unrecoverable error has occurred on the ULA applique.

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. This can be caused by a malfunctioning modem that 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 or shut down.
restarts	The number of times the controller was restarted because of errors.

Statically Assigning UltraNet Addresses

Sometimes it is necessary to statically assign the UltraNet MAC layer address to the interface. This mechanism is needed if dynamic address assignment is not implemented on the Ultra Network Technologies interface on the network. Use the **ultranet address** interface subcommand to assign the MAC layer address of the interface:

ultranet address *ultranet-mac-address*

The argument *ultranet-mac-address* is the MAC address of the UltraNet service line.

Example:

These commands assign address *8/32* to the UltraNet interface.

```
interface ultranet 0
ultranet address 8/32
```

Configuring Dial Backup Service

The dial backup service provides protection against WAN down time by allowing you to configure a backup serial line via a circuit-switched connection.

To configure dial backup, associate a secondary serial interface as a backup to a primary serial interface. This feature requires that an external modem, CSU/DSU device, or ISDN terminal adapter (TA) attached to a circuit-switched service be connected on the secondary serial interface. The external device must be capable of responding to a DTR signal (DTR active) by auto-dialing a connection to a preconfigured remote site.

The dial backup software keeps the secondary line inactive (DTR inactive) until one of the following conditions is met:

- The primary line goes down.
- The transmitted traffic load on the primary line exceeds a defined limit.

These conditions are defined using the interface subcommands described later in this section.

When the software detects either a lost Carrier Detect signal from the primary line device, or that the line protocol is down, the software activates DTR on the secondary line. At that time, the modem, CSU/DSU, or ISDN Terminal Adapter (TA) must be set to dial the remote site. When that connection is made, the routing protocol defined for the serial line will continue the job of transmitting traffic over the dialup line.

You may also configure the dial backup feature to activate the secondary line based upon traffic load on the primary line.

The software monitors the traffic load and computes a five-minute moving average. If this average exceeds the value you set for the line, then the secondary line is activated, and depending upon how the line is configured, some or all of the traffic will flow onto the secondary dialup line.

You may also specify a value that defines when the secondary line should be disabled, and the amount of time the secondary line can take going up or down.

Configuring the Dial Backup Line

Use the **backup interface** interface subcommands to configure the serial interface as a secondary, or dial backup, line. The commands have this syntax:

backup interface *interface-name*
no backup interface *interface-name*

The argument *interface-name* specifies the serial port to be set as the secondary interface line.

Use the **no backup** interface command with the appropriate serial port designation to turn this feature off.

Example:

These commands set serial 1 as the backup line to serial 0.

```
interface serial 0
backup interface serial 1
```

Defining the Traffic Load Threshold

Use the **backup load** interface subcommands to set the traffic load thresholds. The commands have this syntax:

backup load {*enable-threshold*|**never**} {*disable-load*|**never**}
no backup load {*enable-threshold*|**never**} {*disable-load*|**never**}

Enter the arguments *enable-threshold* and *disable-load* using percentage numbers representing the load as a percentage of the primary line's available bandwidth.

When the transmitted or received load on the primary line is greater than the value assigned to the *enable-threshold* argument, the secondary line is enabled.

When the transmitted load on the primary line plus the transmitted load on the secondary line is less than the value entered for the *disable-load* argument, and the received load on the primary line plus the received load on the secondary line is less than the value entered for the *disable-load* argument, then the secondary line is disabled.

If the **never** keyword is used instead of an *enable-threshold* value, the secondary line is never activated due to load. If the **never** keyword is used instead of an *disable-load* value, the secondary line is never deactivated due to load.

By default, no backup loads are defined.

Example:

This example sets the traffic load threshold to 60% on the primary line. When that load is exceeded, the secondary line is activated, and will not be deactivated until the combined load is less than 5% of the primary bandwidth.

```
interface serial 0
backup load 60 5
```

Defining the Backup Line Delay

Use the **backup delay** interface subcommands to define how much time should elapse before a secondary line is set up or taken down (after a primary line transitions). The commands have the following syntax:

> **backup delay** {*enable-delay* | **never**} {*disable-delay* | **never**}
> **no backup delay** {*enable-delay* | **never**} {*disable-delay* | **never**}

Note: The interval configured with the **backup delay** subcommand does not affect the operation of the **backup load** subcommand.

The argument *enable-delay* is the delay in seconds after the primary line goes down before the secondary line is activated.

The argument *disable-delay* is the delay in seconds after the primary line goes up before the secondary line is deactivated.

When the primary line goes down, the router delays the amount of seconds defined by the *enable-delay* argument before enabling the secondary line. If, after the delay period, the primary line is still down, the secondary line is activated.

When the primary line comes back up, the router will delay the amount of seconds defined by the *disable-delay* argument. If the *disable-load* condition described above can be satisfied, or if the secondary is never activated due to load, then the secondary line is also deactivated.

Note: In cases where there are spurious signal disruptions that may appear as intermittent lost carrier signals, it is recommended that some delay be enabled before activating and de-activating a secondary.

Use the **never** keyword to prevent the secondary line from being activated or deactivated.

