

## Chapter 9

# Interfaces

Introduction .....	9-3
Naming Interfaces .....	9-4
Simple Interface Names .....	9-4
Fully Qualified Interface Names .....	9-5
Ethernet .....	9-7
Encapsulations .....	9-8
Configuration .....	9-10
Synchronous Interfaces .....	9-12
Encapsulations .....	9-13
Modem Control Signals .....	9-13
Configuration .....	9-14
Asynchronous Interfaces .....	9-16
Encapsulations .....	9-17
Configuration .....	9-17
Session Timeout .....	9-20
Connecting a Modem to the Asynchronous Port .....	9-21
MIB Counters .....	9-21
Autobauding .....	9-22
Making Asynchronous Ports Respond More Quickly .....	9-23
Testing Serial Data Circuits .....	9-23
Displaying Interfaces .....	9-24
Interface Link Traps .....	9-24
Managing Interfaces with SNMP .....	9-25
Command Reference .....	9-26
connect asyn .....	9-26
disable asyn .....	9-27
disable interface debug .....	9-28
disable interface linktrap .....	9-29
disable syn .....	9-30
disable syn debug .....	9-30
enable asyn .....	9-31
enable interface debug .....	9-31
enable interface linktrap .....	9-32
enable syn .....	9-33
enable syn debug .....	9-33
purge asyn .....	9-34
reset asyn .....	9-34
reset asyn counter .....	9-35
reset asyn history .....	9-36
reset eth .....	9-36
reset eth counters .....	9-37

reset interface counters .....	9-37
reset syn .....	9-38
reset syn counters .....	9-39
set asyn .....	9-40
set eth linkup .....	9-45
set eth maxbandwidth .....	9-45
set eth speed .....	9-46
set interface mtu .....	9-48
set interface traplimit .....	9-49
set syn .....	9-50
show asyn .....	9-53
show eth configuration .....	9-61
show eth counters .....	9-62
show eth macaddress .....	9-69
show eth receive .....	9-70
show eth state .....	9-71
show interface .....	9-73
show syn .....	9-79
show syn counter .....	9-81

## Introduction

---

This chapter describes how to configure, control and monitor interfaces, and the encapsulations supported on each interface. The chapter also describes the format of the Ethernet frame, and the naming conventions that are available for different interface types.

Some interface and port types mentioned in this chapter may not be supported on your router. The interface and port types that are available vary depending on your product model and whether an expansion unit is installed, such as a PIC, NSM, or expansion module. For more information, see the Hardware Reference for the router.

The interfaces described are:

- Ethernet
- synchronous
- asynchronous

Other interfaces are described in:

- [Chapter 8, Switching](#) (Ethernet switch ports and VLANs).
- [Chapter 10, Integrated Services Digital Network \(ISDN\)](#) (Basic Rate ISDN (BRI) and Primary Rate ISDN (PRI) ports).

The term *interface* refers to one of the physical ports on the router or on one of its expansion devices (such as PIC, NSM). The physical ports connect the router to a network, and all data enters and leaves the router via the interface.

Ethernet interfaces described in this chapter are Ethernet ports (labelled ETH), and not the switch ports described in [Chapter 8, Switching](#). The main distinction between models is the combination of different types and numbers of interfaces. See the Hardware Reference for details of the interfaces available on each model.

Asynchronous ports can be used to connect terminals, printers and terminal ports on host computers. See [Chapter 60, Terminal Server](#) for information about using asynchronous ports for terminal serving. See [Chapter 61, Line Printer Daemon \(LPD\)](#) for information about using asynchronous ports for print serving using the Line Printer Daemon (LPD) protocol. See [Chapter 62, Stream Printing](#) for information about using asynchronous ports for print serving using the stream printing service. See [Chapter 63, Permanent Assignments](#) for information about using asynchronous ports for print serving using permanent assignments.

Each frame of data includes a header that informs a receiving router about the protocol carried in the frame. This header is specified by a set of rules referred to as an *encapsulation*. Some interface types can be used with more than one encapsulation. It is important to know about encapsulations for two reasons. Firstly, the information can be useful in debugging network problems, if traces of the packets being transmitted or received on a particular interface can be obtained. Secondly, information about encapsulations can be used to determine whether the router can interoperate with other vendors' routers, since this depends on both routers supporting the same encapsulation(s) for a particular protocol.

Encapsulations supported for synchronous ports are:

- Frame Relay (see [Chapter 13, Frame Relay](#))
- PPP (see [Chapter 14, Point-to-Point Protocol \(PPP\)](#))
- X.25 (see [Chapter 12, X.25](#))

When one of these Layer 2 modules is *attached* to a synchronous interface, this creates a *logical interface*. The term *interface* also refers to these logical interfaces.

## Naming Interfaces

Commands that configure an interface or attach a routing module to use a particular interface, must specify the interface by name. Typically, commands use the **interface=interface** or **over=interface** parameter to specify an interface.

The asynchronous port is called *asyn0*.

Interfaces may be identified by their *simple name*, or for physical interfaces, by their *fully qualified name*.

## Simple Interface Names

Create simple interface names by concatenating the interface type with the interface instance. The interface type is an abbreviation of the full name of the interface. The instance is a non-negative number. The following table describes names for types of interfaces.

Interface Type	Description
<b>Logical</b>	
FR	Frame Relay interface
VLAN	Virtual LAN interface
LAPB	X.25 LAPB interface
PPP	Point-to-Point Protocol interface
X25C	X.25 DCE interface
X25T	X.25 DTE interface
<b>Physical</b>	
ASYN	Asynchronous interface
BRI	Basic Rate ISDN interface
ETH	Ethernet management interface
PORT	Ethernet switch ports (including uplinks)
PRI	Primary Rate ISDN interface
SYN	Synchronous interface

For logical interfaces, the instance number is the module instance number specified in the **add** or **create** command for that module. Instance numbers may be chosen arbitrarily but common practice is to assign them sequentially, starting with 0.

For physical interfaces, the instance number is the physical port number, which the system determines. Physical ports are numbered from left to right as viewed, starting at 0.

Permanent interfaces are numbered first, followed by removable interfaces (interfaces on PIC or NSM cards).

The following table shows examples of valid names for simple interfaces.

Interface name	Description
eth0	Ethernet port 0
fr2	Frame Relay instance 2
asyn4	Asynchronous port 4
port3	Switch port 3
ppp1	Point-to-Point Protocol instance 1
vlan1	Virtual LAN 1

## Fully Qualified Interface Names

Physical interfaces can be identified by their fully qualified interface name. A fully qualified interface name is one that includes the *path* to that interface. It is constructed by concatenating the interface name to the names of the bays and slots on the physical path from the base unit to the physical interface. Fields in the name are separated by dots. For simple interface names, the name consists of the physical interface type name ([“Simple Interface Names” on page 9-4](#)) followed by an instance number. The instance number uniquely identifies the interface in the NSM, PIC card or base unit.

If the interface is located directly on the base unit (not on a PIC or NSM card), then its fully qualified name is the same as its simple name—the interface type name and the interface instance number. For example, eth0 is the first Ethernet port on the base unit.

If the interface is located on a PIC or NSM card installed in the base unit, then the fully qualified name includes the bay where the PIC or NSM card is installed, the interface type name, and the interface instance number. For example, bay0.bri0 is the first Basic Rate ISDN port on the PIC card in PIC bay 0 on the base unit, and nsm0.bri2 is the second Basic Rate ISDN port on the NSM card in NSM bay 0 on the base unit.

If the interface is located on a PIC card installed in an AT-AR040 4-PIC NSM installed in the base unit, then the fully qualified name includes the bay where the NSM card is installed, the bay where the PIC card is installed, the interface type name, and the interface instance number. For example, nsm0.bay1.asyn2 is the third asynchronous port on the PIC card in PIC bay 1 of the NSM card in NSM bay 0 on the base unit.

The following table lists examples of valid fully qualified interface names and the equivalent simple names.

Interface Location	Interface Simple Name	Interface Fully Qualified Name
AR750S base	eth0	eth0
AR750S base	eth1	eth1
AR750S base	asyn0	asyn0
AR750S base	asyn1	asyn1
ETH0 in AR750S Bay 0	eth2	bay0.eth0
ASYN4 in AR750S Bay 1	asyn2	bay1.asyn0
	asyn3	bay1.asyn1
	asyn4	bay1.asyn2
	asyn5	bay1.asyn3
ETH0 in NSM4PIC Bay 0	eth3	nsm0.bay0.eth0
ASYN4 in NSM4PIC Bay 1	asyn6	nsm0.bay1.asyn0
	asyn7	nsm0.bay1.asyn1
	asyn8	nsm0.bay1.asyn2
	asyn9	nsm0.bay1.asyn3
SYNC1 in NSM4PIC Bay 2	syn0	nsm0.bay2.syn0
BRI1 in NSM4PIC Bay 3	bri0	nsm0.bay3.bri0
router base	eth0	eth0
router base	asyn0	asyn0

Any command that takes a physical interface as a parameter accepts either the simple name or the fully qualified name of the interface:

- Parameters that require only an interface instance (e.g. ETH=2) also accept the fully qualified name (e.g. ETH= bay0.eth0).
- Parameters that require a simple name created by concatenating an interface type and an instance number (e.g. OVER=syn0) also accept the fully qualified name (e.g. OVER=nsm0.bay0.syn0).
- Parameters that require a logical instance created by concatenating an interface type, an interface instance number and a logical instance number (e.g. INT=ppp1-1) also accept the fully qualified interface name (e.g. INT=nsm0.bay0.ppp0-1).
- Parameters that identify interfaces by an index number such as *ifIndex* (e.g. INT=1) also accept the fully qualified interface name (e.g. INT=nsm0.bay2.syn0).

For a summary of interfaces, including their fully qualified names, see the output of the [show interface command on page 9-73](#).

The [create config command on page 5-23 of Chapter 5, Managing Configuration Files and Software Versions](#) always generates configuration commands for physical interfaces using fully qualified names.

The interface table in the Enterprise MIB includes the read-only MIB object *arInterfaceFullName* that can be used to access the fully qualified names of all detected interfaces. The existing interface name entry is used to access the simple names of all interfaces.

## Ethernet

---

Ethernet encapsulation is used on *switch ports* and *eth* ports on the router. For more information about switch ports and VLAN tagging in Ethernet frames, see [Chapter 8, Switching](#).

*Ethernet* is a term that describes a particular family of interface types and encapsulations. Other common terms are *802.3* and *CSMA/CD*. Various physical media can carry Ethernet, including thin and thick coaxial cable, twisted pair wires, and optical fibre.

All these forms of Ethernet are characterised by these common features:

- A single medium carries all incoming and outgoing traffic.
- A number of stations may use the same medium for communicating with all other stations on the medium. All stations can see all the traffic on the medium.
- Stations wait for the medium to become free before attempting to send data on it. If more than one station attempts to send data simultaneously a collision results and the data being sent becomes invalid.
- Stations can be connected to or disconnected from the medium without disturbing the other stations on the medium.
- The order in which stations are attached to the physical medium is not important.

Ethernet runs at speeds of 10 Mbps, 100 Mbps, 1 Gbps, or 10 Gbps.

Ethernet is used primarily to provide local area networking rather than wide area networking. The installation of Ethernet media within premises is normally the responsibility of the user of the premises rather than the telecommunications provider.

Ethernet was first defined in 1982. The original definition is generally referred to as Type 1 Ethernet and although it differs slightly from the modern standard, it is not very common today. Subsequent standards defined Type 2 Ethernet, which was largely ratified unchanged by the IEEE as IEEE 802.3.

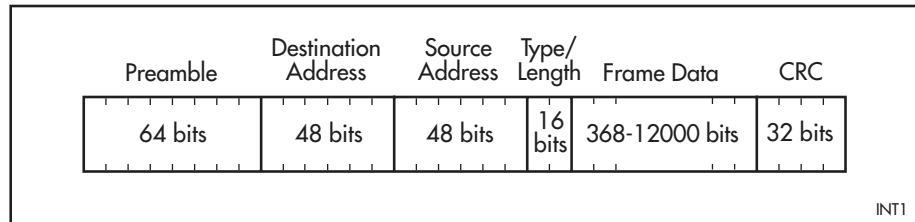
Ethernet interfaces on the router are specified by the IEEE Standard 802.3 or ISO 8802-3 standards. This is the standard used by most implementations. The router physically supports all three versions of Ethernet, and is supplied with Type 2/ 802.3 selected.

Some routers have a 10BASE-T and/or an AUI connector. The AUI connector is a 15-pin connector that provides a common connector for all 10 Mbps Ethernet media variations. Transceivers are available to connect the AUI connector to any 10 Mbps Ethernet media, including thin wire (10BASE2) and twisted pair (10BASET). The AUI interface may also connect directly to a repeater. The AUI connector allows connection to any form of 10 Mbps Ethernet media, albeit with a transceiver. For models with dual 10BASET/AUI connectors, only one connector may be used at any one time.

## Encapsulations

Since Ethernet is a single wire used by many stations at once and with many different protocols, encapsulation of protocol types is used to distinguish the protocols. Ethernet has been developed over a period of time, and the efforts of the Standards bodies following on from the vendors that developed Ethernet have led to different encapsulation types for Ethernet.

The following figure shows an Ethernet frame, which consists of a preamble followed by the data, and terminated with a CRC



The data begins with the station addresses of the receiver and sender of the frame. These address fields are both 6 octets long. Following the addresses is a 2-octet field, referred to here as the type/length field, that contains either a type field or a length.

The type/length field was introduced by the vendors that developed Ethernet and was used to contain a protocol type. Different values in the type field distinguished different protocols. The values that are contained in this field are administered by Xerox Corporation and vendors of network equipment may apply to reserve a type field to define vendor-specific protocols.

The original vendor specifications were extended by the IEEE. This body developed standards in local area networking, including Ethernet. The Ethernet addresses and type/length field appear in the IEEE standards as part of the Ethernet-specific standard, IEEE Standard 802.3. Another standard, IEEE Standard 802.2, specifies the format of the frame after the type/length field. Since IEEE Standard 802.2 applies to other LAN media, such as Token Ring and FDDI, the frame after the type/length field cannot contain anything specific to Ethernet. For this reason the type/length field is used to specify a length, and is, in fact, the length of the rest of the frame.

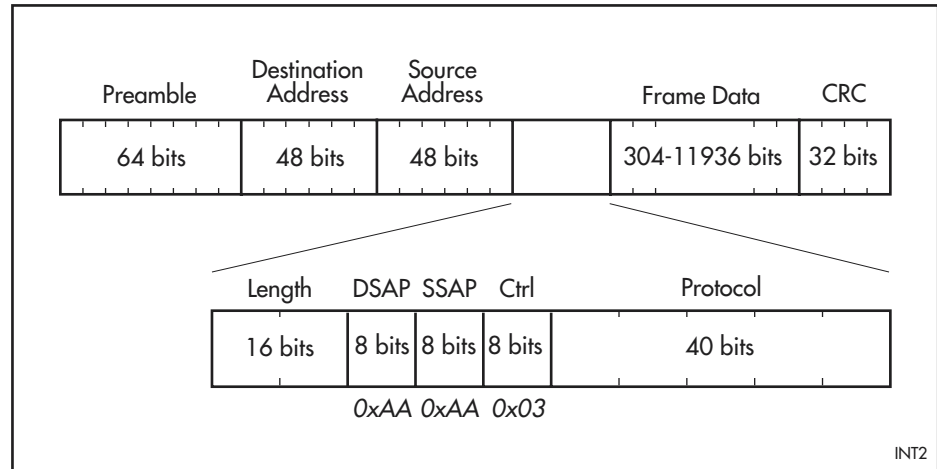
Although there may appear to be a conflict between the use of the type/length field for both a frame type and a length, in practice there is no conflict. The maximum length of an Ethernet frame (including the preamble, addresses and the type/length field) is 1514 octets, so the maximum value of the type/length field as a length is 1500 octets. Ethernet types are assigned values greater than 1500. In the early days of Ethernet, some protocol types were assigned values below 1500, but these have since become obsolete.

When the IEEE introduced the standard that replaced the type field with a length field, parts of the networking community still wanted a way to specify that a particular Ethernet frame was a certain protocol type, without having to implement all of the IEEE Standard 802.2.

IEEE Standard 802.2 defines the two octets after the type/length field as *Service Access Points*, or SAPs, one for the source of the packet and one for the destination. A special SAP value (0xAA or 170 decimal) was defined to indicate that the packet containing this SAP value would use the *SubNetwork Access Protocol* (SNAP) mechanism. In IEEE Standard 802.2, the one or two octets after the SAPs are defined as the control field. For the SNAP format, this is defined



as the single octet 0x03, used to indicate an “unnumbered information” frame. The SNAP format then defines the next 5 octets as a protocol type. Values in this field define the different protocols. The following figure shows the format of an Ethernet frame with SNAP encapsulation.



The router supports the following encapsulation formats:

- Ethernet – type/length field used as a type
- 802.2 – use of IEEE Standard 802.2 with SAPs
- SNAP – use of the SNAP SAP
- Novell (referred to by Novell as *raw 802.3*)—802.2 format packet with destination and source SAPs of 0xFF, but without the other fields of a true 802.2 header

For the correct operation of a software module, Ethernet drivers must receive packets with the appropriate encapsulation and forward them to that module. The packets are specified by an encapsulation format and a discriminator. The following table lists discriminators for each encapsulation format.

Format	Discriminator	Length (octets)
Ethernet	Ethernet type	2
802.2	Destination SAP	1
SNAP	SNAP discriminator	5
Novell	-	-

When a module specifies that the Ethernet drivers receive packets with the Novell format, a discriminator is not required.

When routing DECnet, a router must use a MAC address determined from the DECnet address assigned to the router, rather than the globally unique address assigned by the router manufacturer. With routers from many manufacturers, when DECnet routing is enabled the router must be rebooted so that the new MAC address can take effect. This can be disruptive to network operation. To avoid this, the router can use more than one MAC address simultaneously.

