GIGAswitch/FDDI System

SNMP Guide

part number: EK–GSNMP–MG .A01

This document contains information for managing a GIGAswitch/FDDI System after it is installed on a network.

Revision/Update Information: This is a new document.

Digital Equipment Corporation
Maynard, Massachusetts

First Printing, April 1995

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

Preface v

1 Introduction

SNMP Basics 1–1
Setting up NMS 1–2
SNMP Support 1–3
IP Connectivity 1–3
Privileged Port 1–3
Supported MIBs 1–3
Community Name 1–3

2 Network Management Station

Introduction 2–1
Preparing for Network Management Station (NMS) Use 2–2
Obtaining the Management Information Bases (MIBs) 2–2
Obtaining MIBs 2–2
Initial NMS Commands 2–5
NMS Use and Command Syntax 2–5
DECmcc and POLYCENTER 2–5
HP OpenView 2–5
SunNet Manager 2–6
Default Gateway Configuration 2–7
Setting Community Name 2–7
Read-Only Users 2–9
Read-Write Users 2–9
Traps 2–11

3 GIGAswitch/FDDI MIB Objects

Implemented MIB Object Groups 3–2
GIGAswitch MIB Object Tree 3–2
Important Configuration Management Objects 3–4
Important Network Monitoring Objects 3–5
Important Topology Mapping Objects 3–7
Bridge Traps 3–9
Troubleshooting Bridge Problems 3–10

A GIGAswitch MIB Objects

FilterByReferencedExpression Group Reachability Matrix Objects A–2
FilterByReferencedExpression Group Filtering Objects for SAP, SNAP, DA and SA A–3
FilterByReferencedExpression Group Reachability Row Objects A–5
FilterByBitmapValue DA Filtering Objects A–8
Service Class Objects A–10
Port Number Objects A–11
FilterByBitmapValue SA Filtering Objects A–12
FilterByBitmapValue SAP Filtering Objects A–14
FilterByBitmapValue SNAP Filtering Objects A–17
Default Filter Matrix Objects A–20
cutThrough Objects A–22
Gigabox Objects A–24

B MIB Object Identifiers

Glossary of GIGAswitch FDDI Terms

Figures

3–1 GIGAswitch MIB Object Tree 3–3

Tables

1–1 Other Sources of Information 1–1
3–1 MIB Object Implementation 3–2
3–2 Important Bridge Configuration Objects 3–4
3–3 Important Bridge Performance Objects 3–6
3–4 Important Mapping Objects 3–8
3–5 Important Bridge Traps 3–9
3–6 Important Bridge Troubleshooting Objects 3–10
B–1 GIGAswitch MIB Object Identifiers B–1
B–2 MIB-II Object Identifiers B–9
B–3 Bridge MIB Object Identifiers B–15
B–4 Dec_vendor MIB Object Identifiers B–18
B–5 FDDI MIB Object Identifiers B–28

Preface

Intended Audience

This guide is intended for network managers who will manage a GIGAswitch/FDDI System in an extended local area network (LAN).

Structure of This Guide

This guide describes how to perform common management tasks for the GIGAswitch/FDDI System. The following table shows where to find information:

Refer to:	For Information About:
Chapter 1	Introduction to Simple Network Management Protocol (SNMP).
Chapter 2	How to prepare the Network Management Station (NMS).
Chapter 3	SNMP MIB objects that are useful in managing the GIGAswitch/FDDI System.
Appendix A	MIB Objects
Appendix B	MIB Object Identifiers
Glossary	Definitions of GIGAswitch/FDDI System terms

Additional Resources

The following documents provide additional information:

- *Configuring the SNMP Agent*, AA–PR84A–TE
- RFC1157: SNMP Standard[1]
- RFC1285: FDDI MIB
- RFC1286: Bridge MIB
- RFC1213: MIB-II
- RFC1407: DS3 MIB
- DEC ELAN Vendor MIB Version 2.7
- GIGAswitch/FDDI MIB Version 1.3

[1] RFCs can be obtained from NIC.DDN.MIL in the rfc directory using anonymous ftp.

- ATM MIB - (Included in GIGAswitch/FDDI MIB)
- SONET MIB - (Included in GIGAswitch/FDDI MIB)

Documentation for your network management station (NMS) should also be available for regular use.

Conventions

The following conventions are used in this guide:

Bold typeface	A word or phrase is being emphasized. Also indicates a MIB object.
Italic typeface	The complete titles of manuals.
Return	You press the return key on the keyboard.
Ctrl/O	You must hold down the key labeled Ctrl while you press another key or a pointing device button.

1
Introduction

This guide explains how Simple Network Management Protocol (SNMP) is used to manage the GIGAswitch/FDDI System (including a brief introduction to SNMP), and how to set up a management station and the GIGAswitch/FDDI System for management. It also describes the MIB objects supported by the GIGAswitch/FDDI SNMP agent.

The following Table 1–1 provides sources of additional infromation about SNMP management.

Table 1–1 Other Sources of Information

If You Want To Do This	Refer To
Learn about SNMP and manageable objects	*The Simple Book*, by Marshall T. Rose
Learn about the icmp, udpd, ip, arp, tftp, or BootP groups	Appropriate RFC
Look up an SNMP object, either because it's not described in this manual, or because its description is abbreviated	Appropriate public or private MIB
Configure your network	*Fiber Distributed Data Interface Network and Configuration Guidelines*
Learn how to manipulate SNMP objects using your network management station software	Appropriate user documentation

SNMP Basics

SNMP is a protocol for exchanging information between two management entities: the manager and the agent. The manager is an application that runs on a system referred to as the network management station (NMS). It provides the interface through which a human manager (or another program) performs management activities. The agent is an application that runs on a system which is the object of the management activities. The agent runs on the device which is to be managed: a host, gateway, terminal server, repeater, hub, switch, concentrator, or bridge.

SNMP allows a NMS to view (GET) information from an agent or to modify (SET) information within an agent. Frequently a request to view or modify such information is accompanied by a "community name," which acts as a password, thus providing a basic level of security.

The information that is made available to the NMS by an agent is organized in a hierarchical collection of objects, known as a management information base (MIB). The objects of the MIB are organized in a logical tree. Each object is indexed by its position within the tree. Frequently specific subtrees are also identified as MIBs themselves.

The MIB was defined by the Internet Engineering Task Force (IETF) for SNMP management. It is the repository for all information used to manage a network device. It describes the name, object identifier, data type, and accessibility of every data item that can be read or written via SNMP. Subtrees of the MIB are defined by IETF, vendors, or other organizations.

SNMP supports four operations, three initiated by the NMS; one initiated by the agent:

- GET is used by the NMS to retrieve from the agent the value of a specified MIB object.
- GETNEXT is used by the NMS to retrieve from the agent the value of the object which follows (in the tree) the MIB identifier specified in the command.
- SET is used by the NMS to request the agent to alter the value of a specified MIB object. This action often has side effects on the operation of the device being managed. It can also cause an object to be created.
- TRAP is a message sent by the agent (to specified NMSs) which notifies the NMS of some event taking place on the managed device.

The objects and traps supported by the GIGAswitch/FDDI System are described later in this guide.

Setting up NMS

In order to use SNMP to manage the GIGAswitch/FDDI System a network management station must be configured. Such a station must satisfy three basic requirements to manage the switch:

1. It must be able to send and receive SNMP packets, using UDP/IP.
2. It must have IP network access to the switch to be managed.
3. It must be loaded with all MIBs that are served by the switch.

In addition:

- The NMS must know the IP address of the switch.
  and
- The NMS must know the community name of the community corresponding to the desired operations (for example, read-write).

SNMP Support

There are numerous management stations that support SNMP. Among them are:

- DEC Polycenter MCC
- DEC Polycenter Netview
- HP OpenView
- SunNet Manager

Any of these (and many other) management stations are suitable for managing a GIGAswitch/FDDI System. This guide provides examples from several different management stations in explaining various operations.

IP Connectivity

The NMS must have IP access to the GIGAswitch/FDDI System. It may connect via a router, or it may be on the same extended LAN. The port through which the NMS connects to the switch must be assigned an IP address. The NMS uses this IP address to address the switch.

Privileged Port

The port through which the NMS connects to the switch should be designated a "privileged port" (see *GIGAswitch/FDDI System Out-of-Band Management (OBM) Guide*.)

Supported MIBs

The GIGAswitch/FDDI supports the following MIBs:

- **MIB-II** (RFC1213)
- **FDDI MIB** (RFC1285)
- **Bridge MIB** (RFC1286)
- **DEC ELAN vendor MIB** (proprietary)
- **GIGAswitch MIB** (proprietary)

Community Name

The read community name is by default *public*. The read-write community name is the lowest hardware address assigned to the GIGAswitch/FDDI System. This address can be read from a label on the CLK card or it can be determined using the OBM. For information on how to determined the address using the OBM refer to the *GIGAswitch/FDDI Out-of-Band Management Guide*.

2

Network Management Station

Introduction

This chapter describes how to prepare the network management station (NMS) for use and how to manage the GIGAswitch/FDDI System from the NMS.

Preparing for Network Management Station (NMS) Use

You must have an NMS that is installed and contains the correct software to manage the GIGAswitch/FDDI System via Simple Network Management Protocol (SNMP).