The **never** version is the default value for **backup delay**. In order for the secondary line to be activated when the primary line fails, a **backup delay** value must be specified.

Example:

This example sets a ten-second delay on deactivating the secondary line; however, the line is activated immediately.

```
interface serial 0
backup delay 0 10
```

Dial Backup Configuration Examples

This section contains three examples of different dial backup configurations.

Example 1:

The following example configures serial 1 as a secondary line that activates only when the primary line (serial 0) goes down. The secondary line will not be activated due to load of the primary.

```
interface serial 0
backup interface serial 1
backup delay 30 60
```

The secondary line is configured to activate 30 seconds after the primary line goes down and to remain on for 60 seconds after the primary line is reactivated.

Example 2:

The following example configures the secondary line (serial 1) to be activated only when the load of the primary line reaches a certain threshold.

```
interface serial 0
backup interface serial 1
backup load 75 5
```

In this case, the secondary line will not be activated when the primary goes down. The secondary line will be activated when the load on the primary line is greater than 75% of the primary's bandwidth. The secondary line will then be brought down when the aggregate load between the primary and secondary lines fit within 5% of the primary bandwidth.

Example 3:

This example configures the secondary line to activate once the traffic threshold on the primary line exceeds 25%.

```
interface serial 0
backup interface serial 1
backup load 25 5
backup delay 10 60
```

Once the aggregate load of the primary and the secondary lines return to within 5% of the primary bandwidth, the secondary line is deactivated. The secondary line waits 10 seconds after the primary goes down before activating, and remains active for 60 seconds after the primary returns and becomes active again.

Configuring Dial-on-Demand Routing

Dial-on-demand routing (DDR) provides network connections in an environment using the public switched telephone network (PSTN). Traditionally, networks have been interconnected using dedicated lines for WAN connections. When used with modems (or ISDN terminal adapters), DDR facilitates low-volume, periodic network connections over a PSTN. This section details the commands required to interconnect Cisco routers and the PSTN using data communications equipment (DCE) devices and dial-on-demand routing.

Cisco's DDR Implementation

Cisco's DDR implementation allows attachment to modems and ISDN terminal adapters (TAs) that support V.25 bis dialing. V.25 bis is a CCITT recommendation for initiating calls using an automatic calling unit (ACU). V25 bis is useful in establishing calls when transmitting synchronous data.

The V.25 bis specification describes two modes of establishing or receiving calls: the *direct call mode* and the *addressed call mode.* Cisco routers support connections using the addressed call mode and synchronous, bit-oriented operation. The addressed call mode allows the use of control signals and commands sent over the DCE data interface to establish and terminate calls. These commands are packaged in synchronous data frames (of type HDLC) and are sent when the interface is not in use for data transfer (that is, when a call does not exist on the line).

Cisco routers support connections from synchronous serial interfaces on the router to any DCE device that supports V.25 bis. Typically, these devices are V.32 (9.6 Kbps) or V.32 bis modems (14.4 Kbps). Supported devices also include ISDN TAs for ISDN B-channel connections.

The DDR call-initiating process is driven by certain characteristics of bridged or IP-routed packets.

In order for the router to use a device for dialing out, in addition to V.25 bis, the device must support certain hardware signals. When the router drops DTR, the device must disconnect any calls which are currently connected. When the device connects to the remote side, the device must have DCD become active.

Note: For many V.25 bis devices, the requirement for DCD to be raised in turn requires a special cable to swap DCD and DSR because the V.25 bis specification requires that DSR be raised when a connection is established.

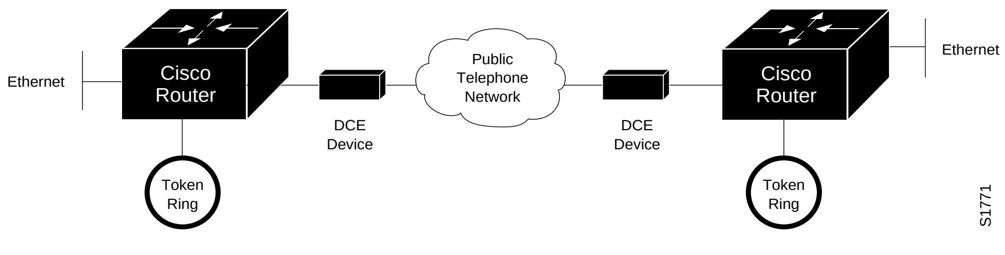
IS-IS, BGP, and OSPF may not be used as routing protocols with DDR. This is because they require an acknowledgment for routing updates, and DDR does not necessarily establish a connection when a routing update is to be transmitted.

For bridged traffic, packets of interest are based on *packet type* as specified with the hexadecimal value in a bridge access list. Refer to the chapter "Configuring Transparent Bridging" for more information about bridge access lists.

For IP traffic, packets of interest are based on the *destination address*; with a routed IP packet, the destination address used is the *next hop address*. Packets of interest are defined with IP access lists. Refer to the chapter "Routing IP" for more information concerning IP access lists.

Figure 1-1 illustrates a typical interconnection featuring dial-on-demand routing.