## Configuration

An Ethernet interface on the router is automatically configured by the software modules when the router starts up. No user configuration of the Ethernet interfaces is required, except to enable software modules to use the interface. This is achieved by adding a software module interface and using the clause

```
interface=ethn
```

where  $n$  is the number of the Ethernet interface being configured.

The modules in the router that are configured to use an Ethernet interface, and the encapsulations used on an interface, can be displayed with the command

```
show eth=n configuration
```

where  $n$  is the number of the Ethernet interface.

A feature of Ethernet is the ability to send packets to more than one station at a time, using multicast addresses. The multicast addresses required by a software module are automatically entered into the list of receive addresses by that module. No action by the user is required.

Note that the list includes the broadcast address and any unicast addresses specified by the software modules that have configured to the Ethernet interface. Unicast addresses are distinguishable from multicast addresses by their first octet. The first octet of a unicast address is even, whereas for a multicast address it is odd. The broadcast address is a special multicast address that is received by all stations on an Ethernet. The router is always configured to receive broadcast packets, even if no software modules are using the interface, so the list always includes the broadcast address.

Eth interfaces on the router can also apply a VLAN tag to frames that they transmit. For more information, see [“VLAN Tagging on Eth Interfaces” on page 21-28 of Chapter 21, Internet Protocol \(IP\)](#).

The default MAC address used by the Ethernet interface can be displayed with the command:

```
show eth[=n] macaddress
```

where  $n$  is the number of the Ethernet interface being configured.

The addresses that the router is configured to receive can be displayed with the command:

```
show eth[=n] receive
```

where  $n$  is the number of the Ethernet interface being configured.

The router maintains a number of counters for each Ethernet interface. These counters are objects in three standard MIBs and the router's enterprise MIB. For more information on MIBs, see [Chapter 54, Simple Network Management Protocol \(SNMP\)](#).

The counters are grouped into categories depending on the MIB to which they belong. The following table lists categories maintained for Ethernet interfaces.

Category	Group	MIB table	RFC
INTERFACE	Interfaces	Interfaces	1213
INTERFACE	Generic interfaces	Interface extensions	1229
DOT3STAT	Transmission	Dot 3 statistics	1398
COLLISION	Transmission	Collision statistics	1398
DIAGNOSTIC	Enterprise MIB	General	-
DIAGNOSTIC	Enterprise MIB	SONIC	-

Counters from each of these four categories are displayed with the command:

```
show eth[=n] counter [=category]
```

where *n* is the number of the Ethernet interface and *category* is one of the four categories of counter. If a category is not specified, all categories are displayed.

The counters in each category may be cleared to zero with the command:

```
reset eth[=n] counter [=category]
```

where *n* is the number of the Ethernet interface and *category* is one of the four categories of counter. If a category is not specified, all counters are cleared.

Using the [reset eth counters command on page 9-37](#) to clear the counters does not clear the MIB counters themselves. Instead, the MIB counter contents are copied to offset storage locations that are subtracted from the MIB counters before being displayed by the [show eth counters command on page 9-62](#).

The state of the connection between the Ethernet interface and network can be displayed with the command:

```
show eth[=n] state
```

where *n* is the number of the Ethernet interface. The command shows whether there is a working link between the Ethernet interface and the Ethernet and, if so, the link's capabilities.

The Ethernet interfaces on the router can be reset with the command:

```
reset eth=n
```

where *n* is the number of the Ethernet interface being reset. The Ethernet interface must be specified. A complete reset of the Ethernet interface is carried out with this command.

**Important** Data being sent or received when the Ethernet interface is reset is lost.

100Mbps Ethernet interfaces normally negotiate automatically with the network to decide which link speed (bit rate and duplex mode) to use. This allows 100Mbps Ethernet interfaces to operate with both 10Mbps and 100Mbps networks. If desired, auto-negotiation can be disabled and a particular link speed can be specified for a 100Mbit Ethernet interface with the command:

```
set eth=n speed={autonegotiate|10mhalf|10mfull|100mhalf|100mfull}
```

**Important** Setting an inappropriate link speed can interfere with the operation of other devices on the network.

For an Ethernet interface connected to a WAN connection, it may be necessary to limit the bandwidth on the base Ethernet port. To impose a maximum bandwidth limit of 128kbps or greater, measured in kbps, use the command:

```
set eth=n maxbandwidth=0..100000
```

By default this bandwidth limiting feature is off, so that it does not limit the bandwidth transmitted. To turn off the bandwidth limiting, use the command:

```
set eth=n maxbandwidth=0
```

This command is valid for some interfaces. When it is not valid for a particular interface an error message is displayed.

## Synchronous Interfaces

---

Synchronous or serial interfaces provide an alternative form of communication to Ethernet. Synchronous interfaces allow simultaneous communication in both the incoming and outgoing directions from the router. Synchronous interfaces typically provide a direct connection between two network devices, although with the use of special procedures, a number of devices can be connected to a single *multidrop* line. The router does not support multidrop lines.

Synchronous interfaces are normally used to provide wide area networking. In this situation, the synchronous interface on the network equipment, such as the router, is connected to equipment belonging to the telecommunications provider. This equipment is typically a modem, or for newer installations, a Network Terminating Unit (NTU). The NTU provides access either directly to another site located some distance away, or into a network run by the network provider.

The router supports synchronous interfaces with speeds of up to 2.048 Mbps, also known as E1. Different numbers of synchronous interfaces are provided on different models of the router. See the Hardware Reference for details of the number, types and physical characteristics of synchronous interfaces on each model of the router.

A feature of synchronous interfaces is that besides having circuits for the data being received and transmitted on the line, circuits also exist for clock signals. An item of network equipment must provide the clock on these circuits, and this is one of the main functions of the modem or NTU to which the synchronous interface is connected. However, in a small number of circumstances, it is useful to be able to generate the clock from within the router. This is especially true for connections between the router and host machines running X.25, where the router and host might be in the same room and no modems or NTUs are provided.

The router automatically generates a clock signal when a DCE transition cable is connected to a synchronous interface (see the Hardware Reference for details of how to construct a cable). The clock speed is set with the command:

```
set syn=n speed=speed
```

## Encapsulations

All encapsulations used by the router on synchronous interfaces are themselves encapsulated using the HDLC (High-level Data Link Control) protocol. The HDLC protocol has the following features:

- Data comes in frames, delimited by special characters called flags.
- When a frame is not being sent, the sender transmits flags continually. This means that there is constant activity on any synchronous line that is running properly.
- The first bytes of data in a frame are interpreted as an address and a control field. The address may or may not have any special meaning, depending on the encapsulation, but the control field usually has some meaning.
- The last bytes of data in a frame are a CRC (Cyclic Redundancy Check) for detecting errors in the frame.

The data in the frame is interpreted by the device receiving the frame depending on the encapsulation type.

Different encapsulations use different methods to distinguish the different layer 3 protocol types. The router currently supports three encapsulations for synchronous interfaces: Frame Relay, Point-to-Point (PPP) and Link Access Procedure for B Channels (LAPB).

## Modem Control Signals

Modem control signals can be configured so that output signals follow input signals or are permanently on or off. The modem control signals are grouped in the associated pairs RTS/CTS and DTR/DSR for RS-232/V.35 and C/I for X.21. The transition cable connected to the synchronous interface determines which pair or pairs of signals are being controlled. Changing from DTE mode to DCE mode by changing the transition cable alters which signal of the pair is the input and which is the output. It is possible to independently control the signals for each mode and to control CD in RS-232/V.35 DCE mode.

The actual modem control outputs that are present and which may be configured depends upon the transition cable used. The following table lists the modem control signals available for each transition cable type.

Transition Cable	Modem Control Outputs	Modem Control Inputs
RS-232 DTE	RTS, DTR	CTS, DSR
V.35 DTE	RTS, DTR	CTS, DSR
X.21 DTE	C	I
RS-232 DCE	CTS, DSR, CD	RTS, DTR
V.35 DCE	CTS, DSR, CD	RTS, DTR
X.21 DCE	I	C

For each transition cable type, one of the modem control inputs is used to determine the operational status of the interface. This status is the `ifOperStatus` object in the interface MIB for the interface and can be displayed using the **show interface** command. For the `ifOperStatus` of a SYN interface to be shown as “Up” the following conditions must apply:

- the SYN interface must be enabled
- a higher layer (e.g. PPP) must be attached to the SYN interface
- the relevant modem control input signal must be asserted.

The following table shows modem control input signals that controls the `ifOperStatus` for each transition cable type.

Cable type	Input signal
RS-232 DTE	CD
V.35 DTE	CD
X.21 DTE	I
RS-232 DCE	DTR
V.35 DCE	DTR
X.21 DCE	C

To avoid spurious `ifOperStatus` transitions due to “glitches” (changes of short duration) on the modem control input signal, the transitions are subjected to some hysteresis. The modem control signal must be asserted continuously for a period of at least one second before the `ifOperStatus` is changed to “Up”, and the modem control signal must be negated continuously for a period of at least two seconds before the `ifOperStatus` is changed to “Down”.

## Configuration

The encapsulation to be used on a synchronous interface is set by using the **create** command for the appropriate module, specifying the desired synchronous interface as the value of the **over** parameter. For example, to use the PPP encapsulation on synchronous interface 1 and to use the name PPP1, use the command:

```
create ppp=1 over=syn1
```

This creates the logical interface `ppp1` attached to synchronous interface 1. This command does not work when there is already an instance of the PPP module numbered 1. The layer 2 module instance number and the synchronous interface number need not be the same, but it simplifies management when they are. For a complete description of attaching layer 2 module instances to +synchronous interfaces, see [Chapter 14, Point-to-Point Protocol \(PPP\)](#), [Chapter 13, Frame Relay](#) and [Chapter 12, X.25](#). See “Naming Interfaces” on [page 9-4](#) for an explanation of the convention used to name interfaces on the router.

The configuration of a synchronous interface may be changed with the command:

```
set syn=n [c={on|off|i}] [cd={on|off|dtr}] [cts={on|off|rts}]
[dsr={on|off|dtr}] [dtr={on|off|dsr}] [I={on|off|c}]
[maxoqlenN=max-queue] [mintxint=min-interval] [rts={on|
off|cts}] [speed=speed]
```

The interface number must be specified, but each of the other parameters is optional. The **speed** parameter sets the clock rate of the interface if it is generating clocks and informs the router of the clock rate if it is receiving clocks. Note that it is not strictly necessary to enter the interface clock speed if it is receiving clocks, but it is advisable as routing modules may need to know this to correctly set routing metrics for the interface. The **maxoqlen** parameter sets the maximum length of the output queue. It is useful to set a maximum output queue length to prevent a heavily loaded interface from causing buffer allocation problems. The **mintxint** parameter sets the minimum time interval between transmitted frames. Setting a minimum interval between transmitted frames may be required when the device at the other end of the synchronous link is unable to receive frames that are transmitted back-to-back. The default values of these parameters set a limit of 100 on the output queue and no minimum delay between transmitted frames. The **c**, **cd**, **cts**, **dsr**, **dtr**, **i** and **rts** parameters are used to set the modem control signals to **on**, to **off**, or to follow the corresponding input signals. The default is **on**.

The configuration of a synchronous interface may be displayed with the command:

```
show syn [=n]
```

where *n* is the number of the synchronous interface. The interface number is optional. If it is omitted the configuration of all synchronous interfaces is displayed.

The router maintains a number of counters for each synchronous interface. The counters are objects in two standard MIBs and the router's enterprise MIB. For more information about SNMP and MIBs, see [Chapter 54, Simple Network Management Protocol \(SNMP\)](#).

Counters are grouped into categories depending on the MIB to which they belong. The following table lists the categories maintained for synchronous interfaces.

Category	Group	MIB table	RFC
INTERFACE	Interfaces	Interfaces	1213
SYN	Transmission	Synchronous port	1659

MIB counters for a synchronous interface may be displayed by using the command:

```
show syn [=n] counter [=category]
```

where *n* is the number of the synchronous interface and *category* is one of the two counter categories. If a category is not specified, all categories are displayed.

Objects from the general input and output signal tables (see RFC 1659) are displayed by the **show syn** command.

The counters in each category may be cleared to zero by using the command:

```
reset syn [=n] counter [=category]
```

where *n* is the number of the synchronous interface and *category* is one of the two counter categories. If a category is not specified, all counters are cleared.

Using the **reset syn counters** to clear the counters does not clear the MIB counters themselves. Instead, the MIB counter contents are copied to offset storage locations that are subtracted from the MIB counters before being displayed by the **show syn counter** command on page 9-81.

Each synchronous interface may be enabled or disabled with the commands:

```
enable syn=n  
disable syn=n
```

where *n* is the number of the synchronous interface. When a synchronous interface is disabled, it does not transmit or receive data. The default state of an interface is enabled. To reset a synchronous interface, use the command:

```
reset syn=n
```

where *n* is the number of the synchronous interface. This is equivalent to an **disable syn** command followed by an **enable syn** command.

Data that is being received or transmitted is lost when the synchronous interface is disabled or reset.

## Asynchronous Interfaces

---

All models of the router have at least one asynchronous interface, or port. This is a standard RJ45, DB9 male or DB9 female connector wired as a DTE (*Data Terminating Equipment*) interface. The asynchronous ports are identified by number, and are numbered sequentially starting from 0. The first interface is called asyn0.

All asynchronous ports use the RS-232C standard. At least four modem control lines are provided with each interface, and these are normally used as DTR, RTS, CTS, and CD. More information about asynchronous ports is in the Hardware Reference for the router.

Asynchronous ports are normally used to connect terminals or modems to the router. The cable types required to do this are described in the Hardware Reference. In general, most VT100-compatible terminals require a *crossed* cable (DTE-to-DTE).

**Important** The term *crossed* refers to the fact that the data pins (TxD and RxD) on the connector at one end of the cable are connected to the opposite pins (RxD and TxD respectively) on the connector at the other end of the cable. This is necessary because both the terminal and the router have DTE interfaces.

An asynchronous port can be configured as a network printer port using the Line Printer Daemon (LPD) protocol. See [Chapter 61, Line Printer Daemon \(LPD\)](#) for more information. To use LPD, IP must be enabled and configured on all routers in the network providing access to the LPD print queues. See [Chapter 21, Internet Protocol \(IP\)](#) for more information.

An asynchronous port can be configured as a stream printer port using the stream printing service. See [Chapter 62, Stream Printing](#) for more information. To use stream printing, IP must be enabled and configured on all routers in the network providing access to the stream printer. See [Chapter 21, Internet Protocol \(IP\)](#) for more information.

An asynchronous port can be permanently assigned. A permanent assignment is a mechanism for assigning one port on the router to another port on the router or to a port on another router in the network. Normally the ports are on different routers, and the assignment provides a permanent pipe through the network. Permanent assignments can be used for network printing, to connect an asynchronous port to a host computer port, or to connect the asynchronous port



on a data logger to a host computer to capture data. See [Chapter 63, Permanent Assignments](#) for more information. To use permanent assignments, IP must be enabled and configured on the routers at both ends of the link. See [Chapter 21, Internet Protocol \(IP\)](#) for more information.

Asynchronous ports may also be used as network interfaces.

## Encapsulations

By default, no encapsulation is used on asynchronous ports. Data is transmitted and received as a clear character stream. This is appropriate for remote terminal or terminal emulation access and remote printing facilities.

The router also supports the asynchronous PPP encapsulation, SLIP (Serial Line Internet Protocol) encapsulation, and CSLIP (Compressed SLIP using Van Jacobson's header compression) encapsulation.

If an asynchronous port is assigned to an Asynchronous Call Control (ACC) call for dial-up connections, the call definition specifies the encapsulation to use or that the user is to be prompted to select an encapsulation during the login process. See [Chapter 18, Asynchronous Call Control](#) for more information about defining calls and specifying encapsulations.

Asynchronous PPP encapsulation is used when the port is assigned to a PPP interface as a network interface with the command:

```
create ppp=ppp-interface over=acc-callname
```

where *callname* is the name of the asynchronous call. See [Chapter 14, Point-to-Point Protocol \(PPP\)](#) for more details about creating PPP interfaces.

SLIP encapsulation is used when the port is assigned to the IP module as an IP interface with the command:

```
add ip interface=slipn ...
```

where *n* is the number of the asynchronous port. See [Chapter 21, Internet Protocol \(IP\)](#) for more details about creating IP interfaces.

## Configuration

Each asynchronous port can be individually configured to suit a wide range of different terminal types. The characteristics of a port can be changed by using the command:

```
set asyn=asyn-number option
```

Options for each port are listed in the following table.

### Asynchronous port options

Option	Description
Attention	Sets the attention character used to return from a virtual terminal session to the router prompt.
Cdcontrol	Controls the way the router interprets the state of the DCD input signal. If <b>cdcontrol</b> is set to <b>ignore</b> the router ignores the state of the DCD input signal. If <b>cdcontrol</b> is set to <b>connect</b> the router terminates existing connections when the DCD signal is deasserted (i.e. when a modem disconnects). If <b>cdcontrol</b> is set to <b>online</b> and the interface is configured as a printer port, output is not sent to the interface unless the DCD signal is asserted.