To manage the GIGAswitch/FDDI System with an NMS, you must perform the following initial activities:

- Obtain and add the required MIBs to your NMS.
- Set the GIGAswitch internet protocol (IP) address.

The GIGAswitch/FDDI System can be managed with any NMS that supports the SNMP.

Obtaining the Management Information Bases (MIBs)

The objects and entities that can be manipulated with SNMP commands are defined by the MIB.

The following MIBs are required for control of the GIGAswitch /FDDI System by the NMS:

- **MIB-II** (RFC1213)
- **FDDI MIB** (RFC1285)
- **Bridge MIB** (RFC1286)
- **DEC ELAN vendor MIB** (Version 2.7)
- **GIGAswitch MIB** (Version 1.3)

MIBs provide a mapping between network device entities that can be managed (monitored or changed), and names that can be used in commands to refer to those entities. Groups of entities or objects in a supported MIB that are not relevant to the GIGAswitch/FDDI System are not implemented.

Obtaining MIBs

The GIGAswitch MIB can be obtained over the Internet, using the same procedure you would with most MIBs. If you do not have all the MIBs required for full network management of the GIGAswitch/FDDI System, you will need to copy and compile any that are missing from your NMS, in addition to the GIGAswitch MIB.

To see what object groups from each MIB are implemented, see **Chapter 3**.

The DDN Network Information Center provides automated access to NIC documents and information using electronic mail. This method would help you to access RFCs, including public MIBs. This is useful for people who do not have a direct Internet link (people on BITNET, CSNET, or UUCP sites). To use the mail service:

1. Send a mail message to **SERVICE@NIC.DDN.MIL**.

2. In the *subject* field of the message, request the type of service you wish, followed by any arguments that apply. The message body is normally ignored, but if the *subject* field is empty, the first line of the message body is used as the request. Some possible service request lines include:

```
HELP
RFC 822
RFC INDEX
RFC 1119.PS
FYI 1
IETF 1IETF-DESCRIPTION.TXT
NETINFO DOMAIN-TEMPLATE.TXT
SEND RFC: RFC-BY-AUTHOR.TXT
SEND IETF/1WG-SUMMARY.TXT
SEND INTERNET-DRAFTS/DRAFT-IETF-NETDATA-NETDATA-00.TXT
HOST DIIS
```

Requests are processed automatically once a day. Large files are broken down into separate messages (however, few files are large enough to require this).

Digital's private MIB information is stored and up-to-date in the */private/mib* directory at gatekeeper.dec.com. Be sure to check the index file and the *README* file for the current contents.

Digital offers Internet ftpmail access to private MIB information, in ASCII text form, at *GATEKEEPER.DEC.COM*. To use ftpmail:

1. Send a mail message to *ftpmail@gatekeeper.dec.com*.
2. Ignore the subject line.
3. Include the word *connect* in the first line of the body.
4. Include **get** commands for each document required. For example:

   `get /private/mib/`*`filename`*

Requests are queued and processed every 30 minutes.

For more timely access, you can use anonymous ftp to access private MIB information as follows:

1. Use the Internet application FTP to connect to *gatekeeper.dec.com*. The Internet address is 16.1.0.2.
2. Log in as user *anonymous*.
3. Enter a password. You may use any string as the password, but we prefer that you use your electronic mail address.
4. Use the *cd* command to get to the directory */private/mib*.
5. Use the *ascii* command to specify that you are retrieving ASCII files.
6. Use the *get* command to get the file or files you require.
7. When finished, use the *quit* command to log out.

An example of an ftp transfer might look like this:

```
% ftp gatekeeper.dec.com
Connected to gatekeeper.dec.com
    220 GATEKEEPER.DEC.COM FTP Service Process
Name: anonymous
331 ANONYMOUS user ok, send real ident as password.
Password: munro@hitech.college.edu
230 User ANONYMOUS logged in at Tue 31-Aug-1993 00:30-EST, job 63
ftp> cd /private/mib
331 Default name accepted.  Send password to connect to it.
ftp> ascii
220 Type A ok.
ftp> get filename
200 Port 19.63 at host nnn.nn.nn.nn accepted.
    150 ASCII retrieve of /PRIVATE/MIB/filename started.
    226 Transfer completed.  40239 (8) bytes transferred.
    40239 bytes received in 23.65 seconds (5.8 Kbytes/s)
ftp> quit
%
```

You can access Request for Comments (RFCs) via ftp from the repository at *NIC.DDN.MIL* in the *rfc* directory.

Copy and load any of the other MIBs required by the GIGAswitch /FDDI System if they are not already on your NMS. Then load the GIGAswitch MIB onto your NMS using the management station's procedures.

Some NMS software contains several standard MIBs in a subdirectory. In DECmcc, these MIBs may be found in the MCC_COMMON directory. Prior to enrolling these MIBs, ensure they are the versions required by the GIGAswitch/FDDI System, as listed in the Preface of this guide.

If your NMS uses POLYCENTER, it has a command procedure for loading MIBs:

```
$ @MCC_COMMON:MCC_TCPIP_MTU
```

The following is an example of loading a MIB using DECmcc:

```
$ @MCC_COMMON:MCC_TCPIP_MTU.COM GIGASWITCH_MIB_V010
DECmcc TCP/IP MIB Translator Utility driver procedure V1.2.1
   31-AUG-1993 09:41:14
This procedure operates in a 2 phased manner --
Phase 1 translates a MIB definition into a management specification.
Phase 2 (as an option) translates the management specification into a
dictionary loadable file and updates the DECmcc data dictionary.

Phase 2 - Do you want to update the DECmcc data dictionary?( Y/N) [N]: Y
    .
    .
Phase 2 for file SONOMA_SW:[USER]GIGASWITCH_MIB_V010; completed.
       * * *
   31-AUG-1993 10:35:11
Normal successful completion.
```

Initial NMS Commands

When initially using the NMS, you may want to:

- Become familiar with your NMS's command syntax.
- Set a default gateway configuration.
- Set your community name (password) for read-only users, read-write users, and traps.

NMS Use and Command Syntax

Different software packages for NMS require different command syntax structure. The syntax often contains an action (set, get), a variable (a MIB object), a type (octet, integer), and a value. Set commands must often identify an action, an object, the type of string that object will accommodate, and the value you are giving it.

DECmcc and POLYCENTER

The syntax for these products is:

```
get node full-snmp-object-specification
getnext node full-snmp-object-specification
set node full-snmp-object-specification value,
 by password community-name
```

DECmcc also provides the assistance of a help key. You must enter **use mode form**, and this provides you with a separate area at the top of the screen where you can enter the verb, entity, arguments, and qualifiers of the command. Once you enter this information, you can press the help key (or PF2) and DECmcc displays all of the top-level subentities. If you enter *set* as the verb and press the help key while the cursor is in the *argument* line, DECmcc provides you with a list of the attributes that can be set on the current entity.

HP OpenView

HP OpenView primarily uses the object ID to refer to MIB objects. A list of object IDs for objects in groups implemented by the GIGAswitch/FDDI System can be found in **Appendix B**. A sample command would be:

```
snmpget giga1 public .1.3.6.1.4.1.36.2.15.3.3.3.1.5.2.0
```

Where:

- *giga1* is the switch (node) name.
- *public* is the community name, or community password.
- *.1.3.6.1.4.1....* is the object ID, or variable.

The common syntax of an OpenView set command is:

```
snmpset node community variable type value
```

Where:

- *node* is the switch, or node name.
- *community* is the community name, or community password.

- *variable* is the object ID.
- *type* is the type of string that variable holds (octet, integer, etc.).
- *value* is the value you are giving that object.

At the end of the command line you could include other sets of *variable type value* information as desired.

If the object ID has a leading decimal, it is interpreted as the full object ID. If it does not, it is an abbreviated ID. HP OpenView prepends *.iso.org.dod.internet* to all MIB objects.

SunNet Manager

SunNet Manager uses schemas, or lists of MIB objects and their valid values, and OID files that contain MIB objects and their object IDs. In order to prepare the SunNet Manager to use a new MIB's objects, you must run it through a program called MIB2SCHEMA, and this will create the OID (object ID) files and schema files. You must then specify the schema file in the SNMP host file (/var/adm/snm.hosts), by listing one entry per system. This can be verified by typing *snm_cmd -v*.

Set commands are entered from within the set tool. The command syntax is as follows:

agent/group/attribute key New Value=*newvalue*

Where:

- *agent* is the name of the overall MIB that contains the attribute, or object, being set.
- *group* is the group name of the MIB group that contains the attribute being set.
- *attribute* is the attribute, or actual MIB object.
- *key* is an optional key of an attribute. If the attribute is in a row in a table, a key is used.
- *newvalue* is the new value for the object.

To invoke the set tool from the command line, use the following syntax:

mgrhost%/usr/snm/bin/snm_set -t *target* -a *agent* -g *group*

Where:

- *target* is the GIGAswitch/FDDI System name.
- *agent* is the name of the MIB containing the object.
- *group* is the name of the MIB group containing the object.

A sample command line for the NMS would look as follows:

```
snm_cmd -f snmp-mibII.schema -g system -t giga1 -s
```

Where:

- *snmp-mibII.schema* is the scheme of the MIB whose objects that will be viewed.
- *system* is the MIB group of the objects that will be viewed.
- *giga1* is the GIGAswitch/FDDI System name.
- *-s* indicates seek all objects in that group.