Figure 1-1 Dial-on-Demand Interconnection



Specifying V.25 bis Dialing

Use the **dialer in-band** interface subcommand to specify that dial-on-demand routing is to be supported on a synchronous serial line. The syntax for this command is:

dialer in-band
no dialer in-band

This specifies that V.25 bis dialing is to be used on the specified interface. This command is only used with serial interfaces.

Note: If an interface is only going to accept calls, but never place calls, then the **dialer in-band** interface subcommand is the only command needed to configure this interface. If an interface is configured in this manner, with no dialer groups (described below), then the interface is always active and the idle timer never disconnects the line. It is up to the remote end (the end that placed the call) to disconnect the line based on idle time.

The **no dialer in-band** command disables dial-on-demand routing for the interface.

Example:

The following example illustrates specification of dial-on-demand routing for serial 0:

```
interface serial 0
dialer in-band
```

Controlling Dialer Idle Time

Use the optional **dialer idle-timeout** interface subcommand to specify the idle time before the line is disconnected. The syntax for this command is:

dialer idle-timeout *number-of-seconds*
no dialer idle-timeout

The argument *number-of-seconds* specifies in seconds how much idle time must occur on an interface before the line is disconnected. Acceptable values are positive, nonzero integers. If no value is entered, the default is 120 seconds.

The **no dialer idle-timeout** command resets the idle timeout to the default.

Example:

The following illustrates specification of an idle timeout of three minutes on interface serial 1:

```
interface serial 1
dialer idle-timeout 180
```

Setting Idle Time for Busy Interfaces

Use the optional **dialer fast-idle** interface subcommand to specify the idle time before the line is disconnected for interfaces for which there is an unusually high level of contention. The syntax for this command is:

> **dialer fast-idle** *number-of-seconds*
> **no dialer fast-idle**

The argument *number-of-seconds* specifies in seconds how much idle time must occur on an interface before the line is disconnected. Acceptable values are positive, nonzero integers. If no value is entered, the default is 20 seconds.

This timer is used when an interface is connected, and a packet is received for a different destination (phone number). If the duration specified for the **dialer fast-idle** command is exceeded and no traffic has been detected on the connection during the fast-idle period, then the link is disconnected. This makes the port available for making connections to an alternate destination (after the **dialer enable-timeout** period expires) and reduces idle time on an interface.

This command only applies if multiple destinations have been specified for a given interface using the **dialer map** interface subcommand.

The **no dialer fast-idle** resets the timeout period to the default.

Example:

The following illustrates specification of an fast-idle timeout of 35 seconds on interface serial 1:

```
interface serial 1
dialer fast-idle 35
```

Specifying Dialer Enable Time

Use the optional **dialer enable-timeout** interface subcommand to set how long an interface stays down before it is available to dial again. The command syntax is:

> **dialer enable-timeout** *number-of-seconds*
> **no dialer enable-timeout**

Acceptable values for *number-of-seconds* are positive, nonzero integers. If no value is entered, the default is 15 seconds. When a call terminates (for example hangs-up or is busy), this is the amount of time that the router waits before the next call can occur on the specific interface.

The **no dialer enable-timeout** command resets the enable timeout value to the default.

Example:

The following example specifies a waiting period of 30 seconds on interface serial 1:

```
interface serial 1
dialer enable-timeout 30
```

Specifying Carrier Wait Time

Use the optional **dialer wait-for-carrier-time** interface subcommand to specify how long to wait for a carrier. The command syntax is:

dialer wait-for-carrier-time *number-of-seconds*
no dialer wait-for-carrier-time

The argument *number-of-seconds* specifies in seconds how long to wait for carrier to come up when a call is placed. Acceptable values are positive, nonzero integers. If no value is entered, the default is 30 seconds. If a carrier signal is not detected in this amount of time, the interface is disabled until the enable timeout occurs.

The wait-for-carrier time is set to a default of 30 seconds, since this is the estimated time required to make a typical connection over telephone lines. However, if connections are being made via ISDN links, this time would probably be configured for five or ten seconds.

The **no dialer wait-for-carrier-time** command resets the carrier wait time value to the default.

Example:

The following illustrates specification of an carrier wait time of 45 seconds on interface serial 1:

```
interface serial 1
dialer wait-for-carrier-timeout 45
```

Specifying a Single DDR Telephone Number

Use the **dialer string** interface subcommand to specify the string (telephone number) to be passed to the DCE device (typically a V.25 bis modem). The command syntax is:

dialer string *dial-string*
no dialer string

The argument *dial-string* specifies the character string to be passed to the V.25 bis DCE device. Cisco routers places the V.25 bis command *CRN* in front of the specified string. One **dialer string** command is specified per interface.

To specify multiple strings, use the **dialer map** command, discussed next. However, you must use the **dialer string** command if you intend to bridge traffic over the serial link or transmit broadcast traffic. In general, you include a **dialer string** or **dialer map** command if you intend to use a specific interface to initiate a dial-on-demand call.