**Asynchronous  
port options**

Option	Description
Databits	Sets the number of data bits per character transmitted by the port.
Defaultservice	Configures the port to automatically connect to a service when a user types anything at the terminal or an attached modem asserts DCD.
Dtrcontrol	Controls the way the router controls the state of the DTR output signal. If <b>dtrcontrol</b> is set to <b>connect</b> the router asserts DTR for the duration of the connection. If <b>dtrcontrol</b> is set to <b>on</b> or <b>off</b> , the DTR line is driven to the designated state.
Echo	Enables or disables the echoing of each character entered at a terminal.
Flow	Sets the flow control mechanism used for the port in both the receive and transmit directions. If <b>flow</b> is set to <b>none</b> the router ignores all incoming flow control characters and lead transitions. If <b>flow</b> is set to <b>character</b> the router uses <b>xon/xoff</b> flow control. If <b>flow</b> is set to <b>hardware</b> the router uses the <b>rts/cts</b> lines for flow control.
Inflow	Sets the flow control mechanism used for the port in the receive direction. If <b>flow</b> is set to <b>none</b> , the router ignores all incoming flow control characters and lead transitions. If <b>flow</b> is set to <b>character</b> , the router uses <b>xon/xoff</b> flow control. If <b>flow</b> is set to <b>hardware</b> , the router uses the <b>rts/cts</b> lines for flow control.
History	Sets the number of commands saved for command line recall.
ldletimeout	Specifies a period of time, in seconds, for a terminal connection's dedicated TTY device idle timer. If the specified time period lapses since the last time the dedicated TTY device received data from the client, the connection is terminated, and the terminal screen displays the login prompt. If <b>0</b> or <b>off</b> are specified, the idle timer remains off, and the session must be explicitly terminated.
Ipaddress	Sets the IP address associated with the port. This parameter may be required when the port is used as a network interface using SLIP or PPP. See <a href="#">Chapter 18, Asynchronous Call Control</a> for more information.
Ipxnetwork	Sets the IPX network number associated with the port. This parameter may be required if the port is used as a network interface using PPP. See <a href="#">Chapter 18, Asynchronous Call Control</a> for more information.
Login	Enables or disables the ability to log into the asynchronous port.
Maxoqlen	Sets the maximum number of character buffers that are permitted on the transmit queue for the port.
Mtu	Sets the Maximum Transmission Unit (MTU), which is the maximum number of bytes per packet that may be transmitted by the port when it is used as a network interface. See <a href="#">Chapter 18, Asynchronous Call Control</a> for more information.
Name	Assigns a text string used to identify the port, such as the name of the person whose terminal is normally connected to the port, or where the terminal is located.
Outflow	Sets the flow control mechanism used for the port in the transmit direction. If <b>flow</b> is set to <b>none</b> , the router ignores all incoming flow control characters and lead transitions. If <b>flow</b> is set to <b>character</b> , the router uses <b>XON/XOFF</b> flow control. If <b>flow</b> is set to <b>hardware</b> , the router uses the <b>RTS/CTS</b> lines for flow control.
Page	Sets the number of lines of output displayed on the terminal before the router pauses and waits for the user to press a key to continue.
Parity	Sets the parity of each character transmitted by the port.

**Asynchronous  
port options**

Option	Description
Prompt	Sets the prompt to a string, the default prompt, or disables the prompt.
Secure	Controls whether a user must log in to the port before router commands can be accepted. See <a href="#">Chapter 40, User Authentication</a> for information about defining users and logging in to the router.
Service	Allocates the port to be a host port for a named service.
Speed	Sets the speed of the port, from 75 bps to 115200 bps. The terminal and port must be set to the same speed. Autobauding is also available, provided the attention character used is set to [Break]. In this mode the port automatically adjusts to the speed of the terminal that is attached, up to 19200 bps.
Stopbits	Sets the number of stop bits per character transmitted by the port.
Tentimervalue	The period, in milliseconds, over which the port bundles characters, when the port is in ten mode.
Type	Sets the terminal type to <b>vt100</b> or <b>dumb</b> . A <b>dumb</b> terminal is used for printing or terminals that do not support VT100 escape sequences.

**Asynchronous  
port defaults**

Asynchronous ports are initially configured with default values listed in the following table.

Option	Default setting
Attention	break
Cdcontrol	ignore
Databits	8
Defaultservice	false
Dtrcontrol	on
Echo	on
Flow	hardware
History	30
Idletimeout	0
Inflow	hardware
Ipaddress	none
Ipxnetwork	none
Login	on
Maxoqlen	0 (Unrestricted)
Mtu	1500
Name	asyn #
Outflow	hardware
Page	22
Parity	none
Prompt	default (CMD>)
Secure	on
Service	none
Speed	9600

Option	Default setting
Stopbits	1
Tentimervalue	100
Type	vt100

To display the complete configuration for a particular asynchronous port, use the command:

```
show asyn=asyn-number
```

To display the complete configuration for all asynchronous ports, use the command:

```
show asyn=all
```

To display summary details for a particular asynchronous port, use the command:

```
show asyn=asyn-number summary
```

To display summary details for all asynchronous ports, use the command:

```
show asyn=all summary
```

The router maintains a separate command history list for each asynchronous port, containing the last commands entered at the port. To display the history list, use the command:

```
show asyn=asyn-number history
```

## Session Timeout

If you disable an asynchronous port, users can still log into the port but will be logged out if the session is idle for a configurable length of time. Users can log in to the disabled port by sending it a break signal. To configure this timeout functionality:

### 1. Set the timeout period.

If you require a different timeout than the default of 60 seconds, use the command:

```
set asyn [enable=break] timeout=1..65535
```

The timeout only applies if **enable=break**, which is its default.

### 2. Manually disable the port.

Use the command:

```
disable asyn
```

Note that if you are logged into an asynchronous port to manage the switch, you cannot disable that port by typing the disable command. You have to run the disable command from a script or from a different session, such as a telnet session.

To log into the asynchronous port:

- Connect as normal to the port through a terminal emulator or modem. Then send a break signal to get a log in prompt. The method of sending a break signal depends on the terminal application.
- To start a new session after the port has timed out, send a break signal. This enables the port, which will then provide you with a log in prompt.

## Connecting a Modem to the Asynchronous Port

If a modem is connected, configure the router to make and/or accept calls via the modem. To set the **cdcontrol** parameter to **connect** and the **flow** parameter to **hardware**, enter the command:

```
set asyn cdcontrol=connect flow=hardware
```

If the terminal or modem is used with communications settings other than the default settings, then configure the asynchronous port to match the terminal or modem settings by using the **set asyn** command.

A port connected to a modem should always be set to a fixed speed matching that of the modem.

## MIB Counters

The router maintains a number of counters for each asynchronous port. The counters are objects in two standard MIBs and the router's enterprise MIB. Counters are grouped into categories depending on the MIB to which they belong. The following table lists the categories maintained for asynchronous ports.

Category	Group	MIB table	RFC
INTERFACE	Interfaces	Interfaces	1213
RS232	Transmission	Asynchronous port	1659
DIAGNOSTIC	Enterprise MIB	Asynchronous interface	-

For more information about SNMP and MIBs, see [Chapter 54, Simple Network Management Protocol \(SNMP\)](#).

To display the MIB counters for an asynchronous port, use the command:

```
show asyn [=n] counter [=category]
```

where *n* is the number of the asynchronous port and *category* is one of the three counter categories. If a category is not specified, all categories are displayed.

Objects from the general input and output signal tables (see RFC 1659) are displayed by the **show asyn** command.

To clear counters in each category to zero, use the command:

```
reset asyn [=n] counter [=category]
```

where *n* is the number of the synchronous port and *category* is one of the three counter categories. If a category is not specified, all counters are cleared.

Using the **reset asyn** command to clear the counters does not clear the MIB counters themselves. Instead, the MIB counter contents are copied to offset storage locations that are subtracted from the MIB counters before being displayed by the **show asyn** command.

To enable or disable each asynchronous port, use the commands:

```
enable asyn=n
disable asyn=n
```

where  $n$  is the number of the asynchronous port. When an asynchronous port is disabled it does not transmit or receive data. When the port is enabled, all configuration parameters are restored to the settings in effect prior to the port being disabled. The default state of an asynchronous port is enabled.

**Important** Data being received or transmitted when the asynchronous port is disabled or reset is lost.

To reset an asynchronous port, use the command:

```
reset asyn= $n$ 
```

where  $n$  is the number of the asynchronous port. Any current connections are disconnected and the configuration parameters are restored from nonvolatile storage.

To reset the command history, use the command:

```
reset asyn history
```

The specific commands to change the parameters of a particular asynchronous port are given in [“Command Reference” on page 9-26](#). As an example, to change the name of port 6 to “test” and the speed to 9600 bps, use the command:

```
set asyn=6 name=test speed=9600
```

All port configuration parameters are held in non-volatile memory and are retained over a power cycle.

## Autobauding

Asynchronous ports may be set to autobauding mode. In this mode the router adjusts the speed of the port to match the speed of the terminal attached to the port, up to a maximum speed of 19200 bps. For autobauding to work, the user should always press the [Enter] or [Return] key on the terminal several times until the router prompt appears on the screen. At this point the router has set the speed of the port. If a key other than [Enter] or [Return] is pressed while the router is setting the port speed, the speed may be incorrectly set. In this case, there is no response from the router or “garbage” characters appear on the terminal screen. To fix this, press [Break] two or more times, followed by [Enter] or [Return] several times.

Some terminals require the [Break] key to be held down for about a second to properly send a [Break]. Additionally, some terminals require a brief pause between multiple [Break]s.

Once the speed is set on an autobauding port, the router does not change it unless one of the following events occurs:

- The router is turned off.
- [Break] is pressed twice, in which case the router “forgets” the current speed and waits for [Enter] or [Return] to be pressed several times to set the speed again.
- The terminal is switched off. This sometimes has the effect of sending [Break]s to the router.

## Making Asynchronous Ports Respond More Quickly

When an asynchronous port is in *ten mode*, it bundles together the characters that it receives within a certain time period, instead of passing them one at a time to a higher protocol layer for processing. The time period over which characters are bundled is set by the *ten timer*.

Bundling reduces the load on the CPU by spreading the character processing overhead across several characters. If a remote terminal session is involved, bundling also reduces the number of packets on the network by sending more characters in each packet. However, bundling reduces terminal responsiveness.

A ten timer value of 100 milliseconds is generally a good compromise between responsiveness and processing overhead. If you need to increase the port's responsiveness, you can reduce the length of the ten timer, by using the command:

```
set asyn[=port-number] tentimervalue=20..100 [other optional  
parameters]
```

Unless you are logged in via the port you want to change, also specify the asynchronous port number.

The default **tentimervalue** is 100 milliseconds.

## Testing Serial Data Circuits

Wide area data circuits are normally leased from the Telecom supplier. A point-to-point circuit has an NTU or modem at each end. These normally allow some limited testing of the circuit to be done. Unfortunately, there are a large number of different types of NTU and modem, so it is not possible to predict the exact functionality. The following gives an indication of the basic features common to most modems and NTUs.

In the remainder of this section, the term 'NTU' is used exclusively.

**Carrier detect** This signal is normally available at the data interface of the NTU as well as being shown on a front panel LED. It must be present for the NTU to operate correctly. If this fails, it usually means that the data circuit is faulty or the NTU at the other end is not functioning. In either case, the Telecom supplier should be called to fix the problem. Some other possible names for this signal include RLSD, 109, CD and EQG. If the circuit quality is poor, this signal may have frequent short transitions. This results in poor link throughput.

**Loopback** This feature is not normally present as an indicator, but rather as one or more front panel buttons, sometimes associated with an LED to show that the NTU is in a test mode. The loopback functionality available varies from NTU to NTU, depending on the type, and exactly what has been selected at installation time. Loopbacks allow the data circuit to be tested in stages, by progressively looping back first the local NTU and then the remote NTU. If for instance, the remote NTU loopback fails, but the local loopback is successful, it indicates a fault in either the data circuit or the remote NTU and the Telecom supplier should be notified. If the remote and local tests are successful, it indicates that the problem is either in the remote NTU or the network equipment at the remote end. The tests should be reversed from the remote end to eliminate the remote NTU.

**Data indicators** These are front panel mounted LEDs on the NTU and can be used to see that data is flowing in both directions.

## Displaying Interfaces

---

The router stores information about interfaces as objects in the Interfaces Table of MIB-II, defined in RFC 1213 *Management Information Base for Network Management of TCP/IP-based internets: MIB-II*. To display the contents of the Interfaces Table, use the command:

```
show interface
```

To display detailed information about a specific interface, use the command:

```
show interface={ifindex|interface}
```

where *ifIndex* is the index of the interface in the Interfaces Table and *interface* is the interface name.

To display counters for all the interfaces, use the command:

```
show interface counter
```

To display data about average queue length and time spent in queue when priority routing is enabled, use the command:

```
show interface={ifindex|interface} debug=priorityqueue
```

For a detailed description of the objects in the Interfaces Table of MIB-II, see [Appendix C, SNMP MIBs](#).

## Interface Link Traps

---

When an interface changes to or from the “Down” state, an SNMP trap can be sent to any SNMP manager stations (trap hosts) that have been defined.

The general operation of link traps is defined in RFC 1157, *Simple Network Management Protocol*. In the typical multi-layered interface environment, each protocol layer for which an interface entry exists in the interface table can generate link up/down traps.

Since interface state changes tend to propagate through the protocol layers, multiple traps may be generated as the result of a single link failure. RFC 1573, *Evolution of the Interfaces Group of MIB-II*, resolves this issue by providing a mechanism for enabling and disabling link trap generation on a specific interface. This allows stacked interfaces to be configured so that only one trap is sent for a link transition.

Link traps are disabled by default on the router. Link traps can be enabled or disabled on a per-interface basis by using the commands:

```
enable interface linktrap
```

```
disable interface linktrap
```

To display current settings for link traps, use the command:

```
show interface
```

The potential exists in a large or busy network for a high volume of trap messages to be generated, especially if the network configuration involves dynamic interfaces created by ISDN or ACC calls. To set the maximum number of link traps generated per minute for each static interface or for all dynamic interfaces, use the command:

```
set interface traplimit
```



## Managing Interfaces with SNMP

---

Router interfaces can be enabled or disabled via SNMP by setting the *ifAdminStatus* object in the *ifTable* of MIB-II MIB to 'Up(1)' or 'Down(2)' for the corresponding *ifIndex*. When it is not possible to change the status of a particular interface the router returns an SNMP error message.

The router's implementation of the *ifOperStatus* object in the *ifTable* of MIB-II MIB supports two additional values—"Unknown(4)" and "Dormant(5)" (e.g. an inactive dial-on-demand interface).

**Important** An unauthorised person with knowledge of the appropriate SNMP community name could bring an interface up or down. Community names act as passwords for the SNMP protocol. Care should be taken when creating an SNMP community with write access to select a secure community name and to ensure that this name is known only to authorised personnel.

## Command Reference

---

This section describes the commands available on the router to configure and manage the Ethernet, synchronous and asynchronous interfaces on the router.

Some interface and port types mentioned in this chapter may not be supported on your router. The interface and port types that are available vary depending on your product model and whether an expansion unit (PIC, NSM) is installed. For more information, see the Hardware Reference for the router.

Some commands require IP and SNMP to be enabled and configured. See [Chapter 21, Internet Protocol \(IP\)](#) for a detailed description of the commands required to enable and configure IP. See [Chapter 54, Simple Network Management Protocol \(SNMP\)](#) for a detailed description of the commands required to enable and configure SNMP.

The shortest valid command is denoted by capital letters in the Syntax section. See [“Conventions” on page lxiv of About this Software Reference](#) in the front of this manual for details of the conventions used to describe command syntax. See [Appendix A, Messages](#) for a complete list of error messages and their meanings.

### connect asyn

---

**Syntax** Connect ASYn=*asyn-number*

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description** This command creates a new terminal session that connects a Telnet session or the terminal (for a router with more than one asynchronous port) directly to a physical asynchronous port. This lets you send commands directly to a device connected to the port. For example, this command can be used to access a modem connected to the port, to send modem commands directly to the modem to change its configuration.

**Examples** To connect to asynchronous port 0, use the command:

```
connect asy=0
```

**Related Commands** [connect](#)  
[disconnect](#)

---

## disable asyn

---

**Syntax**    `DISable ASYn=asyn-number`

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description**    This command disables a specific port that is currently enabled so that no data can be accepted or transmitted through it. By default, an asynchronous port is enabled.

If you are logged into an asynchronous port to manage the switch, you cannot disable that port by typing this command. You have to run the command from a script or from a different session, such as a telnet session.

**Examples**    To disable asynchronous port 3, use the command:

```
dis asy=3
```

**Related Commands**

- [enable asyn](#)
- [disable interface debug](#)
- [show interface](#)
- [reset asyn](#)
- [set asyn](#)
- [show asyn](#)

## disable interface debug

---

**Syntax**    `DISable INTerface={ifIndex|interface} DEBug=PRIOrityqueue`

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB.
- *interface* is a valid Ethernet or VLAN interface.

**Description**    This command disables debugging options for an interface.

The **debug** parameter disables IP priority queue debugging for Ethernet and VLAN interfaces. Priority queue debugging collects a statistical sampling of the lengths of the priority queues and the time packets spend in each queue in order to determine averages. The data can be seen with the **show interface priorityqueue** command.

**Examples**    To disable priority queue debugging on the interface eth0, use the command:

```
dis int=eth0 deb=prio
```

**Related Commands**    [enable interface debug](#)  
[show interface](#)  
[disable debug active](#) in Chapter 4, Configuring and Monitoring the System  
[show debug active](#) in Chapter 4, Configuring and Monitoring the System

## disable interface linktrap

---

**Syntax** `DISable INTerface={ifIndex|interface|DYNAMIC} LInktrap`

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface name

**Description** This command disables link up/down trap generation for the specified interface. Link up/down traps are disabled by default.

The **interface** parameter specifies the interface for which link traps are to be disabled.

The **dynamic** parameter handles the special case of dynamic interfaces that do not yet exist. If link traps are enabled for dynamic interfaces, a trap message is generated whenever a dynamic interface is created or destroyed. This is disabled by default. If **dynamic** is specified, link trap generation is disabled for the creation and destruction of dynamic interfaces. Valid interfaces are:

- eth (such as eth0, eth0-1)
- FR (such as fr0)
- PPP (such as ppp0, ppp1-1)
- VLAN (such as vlan1, vlan1-1)

To see a list of current interfaces, use the **show interface** command.

IP and SNMP must be enabled and correctly configured to generate traps. See [Chapter 21, Internet Protocol \(IP\)](#) for a detailed description of the commands required to enable and configure IP. See [Chapter 54, Simple Network Management Protocol \(SNMP\)](#) for a detailed description of the commands required to enable and configure SNMP.

**Examples** To disable link trap generation for interface ppp0, use the command:

```
dis int=ppp0 li
```

**Related Commands**

- [enable interface linktrap](#)
- [set interface traplimit](#)
- [show interface](#)

## disable syn

---

**Syntax**    DISable SYN=*n*

where *n* is the number of the synchronous interface.