Default Gateway Configuration

If a Trap Address is configured for an NMS that is not on a subnet local to the GIGAswitch/FDDI System, a default gateway must be configured so that traps can be delivered to that NMS.

The default gateway can be configured with an SNMP SET on the SNMP object ipRouteTable. The destination IP address should be specified as 0.0.0.0, and the next hop (ipRouteNextHop) should be specified as the IP address of the gateway.

Here is an example using MCC to set a default gateway:

```
use def qualifier by password 08002b20d900
use def entity snmp gsmko11101p021
set ip routingtable 0.0.0.0 ipRouteDest 0.0.0.0
set ip routingtable 0.0.0.0 ipRouteMask 0.0.0.0
set ip routingtable 0.0.0.0 ipRouteNextHop 16.126.64.254
```

Note

You need not configure this parameter if the NMS is on a directly connected subnet, or if the trap address table is not configured.

The following shows sample SNMP commands used to set a trap address.

```
use default qual by password 08002b20d900
use default entity snmp gigaswitch dec ema eauth1
set eauthTrapUserTable 198.55.32.10 -
eauthTrapUserAddr 198.55.32.10
```

Setting Community Name

Setting the community name for the GIGAswitch/FDDI System provides SNMP with primitive security. It is often a required part of many SNMP set commands, and some SNMP get commands, thus only people who know the community name (referred to as a password in DECmcc and POLYCENTER) will be able to perform these SNMP tasks.

If you do not use the correct community name, or do not use the community name where you should in SNMP commands that require it, the agent may respond with an authentication failure trap, or may result in a message such as "no response from entity."

Most NMSs use *public* by default as a community name. The GIGAswitch/FDDI System, by default (upon initial use and management memory reset), has as its community name its lowest hardware address (which is the lowest number in the hardware address range displayed on the clock card label).

The GIGAswitch SNMP agent supports three communities:

- *read-write community*: The GIGAswitch/FDDI System will only accept an SNMP SET request from a user in this community. Get and GetNext requests are also accepted.
- *read-only community*: Get and GetNext requests are accepted from users in this community.
- *trap community*: traps are sent to all hosts in this community.

Thus, a community string is like a password for in-band management because certain requests can only come from certain user communities, and the requests themselves must contain the community string.

Community names are ASCII character strings from 4 to 32 characters long. There are three types of community names: one each for read-only access, read-write access, and use within TRAP PDUs. The default community strings are:

Group	Default
read-write	the lowest hardware address on the GIGAswitch /FDDI System. Enter letters in lowercase, with no dashes, spaces, or colons.
read-only	public
trap	public

Community names are stored in nonvolatile memory, and can be modified by the network manager using in-band management. If you attempt to enter a Set or Get command that requires the community name/password and either do not include it, or enter it incorrectly, you may get a *no response from entity* message (or a similar message) from the NMS.

For each community, there is a table of authorized users. A table may contain up to eight entries. Each entry may represent a single IP address or multiple IP addresses, because each entry consists of an address and a mask. If the mask is all ones, then the entry will match only a single IP address (otherwise, it will match an address set). If you do not enter a mask, it defaults to all ones. For example, the entry (16.22.18.00, 255.255.255.00) will match user addresses 16.22.18.00 through 16.22.18.255.

Note

The default community table allows all addresses access.

An entry whose address portion contains all ones matches all IP addresses (as will a mask of all zeros). By default, this entry is present in each table, therefore, by default, any user who knows the read-write community string can send SET requests to the GIGAswitch/FDDI System. For greater security, first install the addresses of authorized users, then delete the default entry.

Note

Remember to install addresses in the table before deleting the defaults or you will remove your ability to use Set commands.

Read-Only Users

Access control is specified using **eauthReadOnlyUserTable** of the DEC ELAN MIB. This table can contain up to eight entries (IP addresses). It consists of three columns:

1. **eauthReadOnlyUserAddr** (address)
2. **eauthReadOnlyUserMask** (mask)
3. **eauthReadOnlyUserStatus** (status)

Together, the address and mask specify IP addresses that have read access to the GIGAswitch/FDDI System. The status column is used to delete entries; by writing a value of invalid(2), it returns the value of other(1) when read.

The default entry in this table allows read-only access to all IP addresses using the correct community name. Remember to modify the default settings after initially altering this table so that access will be denied where desired.

To enable access only to IP addresses on a given subnet, you can use wildcards. Bits that must be matched are specified in the address field, with corresponding bits in the mask set to 1. Bits that need not be matched (don't-care bits) are set to zero in both the address and the mask.

Read-Write Users

By default, any IP address using the community name of the GIGAswitch/FDDI System's hardware address in lowercase text has read-write access to the device. The **eauthReadWriteCommunity** entity can be used to change the read-write community string, which is a password required for issuing the SET commands. This password is usually a hardware address. In the case of the GIGAswitch/FDDI System, it is the lowest address on the system. This is the lower number in the hardware address range label on the clock card module.

Note

Do not include any dashes, colons, spaces, or other delimiters in your hardware address entry. Simply enter the hexadecimal characters, ensuring all letters are typed in lowercase.

Access control is specified using **eauthReadWriteUserTable**, which can contain up to eight entries. This table contains three columns:

1. **eauthReadWriteUserAddr** (address)
2. **eauthReadWriteUserMask** (mask)
3. **eauthReadWriteUserStatus** (status)

Together, the address and mask specify IP addresses that have read-write access to the device. The status column is used when deleting entries, and returns the value of other(1) when read.

A sample command that would create an entry in the **eauthreadwriteusertable** might look like this:

```
set snmp switch1 dec ema eauth1 eauthreadwriteusertable-
16.126.64.0 eauthreadwriteuseraddr 16.126.64.0, -
eauthreadwriteusermask %xffffff00, by password
08002b20d900
```

The default entry in this table allows read-write access to all IP addresses using the correct community name. Remember to modify the default settings when initially altering this table so that access will be denied where desired. This can be done by setting status to invalid(2).

Warning

Delete the default entry only *after* adding any new entries. If you forget the read-write community name or find you have deleted all defaults and have no read-write access to the GIGAswitch/FDDI System, reset the system to the factory defaults using the OBM terminal.

You cannot read the read-write community name with the read-only community name. Modify read-write access control carefully because modifications can make the GIGAswitch/FDDI System unreachable for SETs if changed incorrectly.

Following is an example of using the read-write community name as a password:

```
show snmp switchname dot1dbridge dot1dbase -
dot1dbasebridgeaddress, by password 08002b2ffd7d
```

Where:

- *switchname* is the name of the GIGAswitch/FDDI System.

- *08002b2ffd7d* is the community name.

The following example shows how to set the read-write community name using DECmcc:

```
set snmp switchname dec ema eauth1
eauthreadwritecommunity -
%xstring, by pass 08002b2ffd7d
```

Where:

- *switchname* is the name of the GIGAswitch/FDDI System.
- *%xstring* is the new community name.
- *08002b2ffd7d* is the old community name.

Traps

Traps are event reports sent to SNMP directors on NMSs by the SNMP agent on the GIGAswitch/FDDI System. Some traps are in place by default. For instance, if someone attempts a SET command without a password or privileged IP address, it sets a trap and sends information about this attempt to all trap addresses. The GIGAswitch/FDDI System supports the following traps:

newRoot
topologyChange
authenticationFailure
warmStart
lineCardFailureTrap

The trap address table is the list of IP addresses (typically NMSs) to which all traps (events) generated by the GIGAswitch/FDDI System will be sent.

You can configure this list by performing a SET operation on the **eauthTrapUserTable**. A maximum of eight entries can be specified.

Modify the default community name for trap messages by writing **eauthTrapCommunity**. By default, there are no entries in the trap table. To receive traps, specify the IP address to send traps to in **eauthTrapUserTable**. The table can hold up to eight entries.

An example of adding a new address (1.2.3.4) to the trap table:

```
set snmp 16.21.16.128 dec ema eauth1 eauthtrapusertable -
1.2.3.4 eauthtrapuseraddr 1.2.3.4, by password -
"08002b2ffd7d"
```

An example of deleting an address from the trap table:

```
set snmp 16.21.16.128 dec ema eauth1 eauthtrapusertable -
1.2.3.4 eauthtrapuserstatus invalid, by password
"08002b2ffd7d"
```

An example of showing all addresses in the trap table:

```
show snmp 16.21.16.128 dec ema eauth1 eauthtrapusertable
* eauthtrapuseraddr
```

Authentication Failure traps are sent by the GIGAswitch/FDDI System only if the variable **snmpEnableAuthenTraps** has a value of enabled(1). This is the default value.

You can assign levels of severity for traps and enable reporting using DECmcc if you want to report them over the network. You need to create a *notify* request for SNMP in the domain in which the entity is registered. This request should contain the event types you want to watch, and upon complete designation it should be enabled (activated). Over-temperature alerts and power supply failures can be flagged this way at remote locations, using NMS traps and alarms.

See the section titled **Bridge Traps** in **Chapter 3** for more information on traps.

3

GIGAswitch/FDDI MIB Objects

This chapter contains information SNMP MIB objects that are useful in managing the GIGAswitch/FDDI System.