Note: If a **dialer string** command is specified for the interface, and no **dialer-group** subcommand is assigned, or no access lists are defined, then dialing will never be initiated, and an error message will be displayed indicating that dialing will never occur if **debug serial-interface** is enabled.

The *dial-string* number specified in the **dialer string** command is the default number used under the following conditions:

- A **dialer map** command is not included in the interface configuration.
- The *next-hop-address* specified in a packet is not included in any of the **dialer map** interface subcommands recorded—assuming that the destination address passes any access lists specified for dial-on-demand via the **dialer-list** command (discussed later in this section)
- When bridging.

The **no dialer string** command deletes the dialer string specified for the interface.

Example:

The following illustrates specification of a dial-on-demand telephone number on interface serial 1 using the **dialer string** command:

```
interface serial 1
dialer string 14085553434
```

Specifying Multiple Destinations

Use the **dialer map** interface subcommand to define multiple dial-on-demand numbers for a particular interface. The command syntax is:

dialer map *protocol next-hop-address dial-string*
no dialer map *protocol next-hop-address dial-string*

The following arguments can be used with this command:

- The argument *protocol* has only one option at this time: **ip**.
- The argument *next-hop-address* specifies the IP address used to match against addresses to which packets are destined.
- The *dial-string* argument is the telephone number sent to the DCE dialing device when packets with the specified *next-hop-address* are seen (and if they match the access lists defined).

Unlike the **dialer string** command, multiple **dialer map** commands can be specified per interface.

The **dialer map** command complements the **dialer string** command. The *dial-string* defined for the **dialer string** command is considered the *default* dialer string. If a packet does not match the *protocol* and the *next-hop-address* defined in the **dialer map** commands (but does pass an access list that requires initiation of automatic dialing), then the default dialer string is used. This is often the case for broadcasts, and is always the case for bridging.

The **no dialer map** command deletes a particular dialer map entry. To delete the entry, include the complete configuration string (**no dialer map** *protocol next-hop-address dial-string*).

Note: If no **dialer string** command is defined, and the next-hop address included in a packet does not match any of the *next-hop-address* specifications in the **dialer map** command, then dialing will never be initiated, and an error message will be displayed indicating that dialing will never occur if **debug serial-interface** is enabled.

Example:

The following illustrates specification of a dial-on-demand telephone number on interface serial 1 for a specific destination address using the **dialer map** command:

```
interface serial 1
dialer map ip 131.108.2.5 14155553434
```

Assigning Access Lists to a DDR Interface

Use the **dialer-group** interface subcommand to assign an interface to a set of access list expressions. These access list expressions define what packets cause a connection to be established, and what packets keep an interface from being idle. The syntax for this command is:

dialer-group *group-number*
no dialer-group

The argument *group-number* specifies the number of the dialer group to which the specific interface belongs. Acceptable values are nonzero, positive integers. This group number corresponds to a dialer group defined using the **dialer-list** command (described below).

An interface can only be associated with a single dialer group; multiple **dialer-group** assignment is not allowed.

The **no dialer-group** command removes an interface from the specified dialer-group.

Example:

The following example illustrates specification of a dialer group number 1. This dialer group number will be compared against any global **dialer-list** specifications. If there is a group number match, then the destination address of the packet in question is evaluated against the access list specified in the associated **dialer-list** command. If it passes, then a call is initiated or the idle timer is reset (if a call is currently connected).

```
interface serial 1
dialer-group 1
```

Using Access Lists with DDR

Control automatic dialing using DDR with the following combination: standard IP or bridging access lists; the **dialer-list** global command; and the **dialer-group** interface subcommand (described above). Refer to the "Routing IP" and "Configuring Transparent Bridging" chapters for more information about access lists. The **dialer-list** global configuration command has two forms. These have the following syntax:

dialer-list *dialer-group* **list** *list-number*
no dialer-list *dialer-group* **list** *list-number*

dialer-list *dialer-group* **protocol** *protocol-name* **permit**|**deny**
no dialer-list *dialer-group* **protocol** *protocol-name* **permit**|**deny**

Note: The **permit/deny** version of the **dialer-list** command provides a very coarse method of allowing or denying protocols that prevent a link from being considered idle or from causing dialing. In most cases, this form of the **dialer-list** command is not recommended because most network protocols have *periodic* messages. These periodic messages cause the interface to remain active, thus, the line is *never* disconnected. Currently filtering using access lists is only provided for routing IP and bridging. In addition, *only* those access lists applicable to bridging and IP routing are supported by the dial-on-demand facilities.

The argument *dialer-group* specifies the number of a dialer group identified in any **dialer group** interface subcommand.

The argument *list-number* is the access list number specified in any IP or bridging access list.