**Description**    This command puts the synchronous interface into a state where it does not transmit or receive any frames. The interface number must be specified.

Data being received or transmitted when the synchronous interface is disabled or reset is lost.

**Examples**    To disable synchronous interface 2, use the command:

```
dis syn=2
```

**Related Commands**    [enable syn](#)  
                          [reset syn](#)  
                          [show syn](#)

## disable syn debug

---

**Syntax**    DISable SYN=*n* DEBug

where *n* is the number of the synchronous interface.

**Description**    This command disables the output of debug messages for a synchronous interface. The output of debug messages is disabled by default.

**Examples**    To disable debugging messages on synchronous interface 1, use the command:

```
dis syn=1 debug
```

**Related Commands**    [enable syn debug](#)  
                          [show syn](#)

---

## enable asyn

---

**Syntax**    `ENABle ASYn=asyn-number`

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description**    This command enables a specific asynchronous port. The port must currently be disabled. Data is accepted and/or transmitted via the specified port. By default, an asynchronous port is enabled.

**Examples**    To enable asynchronous port 0, use the command:

```
ena asyn=0
```

**Related Commands**    [disable asyn](#)  
[disable interface debug](#)  
[reset asyn](#)  
[set asyn](#)  
[show asyn](#)  
[show interface](#)

---

## enable interface debug

---

**Syntax**    `ENABle INTerface={ifIndex|interface} DEBug=PRIOrityqueue`

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB.
- *interface* is a valid Ethernet or VLAN interface.

**Description**    This command enables debugging options for an interface, and is disabled by default.

The **debug** parameter enables IP priority queue debugging for Ethernet and VLAN interfaces. Priority queue debugging starts a statistical sampling of the lengths of the priority queues and time a packet spends in each queue in order to calculate averages. The data can be seen with the **show interface priorityqueue** command. When priority queue debugging is enabled, messages are also displayed that relate to the interface.

**Examples**    To enable priority queue debugging on the interface eth0, use the command:

```
ena int=eth0 deb=prio
```

**Related Commands**    [disable interface debug](#)  
[show interface](#)  
[disable debug active](#) in Chapter 4, Configuring and Monitoring the System  
[show debug active](#) in Chapter 4, Configuring and Monitoring the System

## enable interface linktrap

---

**Syntax** ENABle INTerface={*ifIndex*|*interface*|DYNAMIC} LInktrap

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface

**Description** This command enables link up/down traps to be generated for an interface. Link up/down traps are disabled by default.

The **interface** parameter specifies the interface for which link traps are to be enabled.

The **dynamic** parameter handles the special case of dynamic interfaces that do not yet exist. If link traps are enabled for dynamic interfaces, a trap message is generated whenever a dynamic interface is created or destroyed. This is disabled by default. The **dynamic** parameter enables link trap generation for the creation and destruction of dynamic interfaces. Valid interfaces are:

- eth (such as eth0, eth0-1)
- FR (such as fr0)
- PPP (such as ppp0, ppp1-1)
- VLAN (such as vlan1, vlan1-1)

To see a list of current interfaces, use the **show interface** command.

IP and SNMP must be enabled and correctly configured to generate traps. See [Chapter 21, Internet Protocol \(IP\)](#), and [Chapter 54, Simple Network Management Protocol \(SNMP\)](#) for these commands.

**Examples** To enable link trap generation for the interface with an ifIndex of 1, use the command:

```
ena int=1 li
```

**Related Commands** [disable interface linktrap](#)  
[set interface traplimit](#)  
[show interface](#)



## enable syn

---

**Syntax**    `ENABle SYN=n`

where *n* is the number of the synchronous interface

**Description**    This command is used to reinitialise and enable a synchronous interface that has been disabled. The default state of a synchronous interface is enabled. The interface number must be specified.

**Examples**    To enable synchronous interface 2, use the command:

```
ena syn=2
```

**Related Commands**    [disable syn](#)  
[reset syn](#)  
[show syn](#)

## enable syn debug

---

**Syntax**    `ENABle SYN=n DEBbug`

where *n* is the number of the synchronous interface

**Description**    This command enables output of debug messages for a synchronous interface. The output of debug messages is disabled by default.

**Examples**    To enable debugging messages on synchronous interface 1, use the command:

```
ena syn=1 deb
```

**Related Commands**    [disable syn debug](#)  
[show syn](#)

---

## purge asyn

---

**Syntax** PURge ASYn={*asyn-number*|ALL}

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description** This command resets a specific asynchronous port to the factory default configuration. If **all** is specified, all ports are reset and all port configurations are lost.

**Examples** To purge the configuration of all asynchronous ports, use the command:

```
pur asy=all
```

**Related Commands** [disable asyn](#)  
[enable asyn](#)  
[reset asyn](#)  
[reset asyn counter](#)  
[reset asyn history](#)  
[set asyn](#)  
[show asyn](#)

---

## reset asyn

---

**Syntax** RESET ASYn=*asyn-number*

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description** This command resets a specific asynchronous port. If a port number is not specified, then the command applies to the port from which it is issued. If a port number is specified, the command applies to the specified port. The port configuration is restored from nonvolatile storage. Any existing connections are terminated.

**Examples** To reset asynchronous port 3, use the command:

```
reset asy=3
```

**Related Commands** [disable asyn](#)  
[enable asyn](#)  
[disable interface debug](#)  
[reset asyn counter](#)  
[reset asyn history](#)  
[set asyn](#)  
[show asyn](#)

---

## reset asyn counter

---

**Syntax** `RESET ASYn=asyn-number COUnters[={Diagnostic|Interface|Rs232}]`

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description** This command simulates an asynchronous counter reset for the specified asynchronous port. Subsequent [show interface](#) commands, then display only the counter increments since the last **reset asyn** command. SNMP requests however, will still return the counter's true value.

If a port is not specified, then the router uses the port number from which the command was entered.

If a category is specified, then the command will apply to the counter in that particular category. If a category is not entered, then the command will apply to the counters for all categories. For a description of the categories, see the description of the **show syn** command.

The control signal transition counters displayed by the **show syn** are reset along with the other counters in the RS-232 category.

**Examples** To reset the interface counter for asynchronous port 3, use the command:

```
reset asy=3 cou=i
```

**Related Commands** [reset asyn](#)  
[reset asyn history](#)  
[show asyn](#)

## reset asyn history

---

**Syntax** RESET ASYn=*asyn-number* History

where *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description** This command clears all commands from the command history for the specified asynchronous port. If a port number is not specified then the command applies to the port or TTY device from which the command is issued. If a port number is specified, the command applies to the specified port.

Port history is automatically reset during the login and logoff processes.

**Examples** To reset the command history for the asynchronous port to which the terminal is connected, use the command:

```
reset asy h
```

To reset the command history for asynchronous port 3, use the command:

```
reset asy=3 h
```

**Related Commands** [reset asyn](#)  
[reset asyn counter](#)  
[show asyn](#)

## reset eth

---

**Syntax** RESET ETH=*n*

where *n* is the number of the Ethernet interface

**Description** This command resets the specified Ethernet interface. The interface must be specified.

Data being transmitted or received by the Ethernet interface is lost. This may affect the operation of some of the protocol modules in the router.

**Examples** To reset Ethernet interface 0, use the command:

```
reset eth=0
```

**Related Commands** [reset eth counters](#)

---

## reset eth counters

---

**Syntax** `RESET ETH[=n] COunters[={COLlision|DIAGnostic|DOT3stat|INTERface}]`

where *n* is the number of the Ethernet interface

**Description** This command simulates an Ethernet counter reset for the specified Eth interface. If a number is not specified, the command applies to all Eth interfaces. If a category is specified, the command applies to that particular category. If one is not specified, then the command applies to all categories.

Subsequent [show eth counters](#) commands then display only the counter increments since the last **reset eth** command. However, SNMP requests still return the counter's true value.

**Examples** To reset the interface counters for Ethernet interface 0, use the command:

```
reset eth=0 cou=int
```

**Related Commands** [show eth counters](#)

---

## reset interface counters

---

**Syntax** `RESET INTERface[={ifIndex|interface}] COunters`

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface

**Description** This command simulates an interface counter reset. This enables subsequent [show interface](#) commands to display only the counter increments since the last **reset interface** command. SNMP requests however, still return the counter's true value.

- eth (such as eth0, eth0-1)
- FR (such as fr0)
- PPP (such as ppp0, ppp1-1)
- VLAN (such as vlan1, vlan1-1)

To see a list of current interfaces, use the **show interface** command.

**Examples** To reset the ppp0 interface MIB counters, use either of the following commands:

```
reset int=ppp0 cou
reset int=1 cou
```

**Related Commands** [show interface](#)

## reset syn

---

**Syntax**    RESET SYN=*n*

where *n* is the number of the synchronous interface.

**Description**    This command is equivalent to the command sequence:

```
disable syn=n
enable syn=n
```

Ideally, this command should never be required but some circumstances may require it. The interface number must be specified.

Data being received or transmitted when the synchronous interface is disabled or reset is lost.

**Examples**    To reset synchronous interface 2, use the command:

```
reset syn=2
```

**Related Commands**    [disable syn](#)  
                          [enable syn](#)  
                          [reset syn](#)  
                          [show syn](#)

---

## reset syn counters

---

**Syntax** `RESET SYN[=n] COUnTERS[={INTERface|SYN}]`

where *n* is the number of the synchronous interface

**Description** This command resets the MIB counters for a synchronous interface. The interface number is optional. If the interface is not specified, then the counters for all interfaces are reset. If a category is specified, the counters in that category are cleared. If the category is not entered, then the counters for all categories are reset. For a description of the categories, see the **show syn counter** command.

Note that this command does not reset the counters themselves. It first initiates a snapshot of the current counter values, then each time the relevant *Show* command is run each of the snapshot values are subtracted from each of these from the new counter values. This gives the same set of values from the various show commands as would display had the counter been reset.

Where there is an equivalent SNMP GET function, the router supplies the counters' original values, and not the ones displayed by the **Show** commands.

The control signal transition counters displayed by the **show syn counter** command are reset along with the other counters in the SYN category.

**Examples** To reset the **syn** counter for synchronous interface 2, use the command:

```
reset syn=2 cou=syn
```

**Related Commands** [show syn counter](#)

## set asyn

**Syntax** SET ASYn[=*asyn-number*] [Attention={Break|*alphabetical control char*|^|None}] [CDcontrol={Connect|Ignore|Online}] [DATAbits={5|6|7|8}] [DEFAultservice={ON|OFF|YES|NO|True|False}] [DTrcontrol={Connect|OFF|ON}] [Echo={ON|OFF|YES|NO|True|False}] [ENable={BREAK|NONE}] [Flow={Character|HARdware|None}] [History=0..99] [IDLEtimeout={10..4294967294|OFF|0}] [INFlow={Character|HAreware|None}] [IPaddress={*ipadd*|NONE}] [IPXnetwork=*network*] [LOGin={ON|OFF|YES|NO|True|False}] [MAXoqlen=0..4294967295] [MTu=40..1500] [NAME=*name*] [OUTFlow={Character|HARdware|None}] [PAGE={0..99|OFF}] [PARity={Even|Mark|None|Odd|SPace}] [PRompt={*prompt*|DEFAult|OFF}] [SECure={ON|OFF|YES|NO|True|False}] [SERvice={*service-name*|None}] [SHELLserver={ON|OFF}] [SPEed={AUTO|75|110|134.5|150|300|600|1200|1800|2000|2400|4800|9600|14400|14.4K|19200|19.2K|28800|28.8K|38400|38.4K|57600|57.6K|115200|115.2K}] [STOpbits={1|2}] [TENTimervalue=20..100] [TIMEout=1..65535] [TYpe={Dumb|VT100}]

where:

- *asyn-number* is the number of the port. Ports are numbered sequentially starting with 0.
- *alphabetical control char* is the '^' character followed by any alphabetical character in upper or lower case such as ^A, ^b, ^z.
- *ipadd* is an IP address in dotted decimal notation.
- *network* is a valid Novell network number, expressed as a hexadecimal number. Leading zeros may be omitted.
- *name* is a character string 1 to 15 characters long. If the string contains spaces, it must be in double quotes. The string is not case-sensitive.
- *prompt* is a character string 1 to 15 characters long. If the string contains spaces, it must be in double quotes. The string is not case-sensitive.
- *service-name* is the name of a service 1 to 15 characters long, with no embedded spaces. The first character must be alphabetic (A–Z). The name is not case-sensitive.

**Description** This command sets characteristics of asynchronous ports. If a port is not specified, then the command applies to the port on which it is issued. If a port number is specified, the command applies to the specified asynchronous port. Multiple options may be specified in the same command.

If the **set asyn** command is issued from a port with User privileges, the port number and the options **ipaddress**, **ipxnetwork**, **mtu**, **service**, and **secure** cannot be specified.

For a Telnet connection only, the options **history**, **page**, **prompt**, **type**, and **idletimeout** may be used to alter the behaviour of the dedicated TTY device.

The **set asyn** command may be rejected if there is no hardware present in the router for the specified port number, the port is currently assigned or a port-pair in a permanent assignment, or the port is a printer port and the printer is active.



The change takes place immediately and the new value is stored in nonvolatile memory.

The **attention** parameter specifies the character used to return from an active session (e.g. a Telnet connection) to the router prompt. If “^” with an alphabetical character is specified then the attention character is the [Ctrl] key and the specified alphabetical character key held down simultaneously. Similarly, “^[” means the attention character is set to the [Ctrl] key with the “[” key. The default is **break** (the [Break] key) for asynchronous ports, and “^P” (the [Ctrl/P] key) for Telnet connections to the router.

If autobauding is enabled, the attention character must be set to [Break] because this is the only character that can be detected before the baud rate is established

The **cdcontrol** parameter specifies how the router interprets the state of the DCD input signal. If **cdcontrol** is set to **connect**, when DCD is deasserted, the router terminates existing connections. This is useful when the port is accessed via a dialup modem. If **cdcontrol** is set to **online**, output is not sent to the port unless the DCD input signal is asserted. When the port is configured as a printer port, and the DTR line of the printer is connected to the DCD input of the router, the router determines if a printer is online and powered up. This ensures that print jobs are not sent to a printer that is offline or off. If **cdcontrol** is set to **ignore**, the router ignores the state of the DCD input regardless of the way the port is used. The default is **ignore**.

The **databits** parameter sets the number of data bits per character transmitted by the port. This should match the terminal setting. The default is 8.

The **defaultservice** parameter is used to configure a port to automatically connect to a service whenever a user types anything at a terminal connected to the port, or (in the case of a modem attached to the port) when the modem asserts DCD. This parameter is valid if a service is associated with the port, using the **service** parameter. A port configured for **defaultservice** can not be used to enter commands to the router because the port connects to the default service whenever anything is typed at the router prompt. The **defaultservice** parameter changes the meaning of the **service** parameter and the way the port operates. If **defaultservice** is set to **off** (the default) the port acts as an interactive service port for the service specified by the **service** parameter, and is used to connect the router to the RS-232 port of a host. If **defaultservice** is set to **on**, the **service** parameter specifies the name of the service (interactive or Telnet) to which an automatic connection is to be made. In outputs of the [show asyn command on page 9-53](#) for a **defaultservice** port, the service name is either prefixed by an asterisk or followed by the string “(default)”.

The **dtrcontrol** parameter controls the way the router controls the state of the DTR output signal. If **dtrcontrol** is set to **connect**, the DTR output of the router is asserted for the duration of a valid connection. If **dtrcontrol** is set to **on** or **off**, the DTR line can be driven to the designated state. The default is **on**. This option is intended for ports that are directly connected to host asynchronous ports that require DTR output to be asserted for the duration of a valid connection.

The **echo** parameter sets the echo mode for the port. If **echo** is set to **on**, characters typed following the prompt are echoed to the terminal screen. If **echo** is set to **off**, characters are not echoed to the terminal screen but the router still receives and processes them. This option has effect when the port is not assigned. When the port is assigned, echoing is controlled by the host. The default is **on**.

The **enable** parameter sets the behaviour of the asynchronous port after you have manually disabled the port by using the [disable asyn command on page 9-27](#). If **enable=break** is specified, you can re-enable the port by sending it a break signal. Further break signals will not affect the port's status. The port remains enabled until it is idle for the **timeout** period, or until you manually re-enter the **disable asyn** command. See ["Session Timeout" on page 9-20](#) for more information about this functionality. If **enable=none** is specified, the port's status does not change even if it receives a break signal. The default is **break**.

The **flow** parameter sets the flow control mechanism used for the port in both the transmit and receive directions. If **flow** is set to **none**, the router ignores all incoming flow control characters or lead transitions. The router does not generate flow control characters and the state of the hardware lines do not change. If **flow** is set to **character**, the router uses XON/XOFF flow control. If **flow** is set to **hardware**, the router uses the RTS/CTS lines for flow control. For finer control, the **inflow** and **outflow** parameters can be used to set different flow control mechanisms for the port in the receive and transmit directions, respectively.

The **history** parameter defines the number of commands saved in the command history for future recall with the [show command history command on page 2-16 of Chapter 2, Using the Command Line Interface \(CLI\)](#). The minimum is 0 and the maximum is 99. Setting the history length to zero for a port does not clear all the commands from the history. To clear command history, use the [reset asyn history command on page 9-36](#). The default history length for asynchronous ports and Telnet connections is 30.

The **idletimeout** parameter specifies a period of time, in seconds, for a terminal connection's dedicated TTY device idle timer. If the specified time period lapses since the last time the dedicated TTY device received data from the client, the connection or session is terminated and the terminal screen displays the login prompt. If **0** or **off** are specified, the idle timer remains off, and the session must be explicitly terminated. The default is 0.

If the dedicated TTY device's idle timeout period is modified while there is an established connection, the idle timer for that session is reset so that it uses the new timeout value. Any idle time accumulated by the connection prior to the issuing of the set command is lost.