Implemented MIB Object Groups

The MIBs used by the GIGAswitch/FDDI System include:

- MIB-II (RFC1213)
- FDDI MIB (RFC1285)
- Bridge MIB (RFC1286)
- DEC Vendor MIB (Vendor MIB Elanext), Version 2.7
- GIGAswitch MIB, Version 1.3

Consult the MIBs directly for complete information.

Table 3–1 MIB Object Implementation

MIB	Object Groups
MIB-II	**system, interfaces, at, ip, icmp, udp, snmp**
FDDI MIB	**snmpFddiSMT, snmpFddiMAC, snmpFddiPORT, snmpFddiATTACHMENT**
Bridge MIB	**dot1dBase, dot1dStp, dot1dTp, dot1dStatic, traps**
GIGAswitch MIB	all
DEC Vendor MIB	**efddi, esystem, ebridge, einterfaces, eauth**

GIGAswitch MIB Object Tree

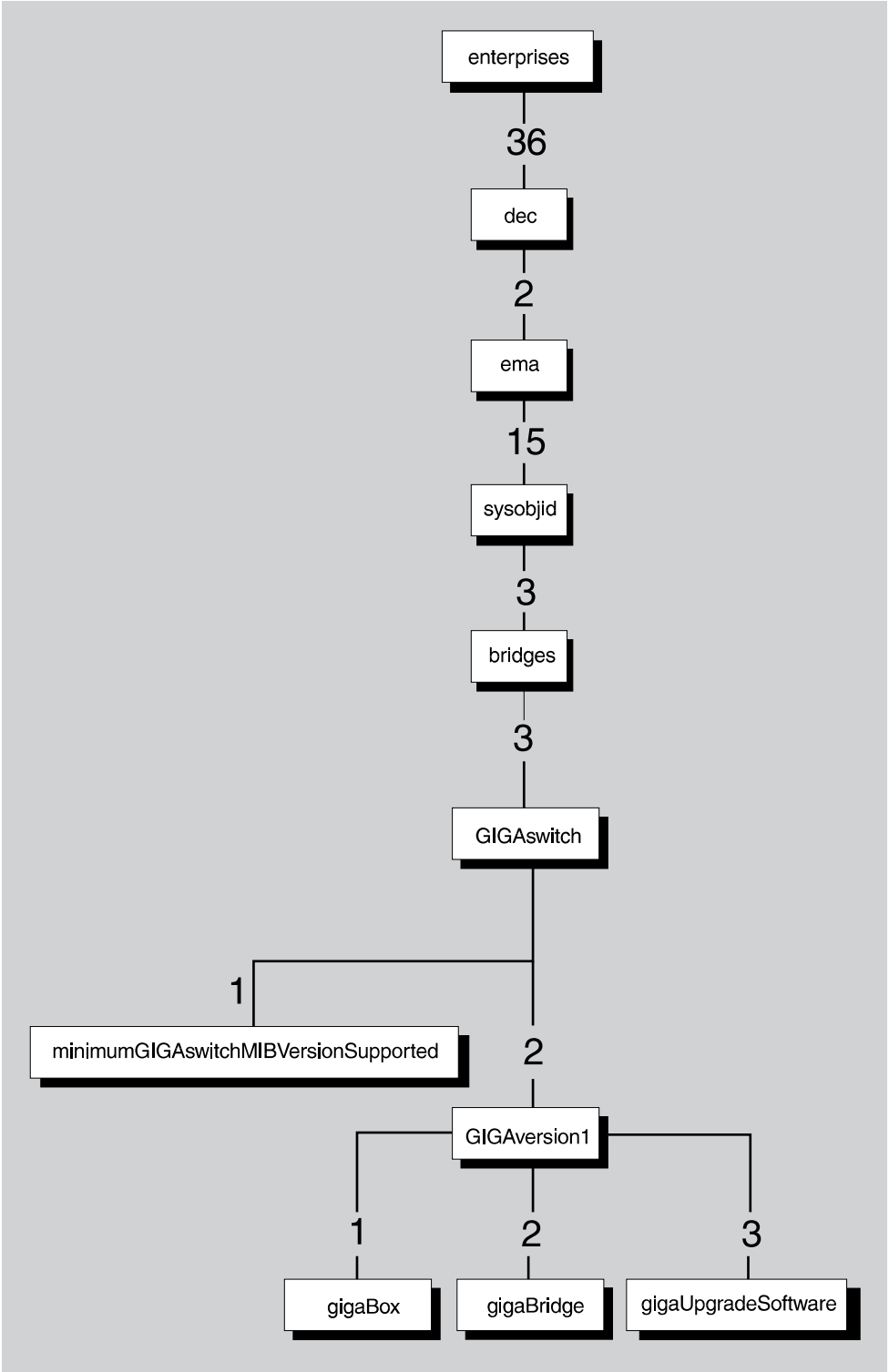
The top of the GIGAswitch MIB object tree is shown in Figure 3–1. This shows you the relative placement of several of the MIB object groups in the GIGAswitch MIB. MIB objects can be referred to by their name, or their number in the MIB tree.

When you specify a MIB object, you need to give its full name or object ID number, or you need to specify its leaf name (which can often be done from a pulldown menu, depending on your NMS).

To access a specific variable within the management information base, you must supply the complete variable, or object name. These object names mirror the hierarchical structure of the information base, and are made up of the sequence of the branches of the MIB tree that lead down to the object.

A MIB object can be entered on an NMS by either typing the complete object name, typing the complete object ID, or using a window-based system to click on each correct sequential tree branch until the desired object is fully specified.

Figure 3–1 GIGAswitch MIB Object Tree



TAY-0246-AF

Important Configuration Management Objects

An extended LAN with bridges operates most efficiently if the root bridge is connected directly to the backbone segment. You can control which bridge becomes the root by setting the object **dot1dStpPriority** to the highest value in the extended LAN. To minimize changes if the root bridge should fail, the backup bridge should be adjacent to the root bridge. Similarly, you can control which bridge becomes the backup bridge by setting the object **dot1dStpPriority** to the second-highest value in the extended LAN. Strongly consider making the GIGAswitch the root bridge, due to its high throughput and availability.

To improve efficiency, you should assign lower path costs to higher-bandwidth ports.

You can set the object **dot1dStpPortPathCost** to bias the bridge during election for root.

You can turn ports on and off by appropriately setting the value of the object **dot1dStpPortEnable** (1 is enabled, 2 is disabled).

You can control how long a node address remains in the forwarding database before it ages out by modifying **dot1dTpAgingTime**.

Table 3–2 lists some vital bridge configuration objects.

Table 3–2 Important Bridge Configuration Objects

Object	Syntax	Description
sysDescr	Display String (255)	Description of entity, with full name and version ID.
sysName	Display String (255)	Domain name of this node.
sysLocation	Display String (255)	Node location, such as "Satellite Equipment Room, 2nd Floor."
sysContact	Display String (255)	Person to contact regarding this node.
dot1dStpPriority	Integer (65535)	Value of the writable portion of the bridge ID.
dot1dStpPortPathCost	Integer (65535)	Path cost addition of this port (added to paths towards the spanning tree root).
dot1dStpPortEnable	integer	port status: 1=enabled, 2=disabled.
dot1dTpAgingTime	Integer	Timeout (in seconds) for aging out learned addresses.

Important Network Monitoring Objects

Table 3–3 lists objects that can help you track bridge performance. Because the GIGAswitch/FDDI System samples every frame on the LAN segments it connects, it can effectively monitor those LANs.

The sum of unicast and multicast packets received (the objects **ifInUcastPkts** plus **ifInNUcastPkts**), and the sum of unicast and multicast packets transmitted (the objects **ifOutUCastPkts** plus **ifOutNUcastPkts**), gives the total traffic on the LAN. The unicast and multicast traffic rates are also useful statistics to monitor individually; sample the counters and the object **sysUpTime**. You can also calculate LAN utilization by sampling the count of total bytes sent and received (the objects **ifOutOctets** plus the objects **ifInOctets**).

You should also monitor the ratio of packets forwarded to packets filtered (the objects **dot1dTpPortInFrames** and **dot1dTpPortInDiscards**). A constant low ratio of frames filtered to frames received indicates poor isolation of traffic (traffic is not being kept local, suggesting possible poor placement of the bridge). You can monitor the number of frames transmitted on a port (the object **dot1dTpPortOutFrames**) to keep an eye on the level of traffic being forwarded on the port (which should be much lower than the frames received).

Counters wrap (return to zero) when they reach their maximum value. Different counters wrap at different rates. Ensure that you sample objects appropriately. Take values at the start and end of a period and compare the difference. Depending on the LAN traffic, some counters may wrap quickly, so adjust your sampling times as necessary.