The argument *protocol-name* is one of the following supported protocols:

- **ip**—IP
- **decnet**—DECnet
- **chaos**—CHAOSnet
- **xns**—XNS
- **novell**—Novell IPX
- **appletalk**—AppleTalk

- **vines**—Banyan Vines
- **apollo**—Apollo Domain
- **pup**—PUP
- **clns**—OSI Connectionless Network Service
- **bridge**—Bridging
- **stun**—Serial tunneling
- **cmns**—OSI Connection-Mode Network Services

Note: Access lists are central to the operation of dial-on-demand routing for Cisco routers. In general, they are used as the screening criteria for determining when to initiate DDR calls. When an access list is found to match with the packet being tested, the idle timer is reset, or a connection is attempted (assuming the line is not active, but is available). If a tested packet is deemed *uninteresting* (it does not match any access list specifications), it will be forwarded if intended for a destination known to be on a specific interface and the link active. However, such a packet will not initiate a DDR call and will not reset the idle timer.

Example:

Dialing occurs when an interesting packet needs to be output on an interface. Using the standard access list method, packets can be classified as interesting or uninteresting. For example, to specify that IGRP updates are not interesting (relative to dial-on-demand automatic dialing), the following access list would be defined:

```
access-list 101 deny igrp 0.0.0.0 255.255.255.255 255.255.255.255
0.0.0.0
```

To permit all other IP traffic, the above would be followed by:

```
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
```

Then, the following command would be used to place list 101 into dialer group 1.

```
dialer-list 1 list 101
```

Dial-on-Demand Configuration Examples

The examples provided in this section illustrate various DDR configurations using access lists linked to DDR facilities provided with Cisco routers.

Example 1:

The following example illustrates how dial-on-demand routing can be used in an IP environment.

```
interface serial 0
ip address 131.108.126.1 255.255.255.0
dialer in-band
dialer idle-timeout 600
dialer string 555-1234
pulse-time 1
dialer-group 1
!
access-list 101 deny   igrp 0.0.0.0 255.255.255.255 255.255.255.255 0.0.0.0
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
dialer-list 1 list 101
ip route 131.108.29.0 131.108.126.2
ip route 131.108.1.0 131.108.126.2
```

Configuration specifications in this example define the following characteristics:

- **dialer in-band** command selects V.25 bis dialing.
- **dialer idle-timeout 600** command sets the dialer idle timeout to ten minutes.
- **dialer string 555-1234** command sets the phone number to be called to 555-1234.
- **pulse-time 1** command (defined in the chapter "Adjusting Interface Characteristics") sets the DTR pulse time to be one second. This is to give the modem enough time to recognize that DTR has dropped so the modem will disconnect the call.

Note: With many modems, the **pulse-time** command must be used so that DTR is dropped for sufficient time to allow the modem to disconnect.

- **dialer-group 1** adds this interface to the dialer group defined with the dialer-list command.
- The first access list statement used specifies that IGRP updates to any address will not cause dialing. However, if a connection is already established, IGRP updates will be transmitted, although they will not reset the idle timers.
- The second access-list statement specifies that all other IP traffic—such as Ping, Telnet, or any other IP packet—will cause dialing.
- The **dialer-list** command then creates dialer group 1 and states that access list 101 is to be used to match for automatic dialing.
- The **ip route** commands specify that there is a route to network 131.108.1.0 and to network 131.108.29.0 via 131.108.126.2. This means that several destination networks are available via the serial port.

Example 2:

The following example illustrates a configuration for bridging.

```
interface serial 0
ip address 131.108.126.10 255.255.255.0
dialer in-band
dialer string 555-1234
```

```
    pulse-time 1
    bridge-group 1
    dialer-group 1
    !
    !
    bridge 1 protocol dec
    access-list 201 deny   0x6002 0x0000
    access-list 201 deny   0x6003 0x0000
    access-list 201 permit 0x0000 0xFFFF
    dialer-list 1 LIST 201
```

As in Example 1, the dialer is set to use V.25 bis, the dialer string is specified and a pulse time is assigned. This interface is then added to bridge group 1 and dialer group 1.

The access lists work as follows:

- The first one states that DEC MOP packets are not to cause dialing.
- The second that DECnet Phase IV should not cause dialing.
- The last that all other bridged packets should cause dialing.

This example could be used in a case where LAT is the only traffic to be transmitted over the link.

Note: Spanning-tree updates and HDLC keepalives implicitly do not cause dialing and do not reset the idle timers.

Example 3:

The following example is a combination of Examples 1 and 2. It illustrates how to handle DDR for bridging and IP routing over a single interface.

The first **access-list** statement specifies that RIP updates to the broadcast address do not cause dialing. However, if a connection is already established, RIP updates are transmitted, although they do not reset the idle timers and do not keep the line up.

```
    interface serial 0
    ip address 131.108.126.10 255.255.255.0
    dialer in-band
    dialer string 555-1234
    pulse-time 1
    bridge-group 1
    dialer-group 1
    !
    !
    bridge 1 protocol dec
    access-list 101 deny udp 0.0.0.0 255.255.255.255 255.255.255.255 0.0.0.0 eq 520
    access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
    access-list 201 deny   0x6002 0x0000
    access-list 201 deny   0x6003 0x0000
    access-list 201 permit 0x0000 0xFFFF
    dialer-list 1 LIST 101
    dialer-list 1 LIST 201
```

Example 4:

The following example demonstrates how to specify multiple destination dial string (phone numbers).

```
interface serial 0
ip address 131.108.126.1 255.255.255.0
dialer in-band
dialer wait-for-carrier-time 10
dialer string 555-1234
pulse-time 1
dialer-group 1
dialer map ip 131.108.126.10 555-8899
dialer map ip 131.108.127.1 555-5555
!
access-list 101 deny   igrp 0.0.0.0 255.255.255.255 255.255.255.255 0.0.0.0
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
dialer-list 1 LIST 101
```

As in Example 1, the serial interface is configured for V.25 bis. It also is set to have a wait-for-carrier-time to ten seconds, which is too short for POTS lines, but might be appropriate for ISDN lines. As before, a default dial string is configured, a pulse time assigned, and a dialer group specified.