The **ipaddress** parameter sets the IP address in dotted decimal notation, associated with the port. This parameter may need to be set if the port is used as a network interface using SLIP or PPP. See [Chapter 18, Asynchronous Call Control](#) for more information. The IP address may be cleared by setting **ipaddress** to **none**. The default is **none**.

The **ipxnetwork** parameter specifies the Novell network number assigned to a user accessing a Novell internetwork via the asynchronous port. See [Chapter 18, Asynchronous Call Control](#) for more information. The network number may be cleared by setting **ipxnetwork** to **none** instead of a network number. The default is **none**.

The **login** parameter specifies whether a user can log into an asynchronous port and issue commands on the router. If **on** is specified, users can log into the router; if **off** is specified, they cannot. No command prompt is displayed, no characters are echoed by the port, and input received by the port is ignored. The default is **on**.



**Caution** If **login** is set to **off** from a terminal or terminal emulation session over the asynchronous port, it becomes impossible to enter any other commands into

that session. In this situation, the router can be reconfigured from a Telnet session when there is an interface with a valid IP address and appropriate routes. Alternatively, power cycling the router removes the unsaved configuration.

The **maxoqlen** parameter sets the maximum number of character buffers permitted on the output queue for this port. Once the queue has reached this limit no further buffers are accepted for transmission from the higher layer. The default is **16**. A value of 0 means the length of the output queue is the default value.

The **mtu** parameter sets the Maximum Transmission Unit for the port. This is the maximum number of bytes in a packet transmitted over this port when it is used as a network interface. See [Chapter 18, Asynchronous Call Control](#) for more information. The minimum MTU is 40 and the maximum is 1500. The default is **1500**.

The **name** parameter assigns a name to the port, as a convenient reference to identify ports. For example, it may be set to the name of the person who normally uses the terminal connected to the port, or the location of the terminal. The default name is "Port #" where "#" is the port number. The name appears in the output of the [show asyn command on page 9-53](#).

The **page** parameter sets the number of lines of command output displayed on the terminal screen before the router pauses and waits for the user to press a key to continue. This number may range from 0 to 99. The default is **22** for both asynchronous ports and Telnet connections. If **page** is set to **off**, paging is disabled.

The **parity** parameter sets the parity of each character transmitted by the port. This should match the terminal setting. The default is **none**.

The **prompt** parameter sets the prompt for the port to the default string, such as CMD>, or a user-specified string, or it disables the prompt. It is often convenient to disable the prompt when the port is being used as a manager port or for debugging network problems because it reduces the clutter on the terminal screen. This option has effect when the port is not assigned. When the port is assigned, prompting is controlled by the host.

The **secure** parameter determines whether a user must log in to the port before router commands are accepted. See [Chapter 40, User Authentication](#) for more information on logging in and defining users of the router. The default is **on** for both asynchronous ports and Telnet connections.

The **service** parameter allocates an asynchronous port to be a host port for the named service. This port must be an unallocated terminal port. The service must already have been defined with the [set service command on page 60-23 of Chapter 60, Terminal Server](#) and be of type **interactive**. If **service** is set to **none** the port is deallocated from the service.

The **shellserver** parameter specifies how to handle characters received on the asynchronous port. Use this parameter to prevent output from a device connected to the port being interpreted as commands. If you specify **on**, characters received on the port are sent to the CLI. If you specify **off**, characters received on the port are ignored. You can still use the [connect command on page 60-16 of Chapter 60, Terminal Server](#) to connect to a device attached to the asynchronous port. The default is **on**.

The **speed** parameter sets the speed (baud rate) of the port. This should match the terminal setting. The attention character must be set to [Break] if autobauding is selected. The port expects to see several [Enter] or [Return] characters to determine the terminal speed setting. If another character is entered initially after the port is reset or cleared, the autobauding feature may not select the correct speed. To restart autobauding in this situation, two consecutive [Break] characters should be entered, followed by two [Enter] or [Return] characters. The default is **auto**.

Autobauding does not work with baud rates exceeding 19200 baud, the maximum for many terminals. A port connected to a modem should not be set to autobauding.

Not all speeds are supported on every router model. If an unsupported speed is specified, an error message is displayed and the command is ignored.

The **stopbits** parameter sets the number of stop bits per character transmitted by the port. This should match the terminal setting. The default is 1.

The **tentimervalue** parameter sets the length of the ten timer, in milliseconds. Reducing the length of the ten timer increases the port's responsiveness (see [“Making Asynchronous Ports Respond More Quickly” on page 9-23](#)). Unless you are logged in via the port you want to change, also specify the asynchronous port number. The default **tentimervalue** is 100.

The **timeout** parameter specifies a length of time in seconds for which the asynchronous port can remain idle before it is disabled and the user is logged out. This parameter only takes effect on a port if you have already manually disabled the port using the [disable asyn command on page 9-27](#). To re-enable the port, send it a break signal. See [“Session Timeout” on page 9-20](#) for more information. The **timeout** parameter is only valid if the **enable** parameter is set to **break**. The default timeout is 60 seconds.

The **type** parameter specifies the type of terminal attached to the port. If **type** is set to **vt100**, the router expects the terminal to support standard VT100 escape sequences and uses them. If **type** is set to **dumb**, the router does not use VT100 escape sequences. The **dumb** option is usually required for ports connected to printers or very old terminals that do not support VT100 escape sequences. The default is **vt100** for both asynchronous ports and Telnet connections.

**Examples** The following command configures asynchronous port 17:

```
set asy=17 da=7 par=odd sp=9600 st=1
```

Each parameter can also be set separately:

```
set asy=17 da=7
set asy=17 par=odd
set asy=17 sp=9600
set asy=17 st=1
```

**Related Commands**

- [disable asyn](#)
- [enable asyn](#)
- [reset asyn](#)
- [set tty](#)
- [show asyn](#)
- [show command history](#)
- [show service](#)
- [show tty](#)

---

## set eth linkup

---

**Syntax** SET ETH=*n* LINKup={30..1000000000|INFINITY}

where

- *n* is the number of the instance on an Ethernet PIC

**Description** This command sets the length of time for which the specified Ethernet interface stays up even if it receives no packets, including no keepalive packets. This command only applies to interfaces on an Ethernet PIC.

The **eth** parameter specifies the instance number of the interface on the PIC.

The **linkup** parameter specifies the time for which the Ethernet interface remains up, in seconds. If you specify **infinity**, the interface always remains up.

**Example** To set the eth1 interface so that the link only goes down if no packets are received for 30 days, use the command:

```
set eth=1 link=2592000
```

**Related Commands** [show eth state](#)

---

## set eth maxbandwidth

---

**Syntax** SET ETH=*n* MAXbandwidth=0..100000

where *n* is the number of the Ethernet interface

**Description** This command imposes a maximum bandwidth limit on an Ethernet interface. This command is valid for some interfaces; when it is not valid, an error message is displayed.

The **maxbandwidth** parameter is measured in 1000s of bits/s. If the bandwidth specified is 0, this feature is turned off and there is no limit imposed. If  $0 < \text{maxbandwidth} \leq 128$  is specified, a limit of 128000bits/s is used.

**Examples** To set a maximum bandwidth limit of 10Mbit/s on the ETH0 interface, use the command:

```
set eth=0 max=10000
```

**Related Commands** [show eth state](#)

## set eth speed

---

**Syntax** SET ETH=*n* SPeEd={AUtonegotiate|10MHalf|10MFull|100MHalf|100MFull}

For the AR770S:

```
SET ETH=n SPeEd={AUtonegotiate|10MHalf|10MFull|100MHalf|100MFull|1000MFull}
```

where *n* is the number of the Ethernet interface

**Description** This command sets the link speed for an Ethernet interface. The interface number must be specified.

For the AR725, AR745, AR750S, and AR750S-DP: The **speed** parameter specifies either the fixed link speed and duplex mode for the interface to use, or autonegotiation of link speed and duplex mode. For 10 Mbps interfaces, the only value is **10mhalf**. For 100 Mbps interfaces, all values are allowed and the default is **autonegotiate**. If **autonegotiate** is specified, the interface negotiates with its Ethernet link partner to choose the fastest speed that both ends of the link can use.

For the AR770S, the **speed** parameter specifies the link speed for the interface to use. For 10 Mbps interfaces, the only value is **10mhalf**. For 100 Mbps interfaces, all values except **1000mfull** are allowed. For 1000 Mbps interfaces, all values are allowed. If **autonegotiate** is specified, the interface negotiates with its Ethernet link partner to choose the fastest speed that both ends of the link can use. The default for 10 Mbps interfaces is **10mhalf**; the default for 100 Mbps and 1000 Mbps interfaces is **autonegotiate**.



**Caution** Setting the Ethernet link speed to a value other than the default for the interface may prevent the interface from working properly and may interfere with the operation of other devices on the network. If the interface link speed is set to a value that the link partner does not support, the interface is unable to send or receive data. If the interface link speed is set to a value different from that of the link partner, the **show eth state** command may indicate that the link is up even when it is not.

If the interface link speed is set to **100mhalf** or **100mfull** and the interface is connected to a 10 Mbps network, the interface emits 100 Mbps signals onto the network. This could make the network unusable.

For the AR725, AR745, AR750S, and AR750S-DP: If the link speed is changed from **10mhalf** or **10mfull** to **100mhalf** or **100mfull**, and the link partner can autonegotiate link speed, the link partner may not detect the change to 100Mbps and the interface is unable to send or receive data. To prevent this, change the link speed from **10mhalf** or **10mfull** to **autonegotiate**, wait a few seconds for auto-negotiation to occur, then change the link speed to **100mhalf** or **100mfull**.

For the AR770S: If the link speed is changed from **10mhalf** or **10mfull** to **100mhalf**, **100mfull** or **1000mfull**, and the link partner can autonegotiate link speed, the link partner may not detect the change to 100Mbps or 1000Mbps, making the interface unable to send or receive data. To prevent this, change the link speed from **10mhalf** or **10mfull** to **autonegotiate**, wait a few seconds for

autonegotiation to occur, then change the link speed to **100mhalf**, **100mfull** or **1000mfull**.

**Example** To force a 100 Mbps Ethernet interface 0 to operate at 10 Mbps half-duplex regardless of the capabilities of the link partner, use the command:

```
set eth=0 sp=10mh
```

**Related Commands** [show eth state](#)

## set interface mtu

**Syntax** SET INterface={*ifIndex*|*interface*} MTU=*value*

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface
- *value* is an integer Valid values are module-dependent - see below.

**Description** This command sets the MTU value for the given interface. The **mtu** parameter specifies the value for the maximum transmission unit.

If the interface being set is a PPP interface, and the **mtu** specified in the command is higher than the current MTU on the interface, the PPP connection is reset. This is done so that the higher value may be negotiated through the PPP protocol. If the negotiated rate is lower than the requested MTU, the MTU value is dropped to the negotiated rate. Valid interfaces are:

- eth (such as eth0, eth0-1)
- FR (such as fr0)
- PPP (such as ppp0, ppp1-1)

To see a list of current interfaces, use the [show interface command on page 9-73](#).

When the MTU is set higher than it is currently, the PPP interface is reset and a short outage on that link may result. The following table lists allowable MTU values.

Interface	Minimum MTU	Maximum MTU	Default MTU
FR	256	1600	1600
PPP (not over Eth)	256	1500	1500
PPP over Eth	256	1492 <sup>a</sup>	1492
ETH	256	1500	1500
The following MTU sizes apply to interfaces that are running Jumbo frames			
	256	9198	1500

a. The maximum setting is 8 bytes smaller than the Ethernet interface

If the interface given by the user is a Frame Relay interface, all logical interfaces in the interface are affected. If a specific interface is given, only that logical interface is affected.

**Examples** To set the MTU value for a PPP3 interface to be 1400, use the command:

```
set int=ppp3 mtu=1400
```

**Related Commands** [show interface](#)



---

## set interface traplimit

---

**Syntax** SET INterface={*ifIndex*|*interface*|DYNamic} TRaplimit=1..60

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface

**Description** This command sets the maximum number of link up/down traps generated in one minute for the specified interface. The default is 20 trap messages per minute. Valid interfaces are:

- eth (such as eth0, eth0-1)
- FR (such as fr0)
- PPP (such as ppp0, ppp1-1)
- VLAN (such as vlan1, vlan1-1)

To see a list of current interfaces, use the **show interface** command.

IP and SNMP must be enabled and correctly configured to generate traps. See [Chapter 21, Internet Protocol \(IP\)](#) for a detailed description of the commands required to enable and configure IP. See [Chapter 54, Simple Network Management Protocol \(SNMP\)](#) for a detailed description of the commands required to enable and configure SNMP.

**Examples** To set the trap limit for interface ppp2 to 40, use the command:

```
set int=ppp2 tr=40
```

**Related Commands** [disable interface linktrap](#)  
[enable interface linktrap](#)  
[show interface](#)

## set syn

**Syntax** SET SYN=*n* [C={ON|OFF|I}] [CD={ON|OFF|DTR}] [CTS={ON|OFF|RTS}] [DATAsense={NORMAl|INVerted}] [DSR={ON|OFF|DTR}] [DTR={ON|OFF|DSR}] [I={ON|OFF|C}] [MAXoqlen=*max-queue*] [MINTxint=*min-interval*] [RTS={ON|OFF|CTS}] [SPeed=*speed*] [TXClock={FALling|RISing}]

where:

- *n* is the number of the synchronous interface.
- *speed* is the baud rate for the interface.
- *max-queue* is the maximum permitted output queue length for the interface.
- *min-interval* is the minimum number of microseconds between transmitted frames.

**Description** This command sets the configuration parameters for a synchronous interface. The interface number must be specified.

The **c** parameter specifies the mode of operation for the X.21 modem control signal output called "C". This parameter is valid when an X.21 DTE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **c** is **on**, the output is always on when the interface is active. When **c** is **off**, the output is always off when the interface is active. When **c** is **i**, then when the interface is active, **c** is on when the modem control input signal **i** is on and **c** is off when **i** is off. An interface is active when there is a layer 2 module attached and the interface is enabled. If the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default value for **c** is **on**.

The **cd** parameter specifies the mode of operation of the RS-232/V.35 modem control signal output called "CD". This parameter is valid when an RS-232 DCE or V.35 DCE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **cd** is **on**, the output is always on when the interface is active. When **cd** is **off**, the output is always off when the interface is active. When **cd** is **dtr**, then when the interface is active, **cd** is on when the modem control input signal **dtr** is on, and off when **dtr** is off. An interface is active when there is a layer 2 module attached and the interface is enabled. If the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The **cts** parameter specifies the mode of operation for the RS-232/V.35 modem control signal output called "CTS". This parameter is valid when an RS-232 DCE or V.35 DCE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **cts** is **on**, the output is always on when the interface is active. When **cts** is **off**, the output is always off when the interface is active. When **cts** is **rts**, then when the interface is active, **cts** is on when the modem control input signal RTS is on, and off when RTS is off. An interface is active when there is a layer 2 module attached and the interface is enabled. If the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The **datasense** parameter specifies the sense of the transmitted data on the synchronous interface. The default is **normal**. The **inverted** option may be used when the synchronous link is a T1 link and the router is responsible for limiting the number of consecutive zeroes transmitted, for example when raw AMI

encoding rather than B8ZS is used. Both ends of the T1 link must be set to **inverted** as the receiver also expects to receive inverted data. Note that 68302 and 68562 hardware types do not support data inversion. The hardware type can be read from the output of the [show syn command on page 9-79](#).

The **dsr** parameter specifies the mode of operation for the RS-232/V.35 modem control signal output called “DSR”. This parameter is valid when an RS-232 DCE or V.35 DCE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **dsr** is **on**, the output is always on when the interface is active. When **dsr** is **off**, the output is always off when the interface is active. When **dsr** is **dtr**, then when the interface is active, DSR is on when the modem control input signal DTR is on, and off when DTR is off. An interface is active when there is a layer 2 module attached and the interface is enabled. When the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The **dtr** parameter specifies the mode of operation for the RS-232/V.35 modem control signal output called “DTR”. This parameter is valid when an RS-232 DTE or V.35 DTE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **dtr** is **on**, the output is always on when the interface is active. When **dtr** is **off**, the output is always off when the interface is active. When **dtr** is **dsr**, then when the interface is active, DTR is on when the modem control input signal DSR is on, and off when DSR is off. An interface is active when there is a layer 2 module attached and the interface is enabled. When the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The **i** parameter specifies the mode of operation for the X.21 modem control signal output called “I”. This parameter is valid when an X.21 DCE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **i** is **on**, the output is always on when the interface is active. When **i** is **off**, the output is always off when the interface is active. When **i** is **c**, then when the interface is active, I is on when the modem control input signal C is on, and off when C is off. An interface is active when there is a layer 2 module attached and the interface is enabled. When the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The value of **maxoqlen** is the maximum number of frames that may be queued for transmission over the synchronous interface at any one time. Each frame passed to the SYN module by the layer 2 module is inserted in the output queue according to its priority and the queue length incremented. When the output queue length is then greater than the **maxoqlen** value, then the oldest, lowest priority frame is discarded. The **maxoqlen** parameter is optional. The default is **100**. A value of 0 means there is no limit on the length of the output queue. We recommend setting **maxoqlen** to at least 5. The most common layer 2 protocols (e.g. PPP and Frame Relay) maintain their own packet transmission queues. A small number of packets are queued by the synchronous interface to ensure maximum possible bandwidth utilisation. If the synchronous interface maximum queue length is set too low, packets may be discarded by the synchronous interface to enforce the maximum queue length. This may disrupt correct operation of the layer 2 protocol and any Quality of Service mechanisms that have been configured. The synchronous interface queue is priority aware, so reducing the maximum length will not provide a latency improvement.

The value of **mintxint** sets the minimum delay (in microseconds) between frames transmitted over an interface. This is useful where the router is communicating with a device that is unable to accept back-to-back frames. The

maximum permitted value is 1000000 (1 second). The **mintxint** parameter is optional. The default value is zero, which means no delay is inserted between transmitted frames.