Table 3–3 Important Bridge Performance Objects

Object	Syntax	Description
rptrOperStatus	Integer	Operational state of bridge: 1 - unknown, 2 - ok, 3 - bridge has failed, 4 - group has failed, 5 - port has failed, 6 - unspecified failure.
ifInUcastPkts	Counter, $0 \rightarrow (2^{32} - 1)$	Number of subnetwork-unicast packets delivered to a higher-layer protocol.
ifInNUcastPkts	Counter, $0 \rightarrow (2^{32} - 1)$	Number of non-unicast packets delivered to a higher-layer protocol.
ifOutUcastPkts	Counter, $0 \rightarrow (2^{32} - 1)$	Number of subnetwork-unicast packets that higher-layer protocols asked to be transmitted to a subnetwork-unicast address.
ifOutNUcastPkts	Counter,$0 \rightarrow (2^{32} - 1)$	Number of subnetwork-unicast packets that higher-layer protocols asked to be transmitted to a non-unicast address.
ifInOctets	Counter, $0 \rightarrow (2^{32} - 1)$	Total octets (bytes) received, including framing characters.
ifOutOctets	Counter, $0 \rightarrow (2^{32} - 1)$	Total octets (bytes) transmitted, including framing characters.
dot1dTpPortInFrames	Counter, $0 \rightarrow (2^{32} - 1)$	Number of frames received by this port from its segment.
dot1dTpPortOutFrames	Counter, $0 \rightarrow (2^{32} - 1)$	Number of frames transmitted by this port to its segment.
dot1dTpPortInDiscards	Counter, $0 \rightarrow (2^{32} - 1)$	Number of fames filtered by the forwarding process.
dot1dStpPortState	disabled, blocking, listening, learning, forwarding, or broken	Status, informs you what state the bridge port is in.

Important Topology Mapping Objects

Table 3–4 lists SNMP objects related to mapping your extended LAN and determining the number and segment address of its component nodes.

Spanning tree data is stored in the **dot1Stp** group of objects, defined in the Bridge MIB. Each bridge knows which one is the root; that information is stored in the object **dot1dStpPortDesignatedRoot**. Bridges immediately adjacent to the root bridge list the root as the designated bridge; that information is stored in the object **dot1dStpPortDesignatedBridge**. The particular port of the root bridge to which they are connected is stored in the object **dot1dStpPortDesignatedPort**. Examine the object **dot1dStpPortState**: the spanning tree is formed by all of the paths connected by bridge ports in the forwarding state (5). You can also identify ports that are not forwarding traffic.

Since the topology can change dynamically, you should monitor these objects periodically. The section titled Bridge Traps in this chapter discusses traps that signal topology changes.

The forwarding database table contains the addresses and port numbers of nodes on a LAN segment serviced by each bridge. Examine the row objects **dot1dTpFdbEntry** for **dot1dTpFdbAddress** and **dot1dTpFdbPort**. Compare the entries from bridge to bridge to determine the exact segment on which each node resides. This also yields the total number of nodes.

The object **dot1dTpFdbStatus** shows whether an address is for the bridge itself (self) or aged out (invalid). Aged addresses are unreliable, because the node may have been removed in the interim. You might want to re-examine this data periodically. Note that the tables can be large, and retrieving them can affect network traffic.

Table 3–4 Important Mapping Objects

Object	Syntax	Description
dot1dStpPort-DesignatedRoot/	Bridge ID 8 octets	Bridge identifier of the root of the spanning tree.
dot1dStpPort-DesignatedBridge	Bridge ID 8 octets	Bridge identifier of the designated bridge in the segment.
dot1dStpPort-DesignatedPort	Octet string 2 octets	Port identifier of the port on the designated bridge for this port's segment.
dot1dStpPortState	Integer	Port's current state as defined by the spanning tree algorithm. Can be: 1(disabled), 2(blocking), 3(listening), 4(learning), 5(forwarding), 6(broken).
dot1dTpFdbTable		
dot1dTpFdbEntry	sequence	Information about a specific unicast MAC address for which the bridge has some forwarding or filtering information.
dot1dTpFdbAddress	MAC address	A unique MAC address.
dot1dTpFdbPort	Integer	Either "0" or the port number
dot1dTpFdbStatus	Integer	Status of the entry: 1(other), 2(invalid), 3(learned), 4(self), 5(management).

Bridge Traps

Table 3–5 lists important bridge SNMP traps.

The GIGAswitch/FDDI System generates a trap on **warmStart**. This trap indicates that the bridge has reinitialized, and that volatile management information has been retained. The trap **newRoot** indicates that the bridge has been elected the root bridge. The trap **topologyChange** indicates that one of the ports on the bridge has changed from the learning state to the forwarding state, or from the forwarding state to the blocking state. See the section titled **Traps** in **Preparing for Network Management Station (NMS) Use** for more information on traps.

To use these traps, configure the trap user table, which is a list of up to eight IP addresses. The object ID of **eauthTrapUserTable** is 1.3.6.1.4.1.36.2.18.1.5.1.2.*n*.

If your NMS is not on the local subnet, set the object esysGatewayAddress or the **ipRouteTable** entry **ipRouteNextHop** so that the trap will reach you. If you set the latter, specify the destination IP address as 0.0.0.0, and the next hop (**ipRouteNextHop**) as the IP address of the gateway.

Table 3–5 Important Bridge Traps

Name	Description
warmStart	The bridge has reinitialized.
newRoot	The bridge is now the root.
topologyChange	The bridge has had a port state transition.

Troubleshooting Bridge Problems

Table 3–6 lists some useful objects for troubleshooting bridge problems.

One possible problem is congestion on a LAN segment. This congestion manifests itself as an excessive number of frames, destined for a given LAN segment, that are discarded because they were queued internally for too long. To document this problem, examine the object **dot1dBasePortDelayExceededDiscards**.

The logical configuration of the LAN (the spanning tree) should stabilize shortly after the network powers up. If it shows signs of instability, this indicates problems with connectivity, or devices going up and down. Examine these objects:

- **dot1dStpTopChanges**—the number of topology changes since power-up.
- **dot1dStpTimeSinceTopologyChange**—when the last change occurred.
- **dot1dStpPortForwardingTransitions**—the number of times a given port went into the forwarding state (should be 0 or 1).

If frames of certain protocols or certain addresses are unexpectedly being filtered, examine the filter table objects to see if filters were incorrectly set up.

Table 3–6 Important Bridge Troubleshooting Objects

Object	Syntax	Description
dot1dBasePort-DelayExceededDiscards	Counter 0-32767	Number of frames discarded by this port due to excessive transit delay through the bridge.
dot1dTpLearned-EntryDiscards	Counter 0-32767	Number of forwarding database entries discarded due to lack of space in the forwarding database.
dot1dStpTopChanges	Counter 0-32767	Number of topology changes detected by this bridge since the agent was last reset or initialized.
dot1dStpTimeSince-TopologyChange	TimeTicks	Time, in hundredths of a second, since the last topology change detected by the bridge.
dot1dStpPort-ForwardTransitions	Counter 0-32767	Number of times this port has gone from a learning state to a forwarding state.

A

GIGAswitch MIB Objects

The following is a listing of the GIGAswitch MIB objects, with abbreviated definitions of each object. This list can be used as an introduction to these objects, or as a quick reference. MIB object IDs are listed in **Appendix B**. A view of the GIGAswitch MIB object tree is shown in **Figure 3–1**.

For the complete MIB, print it from your network management station after copying, or use an NMS tool that allows you to browse or walk through the GIGAswitch MIB.

Note

All objects have *read-write* access, and are *mandatory*, unless otherwise specified.

minimumGIGAswitchMIBVersionSupported

Syntax: integer
Access: read-only
Description: Minimum GIGAswitch System MIB version that is supported.

maximumGIGAswitchMIBVersionSupported

Syntax: integer
Access: read-only
Description: Maximum GIGAswitch System MIB version that is supported.

FilterByReferencedExpression Group Reachability Matrix Objects

ebrNportMatrixNameTable

Syntax: sequence of EbrNportMatrixNameEntry
Access: not accessible
Description: Table for specifying and naming filter matrices.

ebrNportMatrixNameEntry

Syntax: EbrNportMatrxNameEntry
Access: not accessible
Description: A named filter matrix. This entity's sequence includes ebrNportMatrixName, ebrNportMatrixValue, and ebrNportMatrixStatus.

ebrNportMatrixName

Syntax: displaystring (size 32)
Description: A unique filter matrix name.

ebrNportMatrixValue

Syntax: displaystring (size 32)
Description: A matrix is expressed using a shorthand that says what input ports can send packets to what output ports.)

An example of a specification might be: *11:1; 1:3-5; 2:0,3; 4-7,9:4-7,9; 10:*. Semicolons are used to separate expressions. Within each expression, Bridge ports on the left hand side of each colon can send packets to bridge ports on the right hand side. A list of bridge ports on one side of a colon must be separated by commas. Hyphens specify a range of numbers on one side of a colon. If there is no port entry on the right side of the colon, the bridge ports on the left cannot send packets to any bridge port (unless the matrix is combined with some some other matrix in a filter specification, or unless the filter is overridden).

ebrNportMatrixStatus

Syntax: invalid(1) or permanent(2)
Description: Matrices are initially defined as permanent. An invalid entry will remove that matrix from use.

ebrNportMatrixFppnValue

Syntax: Displaystring (size 32)
Description: A matrix is expressed using a shorthand that says what input ports can send packets to what output ports.)