The first **dialer map** command specifies that the number 555-8899 is to be dialed for IP packets with a *next-hop-address* of 131.108.126.10. The second **dialer map** then specifies that the number 555-5555 will be called when an IP packet with a *next-hop-address* of 131.108.127.1 is detected.

Access lists for this example match those defined in Example 1.

Example 5:

The following example illustrates using *floating static routes* and dial-on-demand to perform the same functionality as dial backup, except using V.25 bis for dialing.

```
interface serial 0
ip address 131.108.126.1 255.255.255.0
dialer in-band
dialer wait-for-carrier-time 10
dialer string 555-1234
pulse-time 1
dialer-group 1
dialer map ip 131.108.126.10 555-8899
dialer map ip 131.108.127.1 555-5555
!
access-list 101 deny   igrp 0.0.0.0 255.255.255.255 255.255.255.255
0.0.0.0
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0
255.255.255.255
dialer-list 1 LIST 101
ip route 131.108.42.0 131.108.126.10 150
```

The configuration is essentially the same as in Example 4, except for the addition of an **ip route** statement. This statement specifies that to reach network 131.108.42.0 through this router, traffic must use 131.108.126.10 as the gateway. This causes dialing to occur to 555-8899.

Since the **ip route** statement specifies an administrative distance, dynamic routing information will override the static route—no packets will normally be sent to 131.108.126.10. However, if the route that is learned dynamically disappears, and no other dynamic route is available, then the static route will be used and dialing will occur.

Monitoring Dial-on-Demand Routing

To obtain a general diagnostic display for serial interfaces configured for dial-on-demand routing, use the **show dialer** EXEC command. The command syntax is:

show dialer [**interface** *interface type*]

This command displays information about the dialer interfaces. If **show dialer** is specified, all dialers are displayed. An interface can be specified using **show dialer interface** option. The arguments *interface* and *type* specify the particular interface, and can be formed either as a two-part identifier (such as serial 0, ser 1, s 2, and so on) or as a name (serial1).

The following is a sample of output from the **show dialer** command:

```
Serial 0 - dialer type = IN-BAND
Idle timer (600 secs), Fast idle timer (20 secs)
Wait for carrier (30 secs), Re-enable (15 secs)
Dial String      Successes   Failures    Last called   Last status
8985                   0         2        0:24:10      Failed       Default
8986                   1         0        1:20:07      Successful
```

The following information is provided in this display:

- "IN-BAND" specification indicates DDR is enabled.
- Idle timeout specification (seconds).
- Fast-idle timer specification (seconds).
- Wait-for-carrier timer specification (seconds).
- Enable timeout specification (seconds).
- Dial strings of logged calls (telephone numbers).
- Successful connections (even if no data is passed).
- Failed connections—line busy when called attempted.
- Time that last call occurred to specific dial string.
- Last status of calls to specific dial string (successful or failed).
- If the DDR facility is using the dial string specified with the **dialer string** command, the word `Default` is appended to the `Last status` entry.

If the dialer is connected to another dialer (and the system from which the **show dialer** command was entered was the initiator of the call), a display is provided that indicates the idle time before the line is disconnected (decrements each second) and the duration of the current connection. The following is a sample of this display; it would appear after the third line in the **show dialer** display:

```
Time until disconnect 596 secs
Current call connected 0:00:25
```

After a call disconnects, the system displays the time remaining before being available to dial again. The following is a sample of this display; it would appear after the third line in the **show dialer** display:

```
Time until interface enabled 8 secs
```

If the **show dialer** interface command is issued for an interface on which DDR is not enabled, the system displays an error message. The following is a sample error message:

```
Serial 1 - Dialing not enabled on this interface.
```

If an interface is configured for DDR, the **show interfaces** command now displays the following:

```
Serial 0 is up, line protocol is up (spoofing)
Hardware is MCI Serial
```

The *spoofing* indicates that the line really is not up, but the dialer is forcing the line to masquerade as "up" so that upper level protocols will continue to operate as expected.

Debugging Dial-on-Demand Routing

When DDR is configured for a serial interface, the EXEC commands **debug serial-interface** (described earlier in this chapter) and **debug serial-packet** can provide information associated with DDR activity, as follows:

debug serial-interface

This command enables monitoring of serial interface activity. For DDR, the following information may appear:

- A message indicating that a call is being attempted
- A message stating that no dialer-group has been defined, and that dialing will not occur
- Messages indicating that idle timeouts, re-enable timeouts, and wait-for-carrier timeouts have occurred
- A message from the DCE device indicating the dial string attempted

The **debug serial-packet** command enables monitoring of serial packets.

debug serial-packet

When activated, and DDR is enabled on the interface, information concerning the cause of any calls (called `Dialing cause`) may be displayed. The cause for IP packets lists the source and destination addresses; the cause for bridged packets lists the type of packet, in hexadecimal).