The **rts** parameter specifies the mode of operation for the RS-232/V.35 modem control signal output called "RTS". This parameter is valid when an RS-232 DTE or V.35 DTE transition cable is connected to the synchronous interface, and is ignored if any other transition cable is used. When **rts** is **on**, the output is always on when the interface is active. When **rts** is **off**, the output is always off when the interface is active. When **rts** is **cts**, then when the interface is active, RTS is on when the modem control input signal CTS is on, and off when CTS is off. An interface is active when there is a layer 2 module attached and the interface is enabled. When the layer 2 module takes control of this output, then the setting of this parameter is ignored. The default is **on**.

The **speed** parameter specifies the baud rate for the interface. The baud rate should be specified, even if the interface is being clocked by an external modem. This enters the speed into the interface MIB so that any module that requires this information (e.g. a routing module) can use it (for example, to calculate metrics for the interface). The baud rate may be set to any value between 0 and 10000000. If the interface is configured to generate clocks then the maximum permitted speed is 64000 for 68302-based synchronous interfaces and 38400 for 68562-based synchronous interfaces (see the *Hardware Reference* for more details). The nearest possible value to the specified baud rate is used. The output from the **show syn** command gives the actual baud rate. If the interface is configured to receive clocks and the speed is set to a value greater than 64000 (or 38400), and then the transition cable is changed to generate clocks, the actual baud rate is reduced to 64000 (or 38400). When an RS-232 DCE transition cable is connected to a synchronous interface configured to generate clocks, the maximum clock speed is 38400 bps. When an X.21 DCE or V.35 DCE transition cable is connected to a synchronous interface configured to generate clocks, the maximum clock speed is 2 Mbps. This can be displayed with the **show syn** command, but the configured value for this interface is **not** changed. The default value for **speed** is 48000.

The **txclock** parameter specifies the edge of the transmit clock used to clock the transmit data out of the interface. The default value is **rising** and is appropriate for the majority of installations. At high baud rates it may be helpful to select **falling** if a high level of errors are occurring on the synchronous communications link. This setting may also be helpful at moderate baud rates if the cable from the synchronous interface to the NTU is long. Note that 68302 and 68562 hardware types support rising edge clocking only. The hardware type can be read from the output of the **show syn** command.

**Examples** To set the baud rate for synchronous interface 2 to 38400, use the command:

```
set syn=2 sp=38400
```

To set the DTR modem control output signal to always be off and the RTS modem control output signal to follow the state of the CTS modem control input signal, on a synchronous interface with a V.35 or RS-232 DTE transition cable, use the command:

```
set syn=0 dtr=off rts=cts
```

**Related Commands** [show syn](#)

---

## show asyn

---

**Syntax**    `SHoW ASYn[=port-number|ALL] [{COUnters[={Diagnostic|INTERface|Rs232}}]|History|Summary}]`

where *port-number* is the number of the port. Ports are numbered sequentially starting with 0.

**Description**    This command displays configuration information for one or more asynchronous ports. If a port number is specified, then the information for that port is displayed. If a port number is not specified, information for the port from which the command was issued is displayed. If **all** is specified, then information for all the ports on the router is displayed. If the command is issued from a port with User privilege, the port number may not be specified and the information displayed is for the port from where the command was issued.

If no parameters are specified, then full configuration information for the specified ports is displayed ([Figure 9-1 on page 9-54](#), [Table 9-1 on page 9-55](#)).

The **counter** parameter displays counters from the specified categories ([Figure 9-2 on page 9-58](#), [Table 9-2 on page 9-58](#)). If a category is not specified then counters from all categories are displayed. If **diagnostic** is specified then counters from the asynchronous interface table of the router enterprise MIB are displayed. If **interface** is specified, then interface counters from the interfaces MIB are displayed. Interface MIB counters exist for ports that are in use as network interfaces. If **rs-232** is specified then counters from the asynchronous port table of the RS-232 like hardware devices MIB are displayed. The **counter** parameter may also not be specified from a port with User privilege.

The **history** parameter displays the command history for the specified ports ([Figure 9-3 on page 9-59](#)). The command history can also be displayed with the [show command history command on page 2-16 of Chapter 2, Using the Command Line Interface \(CLI\)](#). After displaying the command history the router prompts for a command number from the list. The user can enter a number and press the Enter or Return key to select a command, or just press Enter or Return to return to the prompt. If a valid command number is entered, then the command is displayed at the prompt ready for editing and execution.

The **summary** parameter displays a one-line summary for the specified port or ports ([Figure 9-4 on page 9-59](#), [Table 9-3 on page 9-59](#)).

Figure 9-1: Example output from the **show asyn** command

```

ASYN 0 : 0000005625 seconds  Last change at: 0000005606 seconds

ASYN information
Name ..... Asyn 0
Status ..... enabled
Mode ..... PPP
PPP Index ..... 1
TX ACCM ..... 00000000
Data rate ..... 38400
Parity ..... none
Data bits ..... 8
Stop bits ..... 1
Test mode ..... no
In flow state (mode) ..... on (Hardware)
Out flow state (mode) ..... off (Hardware)
Autobaud mode ..... disabled
Max tx queue length ..... 100
TX queue length ..... 0
Transmit frame ..... none
RX queue length ..... 0
IP address ..... none
Max transmission unit ..... 1500
Ten timer value ..... 100
IPX Network ..... none
Enable Mode.....break
Enabled Status Time Left....59

Control signals
  DTR (out) ..... on on      1
  RTS (out) ..... on -       1
  CD (in)  ..... off connect 0
  CTS (in) ..... off -       0
  RNG (in) ..... off -       0

TTY information
Instance ..... 18
Login Name .....
Description ..... Asyn 2
Secure ..... yes
Connections to .....
Current connection ..... none
In flow state ..... on
Out flow state ..... on
Attached module ..... ASYN Call Control
Attached module instance .. 2
Type ..... VT100
Service ..... none
Prompt ..... login
Echo ..... yes
Attention ..... break
Manager ..... no
Edit mode ..... insert
History length ..... 20
Page size ..... 22
Idle Timeout (seconds)..... 300

```

Table 9-1: Parameters in output of the **show asyn** command

Parameter	Meaning
Name	The name of the asynchronous port.
Status	Whether the port is enabled or disabled.
Mode	The mode of operation for the port. This is "Ten" for terminal server ports (characters bundled every tenth of a second). For network interfaces, it is either SLIP, SLIP6, CSLIP, CSLIP6, or SLIPAd.
PPP Index	The index for the current PPP session. This field is displayed when the port is being used by ACC for a PPP session.
TX ACCM	The current ACCM used by PPP for transmitted control characters. This field is displayed when the port is being used by ACC for a PPP session.
Data rate	The baud rate for the port. The default is autobaud.
Parity	The parity setting for the port.
Data bits	The number of data bits in each transmitted character and the number expected in each received character.
Stop bits	The number of stop bits transmitted after each character and the number expected after each received character.
Test mode	Whether the interface is in a test mode.
In flow state (mode)	The flow control state and mode for the incoming data path. The flow control state may be "on" or "off", indicating whether the port is able to receive characters. The mode may be "none" (no flow control), "hardware" (RTS/CTS flow control), or "XON/XOFF" (XON/XOFF flow control).
Out flow state (mode)	The flow control state and mode for the outgoing data path. See "In flow state" for a description. The mode is the same for both directions.
Autobaud mode	Whether autobauding is enabled or disabled. When enabled, whether the autobauding process is searching (the port is trying to determine the baud rate of the terminal) or found (the baud rate has been set).
Max tx queue length	The maximum number of character buffers permitted on the transmit queue for the port. This parameter affects a port used as a network interface.
Tx queue length	The length of the queue of character buffers that are waiting to be transmitted to the port.
Transmit frame	The address of the current frame being transmitted by the port, or "none" if no frame is currently being transmitted.
Rx queue length	The length of the queue of character buffers that are waiting to be passed up from the port to higher layers.
IP address	The IP address set for the port, or "none" if no IP address has been set.
Max transmission unit	The maximum number of bytes transmitted in one packet over the interface.

Table 9-1: Parameters in output of the **show asyn** command (Continued)

Parameter	Meaning
Ten timer value	The length of the <i>ten timer</i> , in milliseconds. When an asynchronous port is in <i>ten mode</i> , it bundles together the characters that it receives within a certain time period, instead of passing them one at a time to a higher protocol layer for processing. The ten timer sets the time period over which characters are bundled.
IPX Network	The IPX network number set for the port, or "none" if no IPX network has been set.
Enable Mode	The behaviour of the switch when it receives a break signal. If "break" is displayed, a disabled asynchronous port is enabled when a break signal is received. If "none" is specified, the port's status does not change even if it receives a break signal.
Enable Status Time Left	The remaining length of time in seconds for which the asynchronous port can remain inactive before its status is set to disabled. If a timeout time was not specified in the <b>set asyn</b> command, this value is 0.
Control signals	The control signals present on the interface, their direction (output or input to the router), their state and the number of transitions they have made since the router was powered up or the counters reset. For the DTR and CD signal lines their mode of operation is also displayed.
Instance	The instance number for the TTY device dedicated to this port.
Login name	The login name of the user logged in to this port, if any.
Description	The name assigned to the port.
Secure	Whether the port is secure.
Connections to	A list of TTY devices (if any) to which this port TTY is linked for the purpose of providing multiple sessions.
Current connection	The instance number of the TTY that this port TTY is currently connected to, or "none" if there is no active connection.
In flow state	The input flow control state for the TTY dedicated to this port.
Out flow state	The output flow control state for the TTY dedicated to this port.
Attached module	The module that owns the port. By default this is terminal server.
Attached module instance	The instance of the module that owns the port.
Type	Whether the terminal type setting for the port is dumb or VT100.
Service	The name of the service to which this port belongs, if any.



Table 9-1: Parameters in output of the **show asyn** command (Continued)

Parameter	Meaning
Prompt	Type of prompt given on this port: default off login password confirm encapsulation a user-defined string
Echo	Whether the port echoes input characters.
Attention	The attention character for this port; either none, break, or char. For an asynchronous port the default attention character is "break".
Manager	Whether the port has Manager privilege.
Edit mode	The edit mode for the port; either "?", "insert", or "overstrike". The default is "insert".
History length	The maximum number of commands that are held in the command history for this port. The default is 30.
Page mode/length	The number of lines of command output the router displays before pausing and waiting for the user to press a key, or "off" when page mode is disabled for this port. The default is 22.
Idle Timeout	Maximum period of time in seconds without data being received from a given client before the corresponding session is terminated.

Figure 9-2: Example output from the **show asyn counter** command

Asyn 1: 0000014132 seconds      Last change at: 0000000000 seconds			
RS-232 MIB Counters			
Receive:			
ParityErrs	0		
FramingErrs	0		
OverrunErrs	0		
Diagnostic Counters			
Receive:		Transmit:	
inCharacters	690025	outCharacters	689828
inBuffers	13513	outBuffers	13526
fcsErrors	0	droppedBuffers	0
pppErrors	0		
slipErrors	0		
General:			
disconnects	0		
Interface MIB Counters			
Receive:		Transmit:	
ifInOctets	690025	ifOutOctets	689828
ifInUcastPkts	13513	ifOutUcastPkts	13526
ifInNUcastPkts	0	ifOutNUcastPkts	0
ifInDiscards	0	ifOutDiscards	0
ifInErrors	0	ifOutErrors	0
ifInUnknownProtos	0	ifOutQLen	0

Table 9-2: Parameters in output of the **show asyn counter** command

Parameter	Meaning
ParityErrs	Number of characters received with a parity error.
FramingErrs	Number of characters received with a framing error.
OverrunErrs	Number of characters lost due to an overrun error.
inCharacters	Total number of characters received.
inBuffers	Number of character buffers transferred to a higher layer.
fcsErrors	Number frames received with a frame check sequence error.
pppErrors	Number of PPP frames received with errors.
slipErrors	Number of SLIP frames received with errors.
outCharacters	Total number of characters transmitted.
outBuffers	Number of character buffers transmitted for a higher layer.
droppedBuffers	Number of character buffers discarded because the output queue had reached its maximum allowed length.
disconnects	The number times a SLIP or PPP session has been terminated by the modem disconnecting (dropping CD).
ifInOctets	Number of octets received on this interface.
ifInUcastPkts	Number of unicast packets delivered to a higher-layer protocol.
ifInNUcastPkts	Number of non-unicast packets delivered to a higher-layer protocol.
ifInDiscards	Number of inbound packets discarded though no errors had been detected to preventing them from being deliverable to higher-layer protocol.

Table 9-2: Parameters in output of the **show asyn counter** command (Continued)

Parameter	Meaning
ifInErrors	Number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.
ifInUnknownProtos	Number of packets discarded because they were for an unconfigured protocol.
ifOutOctets	Number of octets transmitted, including framing.
ifOutUcastPkts	Number of unicast packets transmitted or discarded.
ifOutNUcastPkts	Number of non-unicast packets transmitted or discarded.
ifOutDiscards	Number of packets discarded though no errors had been detected preventing their being transmitted.
ifOutErrors	Number of packets not transmitted because of errors.
ifOutQLen	Length of the output packet queue.

Figure 9-3: Example output from the **show asyn history** command

```

1  sh asyn cou
2  sh asyn sum
3  sh asyn hist
4  sh asyn cou
5  login manager
6  sh tty
7  sh asyn=1 cou
8  sh syn cou=int
9  sh asyn=1 cou

Enter command number>

```

Figure 9-4: Example output from the **show asyn summary** command.

```

Port Name           Module Mode   Data Format Attn Secur Mgr Service
-----
001  Asyn 1         ACC    CSLIP  19200,N,8,1 brk  yes  no  -
-----

```

Table 9-3: Parameters in output of the **show asyn summary** command

Parameter	Meaning
Port	Number of the asynchronous port.
Name	Name assigned to the port.
Module	Module that owns the port.
Mode	Mode of operation for the port.
Data Format	Baud rate, parity, number of data bits and number of stop bits configured for the port.
Attn	Attention character for the port; either "-", "brk", or "chr".
Secur	Whether the port is secure.
Mgr	Whether the port has Manager privilege.
Service	The name of the service to which the port is allocated, if any.

**Examples** To show the configuration for asynchronous port 1, use the command:

```
sh asy=1
```

To show all the counters for asynchronous port 1 enter:

```
sh asy=1 cou
```

To see the command history for the asynchronous port to which the terminal is connected enter:

```
sh asy h
```

To obtain an abbreviated display for asynchronous port 1 enter:

```
sh asy=1 s
```

**Related Commands**

- [disable asyn](#)
- [enable asyn](#)
- [reset asyn](#)
- [reset asyn counter](#)
- [reset asyn history](#)
- [set manager asyn](#)
- [set asyn](#)
- [set service](#)
- [set tty](#)
- [show tty](#)

## show eth configuration

**Syntax** SHow ETH=*n* CONfiguration

where *n* is the number of the Ethernet interface

**Description** This command lists the modules in the router that are configured to use an Ethernet interface, and the encapsulations used on an interface (Figure 9-5, Table 9-4).

Figure 9-5: Example output from the **show eth configuration** command

Configuration for ETH instance 0:				
Module	Protocol	Format	Discrim	MAC address
IPG	IP	Ethernet	0800	0000cd000027
IPG	ARP	Ethernet	0806	0000cd000027
IPX	Novell	Novell	-	0000cd000027
DNT		DECnet	Ethernet	6003
aa0004003908				
Bridge		Spanning tree	802.2	42
0000cd000e23				
Bridge		Novell802.3	Novell	-
Bridge		ETHERTALK 2 AARP SNAP		00000080f3 -

Table 9-4: Parameters in output of the **show eth configuration** command

Field	Meaning
Module	Name of the software module that has configured to the Ethernet drivers to receive packets with this encapsulation.
Protocol	Name of the protocol, which is determined from the format and discriminator.
Format	Encapsulation format specified by the module.
Discrim	Discriminator specified by the module to identify packets that should receive the specified format.
MAC Address	Media Access Control source address for which the module wishes to receive packets. This is commonly known as the Ethernet address.

When routing DECnet, a router must use a MAC address determined from the DECnet address assigned to the router, rather than the globally unique address assigned by the router manufacturer. With routers from many manufacturers, when DECnet routing is enabled the router must be rebooted so that the new MAC address can take effect. This can be disruptive to network operation. To avoid this, the router is able to use more than one MAC address simultaneously.

**Related Commands** [show eth counters](#)  
[show eth receive](#)

## show eth counters

**Syntax** `SHoW ETH[=n] COUnTERS[={COLlision|DIAGnostic|DOT3stat|INTERface}]`

where *n* is the number of the Ethernet interface

**Description** This command displays the MIB counters for an Ethernet interface. If the interface is not specified, the counters for all Ethernet interfaces are displayed. If a category is not specified, all counters are displayed.

If **collision** is specified, collision statistics counters from the dot3 MIB are displayed (Figure 9-6 on page 9-62). Collision frequencies are displayed in pairs of columns, representing a histogram. The right hand column (the value axis of the histogram) is a count of individual MAC frames for which the transmission on the specified interface was accompanied by the number of per-frame media collisions specified in the left hand column (the category axis of the histogram).

If **diagnostic** is specified, diagnostic counters specific to the router's Ethernet hardware are displayed. The output is divided into two parts, *device independent* counters that are the same for all router models, and *device dependent* counters that differ depending on whether the interface uses 68360 hardware (Figure 9-7 on page 9-63, Table 9-5 on page 9-64), SONIC hardware (Figure 9-8 on page 9-63, Table 9-5 on page 9-64), M860F hardware (Figure 9-9 on page 9-64, Table 9-5 on page 9-64), or 79C972 hardware (Figure 9-10 on page 9-64, Table 9-5 on page 9-64).

If **dot3stat** is specified, statistics counters from the dot3 MIB are displayed (Figure 9-11 on page 9-67, Table 9-6 on page 9-67).

If **interface** is specified, interface counters from the MIB-II MIB are displayed (Figure 9-12 on page 9-68, Table 9-7 on page 9-68).