An example of a specification might be: *1.1,2.1-14.2:1.1-14.2; 2.2:5.2,6.1; 10.1:*. The specification is exactly like ebrNportMatrixValue, only using FPPNs instead of BPNs

FilterByReferencedExpression Group Filtering Objects for SAP, SNAP, DA and SA

These groups are similar in structure and definition, so they are combined here for convenience. These filtering objects are related to SAP, SNAP, DA and SA.

ebrNportSapNameTable
ebrNportSnapNameTable
ebrNportDANameTable
ebrNportSANameTable

Syntax: sequence of ebrNportSapNameEntry, ebrNportSnapNameEntry, ebrNportDANameEntry, ebrNportSANameEntry
Access: not-accessible
Description: The filters specified by the designated SAP, SNAP, DA or SA.

ebrNportSapNameEntry
ebrNportSnapNameEntry
ebrNportDANameEntry
ebrNportSANameEntry

Syntax: EbrNportSapNameEntry
Access: not-accessible
Description: A part of a filter for a user-specified address or protocol. The sequence of this object is made up of five other MIB objects: ebrNport*x*Name, ebrNport*x*Name*x*, ebrNport*x*MatrixName, ebrNport*x*NameDisp, and ebrNport*x*NameStatus, where *x* is sap, snap, DA or SA.

ebrNportSapName
ebrNportSnapName
ebrNportDAName
ebrNportSAName

Syntax: displaystring (size 32)
Description: A unique name for part of a filter. This object is the index (or "handle") that lets you refer to (or "grasp") a combination of a specified SAP, SNAP, SA, or DA matrix with the NMS.

ebrNportSapNameSap
ebrNportSnapNameSnap
ebrNportDANameDA
ebrNportSANameSA

Syntax: Octet string (SAP size = 1, SNAP size = 5, DA size = 6, SA size = 6)
Description: the octet representing the given SAP, SNAP, DA or SA. ebrNportSapNameTable entry.

ebrNportSapMatrixName
ebrNportSnapMatrixName
ebrNportDAMatrixName
ebrNportSAMatrixName

Syntax: displaystring (size 32)
Description: an ebrNportMatrixName. Multiple matrices for the same SAP, SNAP, DA or SA are logically combined.

ebrNportSapNameDisp
ebrNportSnapNameDisp
ebrNportDANameDisp
ebrNportSANameDisp

Syntax: integer - filter(1), alwaysFilter(2), alwaysForward(3)
Description: The order of precedence of these dispositions from most powerful to least powerful is alwaysFilter, alwaysForward, filter. Multiple matrixes that apply to the same packet are logically OR'ed together. Knowing the order of precedence, and the logical OR operation, you can discern what will actually happen to given packets, and you can design your filters accordingly.

ebrNportSANameDisp has two additional integer values available: lockdown(4) and lockdownportmask(5). Lockdown(4) means that frames sourced from this address are ONLY forwarded if received on the same port as ebrNportPortNum. Lockdownportmask(5) is similar to lockdown(4), but the frame must also be destined to a permitted port as specified by the filter matrix.

ebrNportSapNameStatus
ebrNportSnapNameStatus
ebrNportDANameStatus
ebrNportSANameStatus

Syntax: integer - invalid(1), permanent(2)
Description: invalid(1) removes a filter from use. permanent(2) indicates the filter is preserved through a bridge reset.

FilterByReferencedExpression Group Reachability Row Objects

ebrNportMatrixNameRowTable

Syntax: sequence of EbrNportMatrixNameRowEntry
Access: not-accessible
Description: another view of ebrNportMatrixNameTable.

ebrNportMatrixNameRowEntry

Syntax: EbrNportMatrixNameRowEntry
Access: not-accessible
Description: A row of a named filter matrix. This sequence consists of ebrNportmatrixName, ebrNportMatrixReceivePort, ebrNportMatrixAllowedToGoTo, ebrNportMatrixNameRowStatus.

ebrNportMatrixName

Syntax: display string (size 32)
Description: a unique filter matrix name - same as ebrNportMatrixName.

ebrNportMatrixReceivePort

Syntax: integer
Description: a bridge port number. A zero entry indicates all rows not previously mentioned.

ebrNportMatrixAllowedToGoTo

Syntax: octet string (size 5)
Description: The set of ports that can be allowable destinations. Each octet specifies a set of 8 ports, with 1 meaning the port is included, 0 meaning it is excluded.

ebrNportMatrixNameRowStatus

Syntax: integer: invalid(1) or permanent(2)
Description: invalid means the matrix is unusable.

ebrNportMatrixFppnRowTable

Syntax: sequence of EbrNportMatrixFppnRowEntry
Access: not-accessible
Description: This table is yet another view of ebrNportMatrixNameTable. The tables side-effect one another.

ebrNportMatrixFppnRowEntry

Syntax: EbrNportMatrixFppnRowEntry
Access: not-accessible
Description: A particular row of a particular named filter matrix. This object is made up of four other MIB objects: ebrNportmatrixname and ebrNportMatrixFppnReceivePort, ebrNportMatrixFppnAllowedToGoTo, and ebrNportMatrixFppnRowStatus.

ebrNportmatrixname

Syntax: DisplayString (SIZE (32)
Description: A unique filter matrix name. Same as ebrNportMatrixName.

ebrNportMatrixFpppnReceivePort

Syntax: DisplayString
Description: Port designated by fppn (slot.port).

ebrNportMatrixFpppnAllowedToGoTo

Syntax: octet string (size (5))
Description: The set of ports to which frames received from a specific port are allowed to be forwarded. Specified in octets - the first is ports 1-8, second indicates ports 9-16. A 1 bit indicates the port is included.

ebrNportMatrixFpppnRowStatus

Syntax: integer: invalid(1) or permanent(2)
Description: Permanent(2) indicates the row can be used in filter specifications. Invalid(1) indicates it is not usable.

ebrNportDefaultMatrixValue

Syntax: display string (size 32)
Description: A matrix is expressed using a shorthand that says what input ports can send packets to what output ports.)

An example of a specification might be: *11:1; 1:3-5; 2:0,3; 4-7,9:4-7,9; 10:*. Semicolons are used to separate expressions. Within each expression, Bridge ports on the left hand side of each colon can send packets to bridge ports on the right hand side. A list of bridge ports on one side of a colon must be separated by commas. Hyphens specify a range of numbers on one side of a colon. If there is no port entry on the right side of the colon, the bridge ports on the left cannot send packets to any bridge port (unless the matrix is combined with some other matrix in a filter specification, or unless the filter is overridden).

ebrNportDefaultMatrixFppnValue

Syntax: display string (size 32)
Description: A matrix is expressed using a shorthand that says what input ports can send packets to what output ports.)

An example of a specification might be: *1.1,2.1:1.2; 2.2:5.1,6.1;10.2:*. Semicolons are used to separate expressions. Within each expression, Bridge ports on the left hand side of each colon can send packets to bridge ports on the right hand side. A list of fppns on one side of a colon must be separated by commas. Hyphens specify a range of numbers on one side of a colon. If there is no port entry on the right side of the colon, the bridge ports on the left cannot send packets to any bridge port (unless the matrix is combined with some

other matrix in a filter specification, or unless the filter is overridden).

ebrNportManualFilter

Syntax: display string (size 32)
Description: the list of bridge ports in manual mode (manual mode cancels learning on that port). Bridge ports can be specified the same way as ebrNportDefaultMatrixValue.

ebrNportFppnManualFilter

Syntax: display string (size 32)
Description: the list of front panel port numbers (fppns) in manual mode (manual mode cancels learning on that port). Ports can be specified the same way as ebrNportDefaultMatrixValue.

FilterByBitmapValue DA Filtering Objects

ebrNportStaticDATable

Syntax: sequence of EbrNportStaticDAEntry
Access: not-accessible
Description: This table contains management-specified destination address filtering information about unicast and multicast addresses for N-port bridges.

ebrNportStaticDAEntry

Syntax: ebrNportStaticDAEntry
Access: not-accessible
Description: Specific MAC address data for the bridge's use in management-specified forwarding and/or filtering. This object is made up of a sequence of ebrNportDAAddress, ebrNportDAReceivePort, and ebrNportDAAllowedToGoTo.

ebrNportDAAddress

Syntax: octet string (size 6)
Description: The destination MAC address to which this entry's filtering information applies.

ebrNportDAReceivePort

Syntax: integer
Description: The port from which a frame must be received to use the corresponding ebrNportDAAllowedToGoTo field. Zero indicates this entry is the default for all ports without other applicable entries.

ebrNportDAAllowedToGoTo

Syntax: octet string (size 5)
Access:
Status:
Description: When frames are received from a specific port, and destined to the address specified by ebrNportDAAddress, this object is the set of ports to which they are allowed to be forwarded.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is included in the set of ports; the port is not included if its bit has a value of '0.' Filter matrix rows can be initialized or overwritten, but not deleted.

ebrNportStaticDAFilterCharacteristicsTable

Syntax: sequence of EbrNportStaticDAFilterCharacteristicsEntry

Access: not-accessible
Description: A table that contains information about how static destination address filters (for unicast and multicast addresses) are used within N-port bridges.

ebrNportStaticDAFilterCharacteristicsEntry

Syntax: ebrNportStaticDAFilterCharacteristicsEntry
Access: not-accessible
Description: Information about a specific static MAC destination address filter's use. This object is a sequence of three other objects: ebrNportDestinationAddress, ebrNportDADisp, and ebrNportDAStatus.

ebrNportDestinationAddress

Syntax: octet string (size 6)
Description: The destination MAC Address in a frame to which this entry's information applies. The value of this object is the same s ebrNportDAAddress.