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 start up.

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

Note: PPP does not support the keepalive timer function described in “Keepalive Timers” in the the “IP Routing Protocols” chapter.

Example:

These commands enable PPP encapsulation on serial interface 0 (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
```

Global Configuration Command Summary

The following summary lists the global configuration command used for interface support.

[**no**] **dialer-list** *dialer-group* **list** *list-number*
[**no**] **dialer-list** *dialer-group* **protocol** *protocol-name* **permit**|**deny**

Controls automatic dialing using DDR using standard IP or bridging access lists.

The argument *dialer-group* specifies the number of a dialer group identified in any **dialer group** interface subcommand.

The argument *list-number* is the access list number specified in any IP or bridging access list.

The argument *protocol-name* is the following supported protocol:

- **ip**—IP
- **decnet**—DECnet
- **chaos**—CHAOSnet
- **xns**—XNS
- **novell**—Novell IPX
- **appletalk**—AppleTalk
- **vines**—Vines
- **apollo**—Apollo Domain
- **pup**—PUP
- **clns**—OSI Connectionless Network Service
- **bridge**—Bridging
- **stun**—Serial tunneling
- **cmns**—OSI Connection-Mode Network Service

interface *type unit*

Specifies an interface. The argument *type* specifies the interface type—**ethernet**, **fddi**, **hssi**, **serial**, **tokenring**, or **ultranet**—and the argument *unit* specifies the interface number or card number.

[**no**] **smt-queue-threshold** *number*

Sets the maximum number of unprocessed Station Management (SMT) frames that will be held for processing. The argument *number* specifies the number of unprocessed SMT message that are queued for processing. This value is a positive integer. The default is equal to the number of FDDI interfaces installed in the router. The command **no smt-queue-threshold** restores the queue to the default.

Interface Support Subcommand Summary

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

[**no**] **backup delay** {*enable-delay* | **never**} {*disable-delay* | **never**}

Defines how much time should elapse before a line is set up or taken down. The argument *enable-delay* is the delay in seconds after the primary line goes down before the secondary line is activated. The argument *disable-delay* is the delay in seconds after the primary line goes up before the secondary line is deactivated. The **never** keyword prevents the secondary line from being activated due to a primary line failure. The **no** version disables the specified delay time.

[**no**] **backup interface** *interface-name*

Configures the serial interface as a secondary, or dial backup, line. The argument *interface-name* specifies the serial port to be set as the secondary interface line. Use the **no backup** interface command with the appropriate serial port designation to turn this feature off.

[**no**] **backup load** {*enable-threshold* | **never**} {*disable-load* | **never**}

Sets the traffic load thresholds. The arguments *enable-threshold* and *disable-load* are percentage numbers representing the load as a percentage of the primary line's available bandwidth. The **never** keyword prevents the secondary line from being activated due to load. By default, no backup loads are defined; the **no** form of the command restores this default.

[**no**] **description** *name-string*

Adds a descriptive name to an interface. The argument *name-string* is a comment to be put in the configuration. The **no** version removes the *name-string*.

[**no**] **dialer enable-timeout** *number-of-seconds*

Sets how long an interface stays down before it is available to dial again. Acceptable values are positive, nonzero integers. If no value is entered, the default is 15 seconds.The **no dialer enable-timeout** command resets the enable timeout value to the default.

[**no**] **dialer fast-idle** *number-of-seconds*

Specifies the idle time before the line is disconnected for interfaces for which there is an unusually high level of contention. Acceptable values are positive, nonzero integers. If no value is entered, the default is 20 seconds.This timer is used when an interface is connected, and a packet is received for a different destination. This command only applies if multiple destinations have been specified for a given interface using the **dial map** interface subcommand. The **no dialer fast-idle** resets the timeout period to the default.

[**no**] **dialer-group** *group-number*

Assigns an interface to a set of access list expressions. These access list expressions define what packets cause a connection to be established, and what packets keep an interface from being idle.

The argument *group-number* specifies the number of the dialer group to which the specific interface belongs. Acceptable values are nonzero, positive integers. This group number corresponds to a dialer group defined using the **dialer-list** command (described below).

An interface can only be associated with a single **dialer-group**.

The **no dialer-group** command removes an interface from the specified dialer-group.

[**no**] **dialer idle-timeout** *number-of-seconds*

Specifies the idle time before the line is disconnected.

The argument *number-of-seconds* specifies in seconds how much idle time must occur on an interface before the line is disconnected. Acceptable values are positive, nonzero integers. If no value is entered, the default is 120 seconds.

The **no dialer idle-timeout** command resets the idle timeout to the default.

[**no**] **dialer in-band**

Specifies that V.25 bis dialing is to be used on the specified interface. This is only used with serial interfaces.

The **no dialer in-band** command disables dial-on-demand routing for the interface.

[**no**] **dialer map** *protocol next-hop-address dial-string*

Defines multiple dial-on-demand numbers for a particular interface.