Figure 9-6: Example output from the **show eth counters=collision** command

ETH instance 0:		1245 seconds		Last change at:		0 seconds	
dot3 MIB Collision Statistics Counters							
Collision frequencies:							
1:	11	5:	0	9:	0	13:	0
2:	9	6:	0	10:	0	14:	0
3:	3	7:	0	11:	0	15:	0
4:	0	8:	0	12:	0	16:	0

Figure 9-7: Example output from the **show eth counters=diagnostic** command for 68360-based hardware

ETH instance 0:	438 seconds	Last change at:	0 seconds
Device Independent Diagnostic Counters (68360 hardware)			
EthProtoCacheHit	463	EthProtoCacheMiss	58
DSAPProtoCacheHit	0	DSAPProtoCacheMiss	0
SNAPProtoCacheHit	0	SNAPProtoCacheMiss	0
RxFIFOOverrun	0	TxFIFOUnderrun	0
RxTooFewBuffers	0	TxTooManyFragments	0
BusError	0	TxDescriptorAreaFull	0
Reset	0	TxFrameTooLong	0
LoadCAMFailure	0	TxLostInterrupt	0
Device Dependent Diagnostic Counters (68360 hardware)			
CommandTimeout	0	TxNoPacket	0
Command	0		

Figure 9-8: Example output from the **show eth counters=diagnostic** command for SONIC-based hardware

ETH instance 0:	131 seconds	Last change at:	0 seconds
Device Independent Diagnostic Counters (Sonic hardware)			
EthProtoCacheHit	112	EthProtoCacheMiss	9
DSAPProtoCacheHit	0	DSAPProtoCacheMiss	0
SNAPProtoCacheHit	0	SNAPProtoCacheMiss	0
RxFIFOOverrun	0	TxFIFOUnderrun	0
RxTooFewBuffers	0	TxTooManyFragments	0
BusError	0	TxDescriptorAreaFull	0
Reset	0	TxFrameTooLong	0
LoadCAMFailure	0	TxLostInterrupt	0
Device Dependent Diagnostic Counters (SONIC hardware)			
RxStatusError	0	TxEOLOverrun	0
RxBuffMismatch	0	TxDescCorrupt	0
RxBuffAreaExceeded	0	TxSpuriousBCMError	0
RxBuffExhausted	0	TxBCMError	0
RxDescExhausted	0	TxPacketMonitoredBad	0

Figure 9-9: Example output from the **show eth counters=diagnostic** command for M860F-based hardware

ETH instance 0:	21 seconds	Last change at:	2 seconds
Device Independent Diagnostic Counters			
EthProtoCacheHit	203	EthProtoCacheMiss	26
DSAPProtoCacheHit	0	DSAPProtoCacheMiss	0
SNAPProtoCacheHit	0	SNAPProtoCacheMiss	0
RxFIFOOverrun	0	TxFIFOUnderrun	0
RxTooFewBuffers	0	TxTooManyFragments	0
BusError	0	TxDescriptorAreaFull	0
Reset	0	TxFrameTooLong	0
LoadCAMFailure	0	TxLostInterrupt	0
Device Dependent Diagnostic Counters (M860F hardware)			
LinkChanges	1	MIITimeouts	0
AutoNegCompletes	1	ParallelDetFaults	0

Figure 9-10: Example output from the **show eth counters=diagnostic** command for 79C972-based hardware

ETH instance 1:	35 seconds	Last change at:	1 seconds
Device Independent Diagnostic Counters			
EthProtoCacheHit	212	EthProtoCacheMiss	40
DSAPProtoCacheHit	0	DSAPProtoCacheMiss	0
SNAPProtoCacheHit	0	SNAPProtoCacheMiss	0
RxFIFOOverrun	0	TxFIFOUnderrun	0
RxTooFewBuffers	0	TxTooManyFragments	0
BusError	0	TxDescriptorAreaFull	0
Reset	0	TxFrameTooLong	0
LoadCAMFailure	0	TxLostInterrupt	0
Device Dependent Diagnostic Counters (79C972 hardware)			
LinkChanges	1	ReceiveCollisions	0
AutoNegCompletes	0	ParallelDetFaults	0
MIIPhyDetectTransitions	0	MIIManagementReadErrors	0

Table 9-5: Parameters in output of the **show eth counters=diagnostic** command

Counter	Meaning
EthProtoCacheHit	The number of times for an Ethernet protocol packet the Ethernet type field matched that of a protocol discriminator structure in the cache.
DSAPProtoCacheHit	The number of times for a DSAP protocol packet the DSAP field matched that of a protocol discriminator structure in the cache.
SNAPProtoCacheHit	The number of times for a SNAP protocol packet the SNAP field matched that of a protocol discriminator structure in the cache.
RxFIFOOverrun	The number of times reception of a packet failed due to a FIFO overrun.



Table 9-5: Parameters in output of the **show eth counters=diagnostic** command

Counter	Meaning
RxTooFewBuffers	The number of times that after the reception of a packet or during recovery from a receive buffers exhausted interrupt there were insufficient free buffers to replenish the queue of buffers available for reception.
BusError	The number of times a direct memory access transfer was aborted due to a bus error.
Reset	The number of times the ETHRecover routine was called in response to a serious error.
LoadCAMFailure	The number of times a load of the CAM failed.
EthProtoCacheMiss	The number of times for an Ethernet protocol packet the Ethernet type field matched that of a protocol discriminator structure in the discriminator list but the structure was not in the cache.
DSAPProtoCacheMiss	The number of times for a DSAP protocol packet the DSAP field matched that of a protocol discriminator structure in the discriminator list but the structure was not in the cache.
SNAPProtoCacheMiss	The number of times for a SNAP protocol packet the SNAP field matched that of a protocol discriminator structure in the discriminator list but the structure was not in the cache.
TxFIFOUnderrun	The number of times the transmission of a packet failed due to a FIFO underrun.
TxTooManyFragments	The number of times a packet could not be transmitted because it contained too many fragments.
TxDescriptorAreaFull	The number of times there was insufficient room in the Transmit Descriptor Area because there were so many packets queued for transmission and not yet transmitted.
TxFrameTooLong	The number of times a frame was not transmitted because it exceeded the maximum length of an Ethernet frame.
TxLostInterrupt	The number of times the lost transmit interrupt timer timed out before a packet had been transmitted.
CommandTimeout	[68360 hardware] The number of times a command to the Ethernet hardware did not complete before the timeout timer expired.
Command	[68360 hardware] The code of the command that was to be issued when a command timeout was detected.
TxNoPacket	[68360 hardware] The number of times the Ethernet hardware reported a transmit error, but there was no packet being transmitted or the errored packet could not be identified.
RxStatusError	[SONIC hardware] The number of times the packet length or status fields of a descriptor were found to contain illegal values.
RxBuffMismatch	[SONIC hardware] The number of times when processing Receive Descriptors after a problem due to a SONIC bug has been encountered that the buffer pointed to by the Receive Resource Descriptor and the Receive Descriptor are different.

Table 9-5: Parameters in output of the **show eth counters=diagnostic** command

Counter	Meaning
RxBuffAreaExceeded	[SONIC hardware] The number of times a packet was received that was so large it would have exceeded the receive buffer.
RxBuffExhausted	[SONIC hardware] The number of receive buffers exhausted interrupts.
RxDescExhausted	[SONIC hardware] The number of receive descriptors exhausted interrupts.
TxEOLOverrun	[SONIC hardware] The number of times the Ethernet hardware overran the end of the transmit packet list and caused a transmit error interrupt.
TxDescCorrupt	[SONIC hardware] The number of times when a transmit error occurred that the transmit descriptor was found to be corrupt.
TxSpuriousBCMError	[SONIC hardware] The number of times the Ethernet hardware reported a byte count mismatch in the packet buffer, but no error existed.
TxBCMError	[SONIC hardware] The number of times the Ethernet hardware reported a byte count mismatch in the packet buffer and such an error was found.
PacketMonitoredBad	[SONIC hardware] The number of times the SONIC receive unit monitored a transmitted packet as bad.
LinkChanges	(M860F and 79C972 hardware) The number of times the Ethernet link status (up, down, or unknown) changed.
AutoNegCompletes	(M860F and 79C972) The number of times the Ethernet hardware completed negotiating a set of operating parameters with the link partner (a hub or switch) to which it is connected.
ParallelDetFaults	(M860F and 79C972 hardware) The number of times the Ethernet hardware reported a parallel detection fault while negotiating with its link partner.
MIITimeouts	[M860F hardware] The number of times a command from the Ethernet hardware's controller to its transceiver did not complete before the timeout timer expired.
MIIPhyDetectTransitions	[79C972 hardware] The number of times the Ethernet hardware's controller detected changes in the presence of the transceiver.
MIIManagementReadErrors	[79C972 hardware] The number of times the Ethernet hardware detected an error in communications between its controller and transceiver.
ReceiveCollisions	[79C972 hardware] The number of times the Ethernet hardware detected a late collision on the network while receiving data.

Figure 9-11: Example output from the **show eth counters=dot3stat** command

ETH instance 0: 1295 seconds Last change at: 0 seconds			
dot3 Statistics MIB Counters			
Receive:		Transmit:	
InternalMacRxErrors	0	InternalMacTxErrors	0
FrameTooLongs	0	DeferredTransmissions	5
AlignmentErrors	0	SingleCollisionFrames	11
FCSErrors	2	MultipleCollisionFrames	12
Missed	0	LateCollisions	0
UnwantedBroad	5114	ExcessiveCollisions	0
UnwantedMulticasts	5	CarrierSenseErrors	0
RxQueueLength	0	ExcessiveDeferrals	0

Table 9-6: Parameters in output of the **show eth counters=dot3stat** command

Counter	Meaning
InternalMacRxErrors	Number of frames for which reception failed due to an internal error.
FrameTooLongs	Number of frames received that exceeded the maximum permitted frame size.
AlignmentErrors	Number of received frames with alignment and FCS errors.
FCSErrors	Number of received frames with FCS but not alignment errors.
Missed	Number of packets not received due to: (1) lack of memory resources to buffer the packet, (2) a FIFO overrun, (3) receiver was disabled.
UnwantedBroad	Number of broadcast frames received on this interface for a protocol not configured to any module.
UnwantedMulticasts	Number of multicast frames received on this interface for a protocol not configured to any module.
RxQueueLength	Length of the queue of receive packets between the interrupt routine and the idle level receive packet processing routine.
InternalMacTxErrors	Number of frames not transmitted due to internal error.
DeferredTransmissions	Number of frames delayed by busy medium.
SingleCollisionFrames	Number of successfully transmitted frames that were inhibited by exactly one collision.
MultipleCollisionFrames	Number of successfully transmitted frames that were inhibited by more than one collision.
LateCollisions	Number of times that a collision is detected later than 512 bit times into the transmission of a packet.
ExcessiveCollisions	Number of frames not transmitted due to excessive collisions.
CarrierSenseErrors	Number of times that carrier sense was lost or never asserted during the transmission of a frame.
ExcessiveDeferrals	Number of times transmission of a packet was aborted due to excessive deferrals.

Figure 9-12: Example output from the **show eth counters=interface** command

ETH instance 0: 1239 seconds		Last change at: 0 seconds	
Interface MIB Counters			
Receive:		Transmit:	
ifInOctets	2357609	ifOutOctets	872296
ifInUcastPkts	2588	ifOutUcastPkts	3985
ifInNUcastPkts	420	ifOutNUcastPkts	2
ifExtnsMulticastsRxOKs	5	ifExtnsMulticastsTxOKs	0
ifExtnsBroadcastsRxOKs	5308	ifExtnsBroadcastsTxOKs	2
ifInDiscards	0	ifOutDiscards	0
ifInErrors	2	ifOutErrors	0
ifInUnknownProtos	4888	ifOutQLen	1

Table 9-7: Parameters in output of the **show eth counters=interface** command

Counter	Meaning
ifLastChange	Value of sysUpTime at the time the interface entered its current operational state.
ifInOctets	Number of octets received on this interface.
ifInUcastPkts	Number of unicast packets delivered to a higher-layer protocol.
ifInNUcastPkts	Number of non-unicast packets delivered to a higher-layer protocol.
ifExtnsMulticastsReceivedOKs	Number of frames successfully received for a multicast address other than a broadcast address.
ifExtnsBroadcastsReceivedOKs	Number of frames successfully received for a broadcast address other than a multicast address.
ifInDiscards	Number of inbound packets discarded without errors that prevented them from being deliverable to higher-layer protocol.
ifInErrors	Number of inbound packets with errors that prevented them from being deliverable to a higher-layer protocol.
ifInUnknownProtos	Number of packets discarded because they were for an unconfigured protocol.
ifOutOctets	Number of octets transmitted, including framing.
ifOutUcastPkts	Number of unicast packets transmitted or discarded.
ifOutNUcastPkts	Number of non-unicast packets transmitted or discarded.
ifExtnsMulticastsTransmittedOKs	Number of frames successfully transmitted to a multicast address other than a broadcast address.
ifExtnsBroadcastsTransmittedOKs	Number of frames successfully transmitted to a broadcast address other than a multicast address.
ifOutDiscards	Number of packets discarded though no errors had been detected preventing their being transmitted.
ifOutErrors	Number of packets not transmitted because of errors.
ifOutQLen	Length of the output packet queue.

**Related Commands**

- [reset eth counters](#)
- [show eth configuration](#)
- [show eth receive](#)

## show eth macaddress

---

**Syntax** `SHoW ETH[=n] MACaddress`

where *n* is the number of the Ethernet interface

**Description** This command displays the default MAC address for a specific Ethernet interface. If the interface is not specified, default MAC addresses for all Ethernet interfaces are displayed ([Figure 9-13](#)).

Figure 9-13: Example output from the **show eth macaddress** command

```
MAC address for ETH instance 0:

Address
-----
00-00-cd-00-0d-0e
-----
```

**Related Commands**

- [show eth configuration](#)
- [show eth counters](#)
- [show eth receive](#)

## show eth receive

**Syntax** SHow ETH[=*n*] RECeive

where *n* is the number of the Ethernet interface

**Description** This command displays the multicast addresses that an Ethernet interface has been configured to receive. If the interface is not specified, the multicast addresses for all Ethernet interfaces are displayed ([Figure 9-14](#)).

Note that the list includes the broadcast address and any unicast addresses specified by the software modules that have configured to the Ethernet interface. Unicast addresses are distinguishable from multicast addresses by their first octet. The first octet of a unicast address is even, whereas for a multicast address it is odd. The broadcast address is a special multicast address that is received by all stations on an Ethernet. The router is always configured to receive broadcast packets, even if no software modules are using the interface, so the list always includes the broadcast address.

In [Figure 9-14](#), the first entry is the default MAC address assigned to the router and the next two entries are the multicast addresses required for OSPF routing. The third entry is the MAC address configured by the DECnet routing module, that corresponds to the DECnet address assigned to the router and the fourth entry is a multicast address required for DECnet routing. The last entry is the broadcast address.

Figure 9-14: Example output from the **show eth receive** command

```
Receive addresses for ETH instance 0:
```

```
Address
```

```
-----  
00-00-cd-00-0d-0e  
01-00-5e-00-00-05  
01-00-5e-00-00-06  
aa-00-04-00-97-0b  
ab-00-00-03-00-00  
ff-ff-ff-ff-ff-ff  
-----
```

**Related Commands** [show eth configuration](#)  
[show eth counters](#)

## show eth state

**Syntax** `SHoW ETH[=n] STAtE`

where *n* is the number of the Ethernet interface

**Description** This command displays the state of the link between the Ethernet interface and its link partner (the Ethernet hub or switch to which it is connected). If an Ethernet interface is not specified, information about all Ethernet interfaces is displayed.

The information displayed depends on the capabilities of the Ethernet interface hardware and how the interface is configured (Figure 9-15, Figure 9-16 on page 9-71, Figure 9-17 on page 9-71, Table 9-8 on page 9-72).

Figure 9-15: Example output from the **show eth state** command for 68360-based or SONIC-based hardware

```
State for ETH instance 0:

Link ..... up
Speed ..... 10 Mbps
Duplex mode ..... half
```

Figure 9-16: Example output from the **show eth state** command for 100Mbps interfaces in the default configuration (with link speed auto-negotiation enabled)

```
State for ETH instance 0:

Configured speed/duplex..... 100 Mbps, half duplex
Actual speed/duplex ..... 10 Mbps, half duplex
Duplex mode ..... full
Auto-negotiation ..... complete

Link partner capabilities
Auto-negotiation ..... yes
100BASE-TX full duplex ..... yes
100BASE-TX ..... yes
10BASE-T full duplex ..... yes
10BASE-T ..... yes
```

Figure 9-17: Example output from the **show eth state** command for 100Mbps interfaces with link speed set manually

```
State for ETH instance 0:

Link ..... up
Speed ..... 100 Mbps
Duplex mode ..... half
Auto-negotiation ..... disabled
```

Figure 9-18: Example output from the **show eth state** command for an interface on a PIC

```

State for ETH instance 1:

Link ..... up
LinkUp timeout ..... 2592000 seconds
Speed ..... 10 Mbps
Duplex mode ..... half

```

Table 9-8: Parameters in output of the **show eth state** command

Parameter	Meaning
Configured speed/duplex	The port speed mode configured for this port. Either "Autonegotiate" or a combination of a speed (either 10 Mbps, 100 Mbps, or 1000 Mbps) and a duplex mode (half duplex or full duplex) and optionally by autonegotiation.
Actual speed/duplex	The port speed and duplex mode that this port is actually running at. A combination of a speed (either 10 Mbps, 100 Mbps, or 1000 Mbps) and a duplex mode (half duplex or full duplex).
Link	The current state of the link; one of up or down.
LinkUp timeout	The period of time for which the link stays up after it last receives a packet successfully.
Speed	The speed at which the link is operating.
Duplex mode	Whether the duplex mode in which the link is operating is half duplex or full duplex.
Auto-negotiation	Whether the negotiation process is in progress or complete between the Ethernet interface and its link partner to set link parameters.
Link partner capabilities	Information about the capabilities of the link partner.
Auto-negotiation	Whether the link partner can automatically negotiate link parameters.
100BASE-TX full duplex	Whether the link partner can operate at 100 Mbps full-duplex.
100BASE-TX	Whether the link partner can operate at 100 Mbps half-duplex.
10BASE-TX full duplex	Whether the link partner can operate at 10 Mbps full-duplex.
10BASE-TX	Whether the link partner can operate at 10 Mbps half-duplex.