ebrNportDADisp

Syntax: integer: filter(1), alwaysFilter(2), alwaysForward(3)
Description: This object specifies how to interpret the ebrNportPortNum and ebrNportDAAllowedToGoTo objects. The order of precedence of these dispositions from most powerful to least powerful is alwaysFilter, alwaysForward, filter. Multiple matrixes that apply to the same packet are logically OR'ed together. Knowing the order of precedence, and the logical OR operation, you can discern what will actually happen to given packets, and you can design your filters accordingly.

ebrNportDAStatus

Syntax: integer - invalid(2), permanent(3)
Description: options include other(1), invalid(2) - removes the entry, permanent(3) - entry is preserved across a bridge reset,

Service Class Objects

ebrNportDASvcTable

Syntax: sequence of EbrNport DASvcEntry
Access: not-accessible
Description: A table that contains resource information about unicast and multicast destination addresses.

ebrNportDASvcEntry

Syntax: EbrNportDASvcEntry
Access: not-accessible
Description: Resource information about a specific MAC destination address. This object consists of three other objects: EbrNportSvcAddress, ebrNportSvc, and ebrNportSvcStatus.

ebrNportSvcAddress

Syntax: octet string (size 6)
Description: The destination MAC address in a frame to which this entry's resource information applies.

ebrNportSvc

Syntax: integer
Description: The service class used to allocate resources and implement service policy within the bridge. The service class associated with a frame is a function of either the frame's destination address or the frame's protocol. The default value for this field is 0. This value is used to assign the single path service class in the absence of a service class associated with the frame's protocol. There are 16 possible service classes, 12-15 are commonly used. The ebrNportSvc allows 32 values to be specified. The specification of a value greater than 15 is used to connote that the service class (ebrNportSvc-16), is to be associated with the frame and that this service class has precedence over a service class associated with a frame's protocol.

ebrNportSvcStatus

Syntax: integer: invalid(1), permanent(2)
Description: invalid(1) removes an entry from use. permanent(2) indicates the entry is preserved through a bridge reset.

Port Number Objects

ebrNportPortNumTable

Syntax: sequence of EbrNportPortNumEntry
Access: not-accessible
Description: A table that contains static (that is, configured by network management) information about the bridge port on which unicast or multicast addresses reside.

ebrNportPortNumEntry

Syntax: ebrNportPortNumEntry
Access: not-accessible
Description: Information about a specific MAC address. This object is a sequence of three other objects: ebrNportPortNumAddress, ebrNportPortNum, and ebrNportPortNumStatus.

ebrNportPortNumAddress

Syntax: octet string (size 6)
Description: The MAC address to which this entry's information applies.

ebrNportPortNum

Syntax: integer
Description: The port number to which this address will be translated (to which frames destined for the address will be forwarded). If zero, no port number is specified. If greater than 127, this value is a giganet destination address instead of a bridge port number.

ebrNportPortNumStatus

Syntax: integer: invalid(1), permanent(2)
Description: invalid(1) removes a filter from use. permanent(2) indicates the filter is preserved through a bridge reset.

ebrNportFppnPortNum

Syntax: DisplayString
Description: The front panel port number to which this address will be translated (to which frames destined for the address will be forwarded). You may not specify both ebrNportPortNum and ebrNportFppnPortNum at the same time to set an entry in this table.

FilterByBitmapValue SA Filtering Objects

ebrNportStaticSATable

Syntax: sequence of EbrNportStaticSAEntry
Access: not-accessible
Description: A table that contains static (management-specified) source address filtering information about unicast and multicast addresses.

ebrNportStaticSAEntry

Syntax: EbrNportStaticSAEntry
Access: not-accessible
Description: Information about a specific MAC address for which the bridge has some static forwarding and/or filtering settings. This object is a sequence of three other objects: ebrNportSAAddress, ebrNportSAReceivePort, and ebrNportSAAllowedToGoTo.

ebrNportSAAddress

Syntax: octet string (size 6)
Description: The source MAC address to which this entry's filtering information applies.

ebrNportSAReceivePort

Syntax: integer
Description: The port from which a frame must be received to use the corresponding ebrNportSAAllowedToGoTo field. Zero indicates this entry is the default for all ports without other applicable entries.

ebrNportSAAllowedToGoTo

Syntax: octet string (size 5)
Description: When frames are received from a specific port, and come from an address specified by ebrNportSAAddress, this object is the set of ports to which they are allowed to be forwarded.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is included in the set of ports; the port is not included if its bit has a value of '0'. Filter matrix rows can be initialized or overwritten, but not deleted.

ebrNportStaticSAFilterCharacteristicsTable

Syntax: sequence of EbrNportStaticSAFilterCharacteristicsEntry
Access: not-accessible

Description: A table that contains information about how static source address filters (for unicast and multicast addresses) are used within N-port bridges.

ebrNportStaticSAFilterCharacteristicsEntry

Syntax: ebrNportStaticSAFilterCharacteristicsEntry
Access: not-accessible
Description: Information about a specific static MAC source address filter's use. This object is a sequence of three other objects: ebrNportSourceAddress, ebrNportSADisp, and ebrNportSAStatus.

ebrNportSourceAddress

Syntax: octet string (size 6)
Description: The source MAC address in a frame to which this entry's information applies. The value of this object is the same as ebrNportSAAddress.

ebrNportSADisp

Syntax: integer: filter(1), alwaysFilter(2), alwaysForward(3), lockdown(4), lockdownportmask(5)
Description: This object specifies how to interpret the ebrNportPortNum and ebrNportSAAllowedToGoTo objects. The order of precedence of these dispositions from most powerful to least powerful is alwaysFilter, alwaysForward, filter. Multiple matrixes that apply to the same packet are logically OR'ed together. Knowing the order of precedence, and the logical OR operation, you can discern what will actually happen to given packets, and you can design your filters accordingly.

ebrNportSANameDisp has two additional integer values available: lockdown(4) and lockdownportmask(5). Lockdown(4) means that frames sourced from this address are ONLY forwarded if received on the same port as ebrNportPortNum. Lockdownportmask(5) is similar to lockdown(4), but the frame must also be destined to a permitted port as specified by the filter matrix.

ebrNportSAStatus

Syntax: integer - invalid(2), permanent(3)
Description: options include other(1), invalid(2) - removes the entry, permanent(3) - entry is preserved across a bridge reset,

FilterByBitmapValue SAP Filtering Objects

ebrNportSapProtoTable

Syntax: sequence of EbrNportSapProtoEntry
Access: not-accessible
Description: A table that contains filtering information about 802.2 SAPs in the DSAP field.

ebrNportSapProtoEntry

Syntax: EbrMultiSapProtoEntry
Access: not-accessible
Description: A table that contains filtering information about 802.2 SAPs in the DSAP field. This object is a sequence of three other objects: ebrNportSapValue, ebrNportSapReceivePort, and ebrNportSapAllowedToGoTo.

ebrNportSapValue

Syntax: octet string (size 1)
Description: The 802.2 DSAP in a frame to which this entry's filtering information applies.

ebrNportSapReceivePort

Syntax: integer
Description: The port from which a frame must be received to use the corresponding ebrNportSapAllowedToGoTo field. Zero indicates this entry is the default for all ports without other applicable entries.

ebrNportSapAllowedToGoTo

Syntax: octet string (size 5)
Description: When frames are received from a specific port, and come from an address specified by ebrNportSapAddress, this object is the set of ports to which they are allowed to be forwarded.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is included in the set of ports; the port is not included if its bit has a value of '0'. Filter matrix rows can be initialized or overwritten, but not deleted.

ebrNportSapSvcTable

Syntax: sequence of EbrNportSapSvcEntry
Access: not-accessible
Description: A table that contains resource information (for example, queuing) about 802.2 SAPs in the DSAP field for N-port bridges.

ebrNportSapSvcEntry

Syntax: EbrNportSapSvcEntry
Access: not-accessible
Description: Resource information about an 802.2 SAP in the DSAP field. This object consists of three other objects: EbrNportSapSvcSapValue, ebrNportSapSvc, and ebrNportSapSvcStatus.

ebrNportSapSvcSapValue

Syntax: octet string (size 1)
Description: The 802.2 DSAP in a frame to which this entry's resource information applies.

ebrNportSapSvc

Syntax: integer
Description: The service class used to allocate resources and implement service policy within the bridge. The service class associated with a frame is a function of either the frame's destination address or the frame's protocol. The default value for this field is 0. This value is used to assign the single path service class in the absence of a service class associated with the frame's protocol. There are 16 possible service classes, 12-15 are commonly used. The ebrNportSvc allows 32 values to be specified. The specification of a value greater than 15 is used to connote that the service class (ebrNportSvc-16), is to be associated with the frame and that this service class has precedence over a service class associated with a frame's protocol.

ebrNportSapSvcStatus

Syntax: integer: invalid(1), permanent(2)
Description: invalid(1) removes an entry from use. permanent(2) indicates the entry is preserved through a bridge reset.

ebrNportStaticSapFilterCharacteristicsTable

Syntax: sequence of EbrNportStaticSapFilterCharacteristicsEntry
Access: not-accessible
Description: A table that contains information about how 802.2 DSAP filters are used.

ebrNportStaticSapFilterCharacteristicsEntry

Syntax: ebrNportStaticSapFilterCharacteristicsEntry
Access: not-accessible
Description: Information about a specific 802.2 DSAP filter's use. This object is a sequence of three other objects: ebrNportSapFilterCharacteristcsSapValue, ebrNportSapDisp, and ebrNportSapStatus.

ebrNportSapFilterCharacteristicsSapValue

Syntax: octet string (size 1)
Description: The 802.2 DSAP in a frame to which this entry's information applies. The value of this object is the same s ebrNportSapValue.

ebrNportSapDisp

Syntax: integer: filter(1), alwaysFilter(2), alwaysForward(3)
Description: This object specifies how to interpret the ebrNportSapAllowedToGoTo object for the specified SAP. The order of precedence of these dispositions from most powerful to least powerful is alwaysFilter, alwaysForward, filter. Multiple matrixes that apply to the same packet are logically OR'ed together. Knowing the order of precedence, and the logical OR operation, you can discern what will actually happen to given packets, and you can design your filters accordingly.

ebrNportSapStatus

Syntax: integer - invalid(2), permanent(3)
Description: options include other(1), invalid(2) - removes the entry, permanent(3) - entry is preserved across a bridge reset.