The argument *protocol* has only one option at this time: **ip**.

The argument *next-hop-address* specifies the IP address used to match against addresses to which packets are destined.

The *dial-string* argument is the dial string (telephone number) sent to the DCE dialing device when packets with the specified *next-hop-address* are seen (and if they match the access lists defined).

The **no dialer map** command deletes a particularly dialer map entry. To delete the entry, you include the complete configuration string (**no dialer map** *protocol next-hop-address dial-string*).

[**no**] **dialer string** *dial-string*

Specifies the string (i.e., telephone number) to be passed to the DCE device (typically a V.25 bis modem).

The argument *dial-string* specifies the character string to be passed to the V.25 bis DCE device. Cisco routers place the V.25 bis command *CRN* in front of the specified string. One **dialer string** command is specified per interface.

The **no dialer string** command deletes the dialer string specified for the interface.

[**no**] **dialer wait-for-carrier-time** *number-of-seconds*

Specifies how long to wait for the carrier to come up when a call is placed. Acceptable values are positive, nonzero integers. If no value is entered, the default is 30 seconds. If a carrier signal is not detected in this amount of time, the interface is disabled until the enable timeout occurs.

The wait-for-carrier time is set to a default of 30 seconds since this is the estimated time required to make a typical connection of telephone lines. However, if connections are being made via ISDN links, this time would probably be configured for 5 or 10 seconds.

The **no dialer wait-for-carrier-time** command resets the carrier wait time value to the default.

[**no**] **early-token-release**

Enables a method whereby these token ring interfaces can release the token back onto the ring immediately after transmitting. By default, early token release is not enabled on the interface. The **no early-token-release** command disables this feature.

encapsulation *encapsulation-type*

Assigns an encapsulation method. The *encapsulation-type* argument is a keyword that identifies one of the following 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

fddi cmt-signal-bits *signal-bits* **phy-a**|**phy-b**

Controls the information transmitted during the CMT signaling phase. The argument *signal-bits* is written as a hexadecimal number preceded by "0x"; for example, "0x208." The keywords **phy-a** and **phy-b** select the Physical Sublayer (Physical A or Physical B station), for control of each fiber.

fddi tl-min-time *microseconds*

Controls the FDDI TL_MIN time (the minimum time to transmit a Physical Sublayer, or PHY line state before advancing to the next Physical Connection Management, or PCM state, as defined by the X3T9.5 specification). The specification defines the argument *microseconds* to be a value of 30. This value is used during the Connection Management (CMT) phase to ensure that signals are maintained for at least the value of TL_MIN so the remote station can acquire the signal.

Note: Interoperability tests have shown that some implementations of the FDDI standard need more than 30 microseconds to sense a signal.

fddi token-rotation-time *microseconds*

Controls ring scheduling during normal operation, and detects and recovers from serious ring error situations. The argument *microseconds* determines the token rotation time (TRT). The default value is 5000 microseconds.

fddi valid-transmission-time *microseconds*

Recovers from a transient ring error. The argument *microseconds* sets the transmission valid timer (TVX) interval. The default valid transmission timer value is 2500 microseconds.

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.

ultranet address *ultranet-mac-address*

Assigns the MAC layer address of the interface. The argument *ultranet-mac-address* is the MAC address of the UltraNet service line.

Interface Support EXEC Command Summary

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

clear counters [*interface-name*]

The argument *interface-name* is the name of the interface (examples are serial0, ethernet 1, and so on).

clear interface [*interface-name*]

Resets the hardware logic on an interface. Argument *type* is the specific interface type keyword (**ethernet**, **fddi**, **hssi**, **serial**, **tokenring** or **ultranet**); the argument *unit* is the interface number (0, 1, 2, and so on).

cmt connect [*interface-name* [**phy-a**|**phy-b**]]

Starts the FDDI CMT process.

cmt disconnect [*interface-name* [**phy-a** | **phy-b**]]

Stops the FDDI CMT process.

[**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. Use the **undebug** version of the command to stop **debug** messaging.

[**un**]**debug fddi-cmt-events**

Enables or disables logging of FDDI Connection Management (CMT) transactions.

[**un**]**debug fddi-smt-packets**

Enables or disables logging of FDDI station management (SMT) packets.

[**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 serial-interface**

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

[**un**]**debug serial-packet**

Enables logging of serial-interface events for network servers equipped with serial network interfaces. Provides more detail than **debug serial-interface**.

[**un**]**debug token-ring**

Enables or disables 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** | **cbus** | **fddi**}

Displays current internal status information for different interface cards.

show dialer [**interface** *interface type*]

Displays information about the dialer interfaces.

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**, **fddi**, **hssi**, **serial**, **tokenring**, or **ultranet**. The argument *unit* specifies the interface unit or card number.

show version

Displays the configuration of the system hardware, the software version, the names and sources of configuration files, and the boot images.