**Examples** To show the state of Ethernet interface 1, use the command:

```
sh eth=1 sta
```

**Related Commands** [set eth linkup](#)  
[set eth speed](#)



## show interface

**Syntax** `SHoW INtErface[={ifIndex|interface}] [COUnTERS  
[PRIOrityqueue]`

where:

- *ifIndex* is a decimal value specifying the entry in the interface MIB
- *interface* is a valid interface name

<b>Description</b>	This command displays the contents of the interface MIB. If an interface is not specified, summary information for all interfaces is displayed (Figure 9-19 on page 9-74, Table 9-9 on page 9-74). If an interface is specified, detailed information is displayed about it including the counters (Figure 9-20 on page 9-75, Table 9-10 on page 9-75).
--------------------	---

The **counters** parameter displays interface counters for all interfaces (Figure 9-21 on page 9-76, Table 9-11 on page 9-76). When the interface is a VLAN, the command displays counters for packets switched by the CPU, not those switched in hardware at wire speed.

The **priorityqueue** parameter is valid for Ethernet and VLAN interfaces only. It displays a statistical sampling of the average lengths of queues and average time a packet spends in each queue. The means for the Average Time in Queue statistics are weighted by the number of packets received in each queue.

The **counters** and **priorityqueue** parameters are mutually exclusive.

Figure 9-19: Example output from the **show interface** command

```

Interfaces                                     sysUpTime:                05:05:32

DynamicLinkTraps.....Disabled
TrapLimit.....20

Number of unencrypted PPP/FR links.....0

ifIndex Interface ifAdminStatus ifOperStatus ifLastChange
-----
 1      eth0      Up           Up           00:00:03
 2      bri0      Up           Down          01:06:04
 2      eth1      Down          Down          00:00:00
 3      syn0      Up           Down          00:00:00
-----

Interface name summary

Interface Full name
-----
asyn0      asyn0
asyn1      asyn1
eth0       eth0
eth1       eth1
eth2       bay1.eth0
syn0       bay0.syn0
eth3       nsm0.bay0.eth0
asyn6      nsm0.bay1.asyn0
asyn7      nsm0.bay1.asyn1
asyn8      nsm0.bay1.asyn2
asyn9      nsm0.bay1.asyn3
bri0       nsm0.bay3.bri0
-----

```

Table 9-9: Parameters in output of the **show interface** command

Parameter	Meaning
sysUpTime	Elapsed time since the last router restart.
DynamicLinkTraps	Whether link traps are enabled for dynamic interfaces.
TrapLimit	Maximum number of link up/down traps for dynamic interfaces that is generated in one minute.
Number of unencrypted PPP/FR links	Total number of PPP interfaces and Frame Relay circuits that are configured to send plaintext. Does not include disabled PPP interfaces or enabled Frame Relay circuits on disabled Frame Relay interfaces.
ifIndex	Index of the interface in the interface table.
Interface	Name of the interface.
ifAdminStatus	Whether the administratively-set (configured) state of the interface is Up, Down, or Testing.
ifOperStatus	Whether the operational state of the interface is Up, Down, Testing, Unknown, or Dormant.
ifLastChange	Value of <i>sysUpTime</i> at the time the interface entered its current operational state.

Figure 9-20: Example output from the **show interface** command for a specific interface

```

Interface..... eth0
  ifIndex..... 1
  ifMTU..... 1500
  ifSpeed..... 10000000
  ifAdminStatus..... Up
  ifOperStatus..... Up
  ifLinkUpDownTrapEnable... Disabled
  TrapLimit..... 20

Interface Counters

  ifInOctets ..... 21484          ifOutOctets ..... 13775
  ifInUcastPkts ..... 165          ifOutUcastPkts ..... 134
  ifInNUcastPkts ..... 19          ifOutNUcastPkts ..... 0
  ifInDiscards ..... 0            ifOutDiscards ..... 0
  ifInErrors ..... 0              ifOutErrors ..... 0
  ifInUnknownProtos ... 30

```

Table 9-10: Parameters in output of the **show interface** command for a specific interface

Parameter	Meaning
Interface	Name of the interface.
ifIndex	Index of the interface in the interface table.
ifMTU	Size in octets of the largest packet that can be transmitted on the interface.
ifSpeed	Estimate of the interface's current speed in bits per second, or 0 if the interface is down.
ifAdminStatus	Whether the administratively-set (configured) state of the interface is Up, Down, or Testing.
ifOperStatus	Whether the operational state of the interface is Up, Down, Testing, Unknown, or Dormant.
ifLinkUpDownTrapEnable	Whether link traps are enabled for the interface.
TrapLimit	Maximum number of link up/down traps for dynamic interfaces that is generated in one minute.
Interface Counters	Counters for the interface.
ifInOctets	Number of octets (bytes) received by the interface.
ifInUcastPkts	Number of unicast packets received by the interface.
ifInNUcastPkts	Number of multicast packets received by the interface.
ifInDiscards	Number of packets discarded by the interface. Not applicable for a port interface.
ifInErrors	Number of packets received with errors by the interface.
ifUnknownProtos	Number of packets received by the interface but discarded because their protocol is unsupported.
ifOutOctets	Number of bytes transmitted by the interface.
ifOutUcastPkts	Number of unicast packets transmitted by the interface.

Table 9-10: Parameters in output of the **show interface** command for a specific interface (Continued)

Parameter	Meaning
ifOutNUcastPkts	Number of multicasts transmitted by the interface.
ifOutDiscards	Number of output packets discarded by the interface. Not applicable for a port interface.
ifOutErrors	Number of packets that should have been transmitted but were not because of errors.

Figure 9-21: Example output from the **show interface counter** command

```

Interface Counters

Interface: eth0
  ifInOctets ..... 22852      ifOutOctets ..... 15565
  ifInUcastPkts ..... 184     ifOutUcastPkts ..... 148
  ifInNUcastPkts ..... 19     ifOutNUcastPkts ..... 0
  ifInDiscards ..... 0       ifOutDiscards ..... 0
  ifInErrors ..... 0         ifOutErrors ..... 0
  ifInUnknownProtos ... 30

Interface: ISDN Basic Rate Interface
  ifInOctets ..... 0         ifOutOctets ..... 0
  ifInUcastPkts ..... 0     ifOutUcastPkts ..... 0
  ifInNUcastPkts ..... 0     ifOutNUcastPkts ..... 0
  ifInDiscards ..... 0     ifOutDiscards ..... 0
  ifInErrors ..... 0       ifOutErrors ..... 0
  ifInUnknownProtos .... 0

```

Table 9-11: Parameters in output of the **show interface counter** command

Parameter	Meaning
Interface	Name of the interface.
ifInOctets	Number of octets (bytes) received by the interface.
ifInUcastPkts	Number of unicast packets received by the interface.
ifInNUcastPkts	Number of multicast packets received by the interface.
ifInDiscards	Number of packets discarded by the interface. Not applicable for a port interface.
ifInErrors	Number of packets received with errors by the interface.
ifUnknownProtos	Number of packets received by the interface but discarded because their protocol is unsupported.
ifOutOctets	Number of bytes transmitted by the interface.
ifOutUcastPkts	Number of unicast packets transmitted by the interface.
ifOutNUcastPkts	Number of multicasts transmitted by the interface.
ifOutDiscards	Number of output packets discarded by the interface. Not applicable for a port interface.
ifOutErrors	Number of packets that should have been transmitted but were not because of errors.

Figure 9-22: Example output from the **show interface priorityqueue** command

Interface ..... eth0				
Priority Queue debug enabled ..... YES				
Maximum total priority queue length ... 256				
Average Queue Length (in packets)				
Queue	Current	Last 5 secs	Last 30 secs	Last 60 secs
-----				
P0	0	0	0	Unknown
P1	1	2	1	Unknown
P2	1	4	2	Unknown
P3	11	5	2	Unknown
P4	5	15	6	Unknown
P5	0	24	13	Unknown
P6	139	137	168	Unknown
P7	50	19	55	Unknown
Mean	26	26	31	Unknown
Average Time in Queue (ms)				
Queue	Current	Last 5 secs	Last 30 secs	Last 60 secs
-----				
P0	0	0	0	Unknown
P1	0	1	0	Unknown
P2	0	0	0	Unknown
P3	12	5	2	Unknown
P4	0	4	1	Unknown
P5	0	0	2	Unknown
P6	8907	8919	4848	Unknown
P7	16514	19760	20001	Unknown
Mean	2	2	2	Unknown
-----				

Table 9-12: Parameters in output of the **show interface priorityqueue** command

Parameter	Meaning
Interface	The name of the interface.
Priority Queue debug enabled	Whether priority queue debugging is enabled.
Maximum total priority queue length	The total number of packets that can be queued.
<b>Average Queue Length (in packets)</b>	
Queue	The priority level of the queue.
Current	The number of packets currently in the queue.
Last 5 secs	The average number of packets in the queue for the last 5 seconds.
Last 30 secs	The average number of packets in the queue for the last 30 seconds.
Last 60 secs	The average number of packets in the queue for the last 60 seconds. "Unknown" is displayed in the column when priority queue debugging has not been enabled long enough to calculate an average (less than a minute).
Mean	The average of each column.

Table 9-12: Parameters in output of the **show interface priorityqueue** command (Continued)

Parameter	Meaning
<b>Average Time in Queue (ms)</b>	
Queue	The priority level of the queue.
Current	The time that packets currently queued have been waiting.
Last 5 secs	The average time that packets have been in the queue for the last 5 seconds.
Last 30 secs	The average time that packets have been in the queue for the last 30 seconds.
Last 60 secs	The average time that packets have been in the queue for the last 60 seconds. "Unknown" is displayed in the column when priority queue debugging has not been enabled long enough to calculate an average (less than a minute).
Mean	The average of each column. The Average Time in Queue statistics are weighted by the number of packets received in each queue.

**Examples** To display the general state of all interfaces, use the command:

```
sh int
```

**Related Commands** [disable interface debug](#)  
[disable interface linktrap](#)  
[enable interface debug](#)  
[enable interface linktrap](#)  
[set interface traplimit](#)

## show syn

**Syntax** `SHoW SYN[=n]`

where *n* is the number of the synchronous interface

**Description** This command displays the configuration of a synchronous interface and the state of the interface control signals (Figure 9-23, Table 9-13). If an interface is not specified, then information is displayed for all interfaces.

Figure 9-23: Example output from the **show syn** command

SYN instance 1:			
3088 seconds	Last change at:	0 seconds	
Module ..... PPP			
State ..... enabled			
Active ..... yes			
Interface type ..... RS-232 DTE			
Clocks ..... receive			
Actual baud rate ..... determined externally			
Configured baud rate ..... 48000			
Max output queue length ... 100			
Min interframe delay ..... no delay			
Date sense ..... normal			
Tx clock edge ..... rising			
Hardware type ..... 68302			
Debug ..... on			
Control signals	State	Output mode	Transitions
CTS (in) .....	off		8
DCD (in) .....	off		2
DSR (in) .....	off		6
RTS (out) .....	off	follow CTS	6
LL (out) .....	off	layer 2 control	6
RL (out) .....	off		4

Table 9-13: Parameters in output of the **show syn** command

Parameter	Meaning
Module	Name of the layer 2 module attached to this interface, if any.
State	Whether the interface is enabled.
Active	Status of the interface. Yes if enabled and a layer 2 module attached.
Interface type	Type of the transition cable attached to the interface.
Clocks	Direction of the interface clocks, receive or generate.
Actual baud rate	Actual baud rate of the interface, may be different from configured baud rate for generate clocks.
Configured baud rate	Baud rate configured by the <a href="#">set syn command on page 9-50</a> (or the default).
Max output queue length	Maximum output queue length configured by the <a href="#">set syn command</a> (or the default).
Min interframe delay	Minimum transmit interframe delay configured by the <a href="#">set syn command</a> (or the default).
Date sense	Whether the sense of the transmitted data is normal or inverted.

Table 9-13: Parameters in output of the **show syn** command (Continued)

Parameter	Meaning
Tx clock edge	Transmit clock edge used to clock out the transmitted data; either rising or falling.
Hardware type	Whether the hardware device for the interface is 68302, 68360, or 68562.
Debug	Whether debug messaging is on.
Control signals	Modem control signals available for the current interface type.
State	Whether the configured state of the modem control signals is on, off, not monitored, or not controlled.
Output mode	Control mode for modem control output signals; either "always off", "always on", "layer 2 control" and "follow XXX" where "XXX" is the name of the paired modem control input signal.
Transitions	Number of transitions the modem control signal has made.

**Related Commands**   [show syn counter](#)  
[set syn](#)



## show syn counter

**Syntax** `SHoW SYN[=n] COUnTer[={INTeRface|SYN}]`

where *n* is the number of the synchronous interface

**Description** This command displays the MIB counters for a synchronous interface. The interface number is optional. If an interface is not specified, then information is displayed for all interfaces. If a category is not specified, all counters are displayed.

If **interface** is specified, counters from the interfaces table from the interface MIB are displayed (Figure 9-24, Table 9-14).

If **syn** is specified, the counters from the synchronous interface table of the router's enterprise MIB and the counters from the synchronous port table of the RS-232-like hardware devices MIB are displayed (Figure 9-25 on page 9-82, Table 9-15 on page 9-83).

Figure 9-24: Example output from the **show syn counter=interface** command

SYN instance 0: 3098 seconds Last change at: 0 seconds			
Interface MIB Counters			
Receive:		Transmit:	
ifInOctets	21349408	ifOutOctets	18683238
ifInUcastPkts	65391	ifOutUcastPkts	34271
ifInNUcastPkts	0	ifOutNUcastPkts	0
ifInDiscards	0	ifOutDiscards	0
ifInErrors	93	ifOutErrors	0
ifInUnknownProtos	0	ifOutQLen	1

Table 9-14: Parameters in output of the **show syn counter=interface** command

Counter	Meaning
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.
ifInOctets	The number of octets received on this interface.
ifInUcastPkts	The number of unicast packets delivered to a higher-layer protocol.
ifInNUcastPkts	The number of non-unicast packets delivered to a higher-layer protocol.
ifExtnsMulticastsReceivedOKs	The number of frames successfully received for a multicast address other than a broadcast address.
ifExtnsBroadcastsReceivedOKs	The number of frames successfully received for a broadcast address other than a multicast address.
ifInDiscards	The number of inbound packets discarded without errors that prevented them from being deliverable to higher-layer protocol.
ifInErrors	The number of inbound packets with errors that prevented them from being deliverable to a higher-layer protocol.

Table 9-14: Parameters in output of the **show syn counter=interface** command

Counter	Meaning
ifInUnknownPkts	The number of packets discarded because they were for an unconfigured protocol.
ifOutOctets	The number of octets transmitted, including framing.
ifOutUcastPkts	The number of unicast packets transmitted or discarded.
ifOutNUcastPkts	The number of non-unicast packets transmitted or discarded.
ifExtnsMulticastsTransmittedOKs	The number of frames successfully transmitted to a multicast address other than a broadcast address.
ifExtnsBroadcastsTransmittedOKs	The number of frames successfully transmitted to a broadcast address other than a multicast address.
ifOutDiscards	The number of packets discarded though no errors had been detected preventing their being transmitted.
ifOutErrors	The number of packets not transmitted because of errors.
ifOutQLen	The length of the output packet queue.

Figure 9-25: Example output from the **show syn counter=syn** command

SYN instance 0: 1051 seconds Last change at: 0 seconds			
Synchronous Counters			
Receive:		Transmit:	
Frames	0	Frames	0
OverlengthFrames	0	CTSLosses	0
UnderlengthFrames	0	Underruns	0
CRCErrors	0	LostInterrupts	0
Aborts	0	DroppedFrames	0
NonOctetAligneds	0	NoPackets	0
Overruns	0	QueueLength	0
NonmatchAddresses	0	Recovers	0
Misseds	0	SDMABusErrors	0
TooFewBuffers	0	CommandTimeouts	0
QueueLength	0	LastCommand	0

Table 9-15: Parameters in output of the **show syn counter=syn** command for 68302- and 68360-based synchronous interfaces

Counter	Meaning
Frames	The number of frames received or queued for transmission by a higher layer.
OverlengthFrames	The number of frames discarded because they were too long.
UnderlengthFrames	The number of frames discarded because they were too short.
CRCErrors	The number of frames received with a CRC error.
Aborts	The number of received frames terminated by an abort.
NonOctetAligneds	The number of frames discarded because they did not contain an integral number of octets.
Overruns	The number of frames not received because of a receiver overrun.
NonmatchAddresses	The number of frames not received because the address of the HDLC frame was not one that was configured to be received.
Misseds	The number of frames that could not be received due to lack of a receive buffer.
TooFewBuffers	The number of received frames discarded because the number of buffers in the router reached a critical level.
QueueLength	The number of frames on the receiver or transmitter queue.
CTSLosts	The number of frames during which the CTS input was negated.
Underruns	The number of frames not transmitted because a transmitter underrun occurred during transmission.
LostInterrupts	The number of frames that failed to transmit because no interrupt was received to signal transmission complete.
DroppedFrames	The number of frames discarded because the maximum transmit queue length was exceeded.
NoPackets	The number of times the 68302 or 68360 reported a transmit error, but there was no packet being transmitted or the errored packet could not be identified.
Recovers	The number of times the 68302 or 68360 was reset due to a serious error or a <a href="#">reset syn command on page 9-38</a> .
SDMABusErrors	The number of bus errors experienced by the 68302 or 68360 for this interface.
CommandTimeouts	The number of commands to the 68302 or 68360 that failed to complete.
LastCommand	The code of the command that was to be issued when a command timeout was detected.

**Related Commands** [reset syn counters](#)  
[show syn](#)