FilterByBitmapValue SNAP Filtering Objects

ebrNportSnapProtoTable

Syntax: sequence of EbrNportSnapProtoEntry
Access: not-accessible
Description: A table that contains filtering information about 5-byte SNAP Protocol IDs (PIDs).

ebrNportSnapProtoEntry

Syntax: EbrMultiSapProtoEntry
Access: not-accessible
Description: A table that contains filtering information about 5-byte SNAP Protocol IDs (PIDs). This object is a sequence of three other objects: ebrNportSnapValue, ebrNportSnapReceivePort, and ebrNportSnapAllowedToGoTo.

ebrNportSnapValue

Syntax: octet string (size 5)
Description: The SNAP Protocol ID (PID) in an 802 frame to which this entry's filtering information applies.

ebrNportSnapReceivePort

Syntax: integer
Description: The port from which a frame must be received to use the corresponding ebrNportSnapAllowedToGoTo field. Zero indicates this entry is the default for all ports without other applicable entries.

ebrNportSnapAllowedToGoTo

Syntax: octet string (size 5)
Description: The set of ports to which frames received from a specific port and containing the SNAP PID in an 802 frame specified by ebrNportSnapValue are allowed to be forwarded.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is included in the set of ports; the port is not included if its bit has a value of '0'. Filter matrix rows can be initialized or overwritten, but not deleted.

ebrNportSnapSvcTable

Syntax: sequence of EbrNportSnapSvcEntry
Access: not-accessible
Description: A table that contains resource information (queuing) about 5-byte SNAP Protocol IDs (PIDs).

ebrNportSnapSvcEntry

Syntax: EbrNportSnapSvcEntry
Access: not-accessible
Description: Resource information about a 5-byte SNAP Protocol ID (PID). This object consists of three other objects: EbrNportSnapSvcSnapValue, ebrNportSnapSvc, and ebrNportSnapSvcStatus.

ebrNportSnapSvcSnapValue

Syntax: octet string (size 1)
Description: The SNAP Protocol ID (PID) in a frame to which this entry's resource information applies.

ebrNportSnapSvc

Syntax: integer
Description: The service class used to allocate resources and implement service policy within the bridge. The service class associated with a frame is a function of either the frame's destination address or the frame's protocol. The default value for this field is 0. This value is used to assign the single path service class in the absence of a service class associated with the frame's protocol. There are 16 possible service classes, 12-15 are commonly used. The ebrNportSvc allows 32 values to be specified. The specification of a value greater than 15 is used to connote that the service class (ebrNportSvc-16), is to be associated with the frame and that this service class has precedence over a service class associated with a frame's protocol.

ebrNportSnapSvcStatus

Syntax: integer: invalid(1), permanent(2)
Description: invalid(1) removes a specified service class from use, permanent(2) indicates the entry is preserved through a bridge reset.

ebrNportSnapFilterCharacteristicsTable

Syntax: sequence of EbrNportSnapFilterCharacteristicsEntry
Access: not-accessible
Description: A table that contains information about how SNAP Protocol ID (PID) filters are used.

ebrNportSnapFilterCharacteristicsEntry

Syntax: ebrNportSnapFilterCharacteristicsEntry
Access: not-accessible
Description: 802.2 SNAP Protocol ID (PID) filter use. This object consists of three other objects: ebrNportSnapFilterCharacteristcsSnapValue, ebrNportSnapDisp, and ebrNportSnapStatus.

ebrNportSnapFilterCharacteristicsSnapValue

Syntax: octet string (size 1)

Description: The SNAP Protocol ID (PID) in a frame to which this entry's information applies. The value of this object is the same as ebrNportSnapValue.

ebrNportSnapDisp

Syntax: integer: filter(1), alwaysFilter(2), alwaysForward(3)
Description: This object specifies how to interpret the ebrNportSnapAllowedToGoTo object for the specified SNAP PID. The order of precedence of these dispositions from most powerful to least powerful is alwaysFilter, alwaysForward, filter. Multiple matrixes that apply to the same packet are logically OR'ed together. Knowing the order of precedence, and the logical OR operation, you can discern what will actually happen to given packets, and you can design your filters accordingly.

ebrNportSnapStatus

Syntax: integer - invalid(2), permanent(3)
Description: options include other(1), invalid(2) - removes the entry, permanent(3) - entry is preserved across a bridge reset.

Default Filter Matrix Objects

ebrNportSwTable

Syntax: sequence of EbrNportSwEntry
Access: not-accessible
Description: This table describes the default filter matrix.

ebrNportSwEntry

Syntax: EbrNportSwEntry
Access: not-accessible
Description: The entry describes a row of the default filter matrix. This object is comprised of two other objects: ebrNportSwReceivePort and ebrNportSwAllowedToGoTo.

ebrNportSwReceivePort

Syntax: integer
Description: The bridge port on which a frame must be received to use the corresponding ebrNportSwAllowedToGoTo field. A zero indicates this will be the default for all ports without any management specified filter information.

ebrNportSwAllowedToGoTo

Syntax: octet string (size 5)
Description:

This object specifies the default forwarding disposition after all DA, SA, and protocol filters have been applied. If no filter applies, the ebrNportSwAllowedToGoTo object indicates this information.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is included in the set of ports; the port is not included if its bit has a value of '0'.

ebrNportSwManualFilter

Syntax: octet string (size 5)
Description: A switch that controls address filtering. Specifying a one in a bit position says that the port is in manual mode.

Each octet of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying port 9 through 16 and so on. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. If a bit has a value of '1', then the corresponding port is

included in the set of ports; the port is not included if its bit has a value of '0'.

Manual mode means that the bridge purges the learned entries for that port from its forwarding database, stops its learning process on that port, and forwards to that port only frames with destination and source addresses that have been specified via management.

cutThrough Objects

cutThroughTable

Syntax: sequence of CutThroughEntry
Access: not-accessible
Description: This table controls the enabling of some hardware performance enhancements that avoid store-and-forward delays during packet forwarding.

cutThroughEntry

Syntax: CutThroughEntry
Access: not-accessible
Description: Information about the enabling of some hardware performance enhancements that avoid store-and-forward delays for a specific bridge port. This object is a sequence of three other objects: cutThroughBridgePort, cutThroughInbound, and cutThroughOutbound.

cutThroughBridgePort

Syntax: integer
Description: The dot1dBasePort bridge port number for this table entry.

cutThroughInbound

Syntax: integer: true(1), false(2)
Description: If true, some hardware performance enhancements are enabled to avoid store-and-forward delays when packets enter the GIGAswitch. If false, the whole packet is buffered on the linecard at which the packet enters the GIGAswitch before it is sent to the linecard at which the packet leaves the GIGAswitch. Cut-through is normally enabled.

cutThroughOutbound

Syntax: integer: true(1), false(2)
Description: If true, some hardware performance enhancements are enabled to avoid store-and-forward delays when packets leave the GIGAswitch. If false, the whole packet is buffered on the linecard at which the packet leaves the GIGAswitch before it is sent out of the GIGAswitch. Cut-through is normally enabled.

cutThroughFppnTable

Syntax: sequence of CutThroughFppnEntry
Access: not-accessible
Description: This table controls the enabling of some hardware performance enhancements that avoid store-and-forward delays during packet forwarding.

cutThroughFppnEntry

Syntax: CutThroughFppnEntry
Access: not-accessible
Description: Information about the enabling of some hardware performance enhancements that avoid store-and-forward delays for a specific bridge port. This object is a sequence of three other objects: cutThroughFppnBridgePort, cutThroughFppnInbound, and cutThroughFppnOutbound.

cutThroughFppnBridgePort

Syntax: DisplayString
Description: The dot1dBasePort bridge port number for this table entry.

cutThroughFppnInbound

Syntax: integer: true(1), false(2)
Description: If true, some hardware performance enhancements are enabled to avoid store-and-forward delays when packets enter the GIGAswitch. If false, the whole packet is buffered on the linecard at which the packet enters the GIGAswitch before it is sent to the linecard at which the packet leaves the GIGAswitch. Cut-through is normally enabled.

cutThroughFppnOutbound

Syntax: integer: true(1), false(2)
Description: If true, some hardware performance enhancements are enabled to avoid store-and-forward delays when packets leave the GIGAswitch. If false, the whole packet is buffered on the linecard at which the packet leaves the GIGAswitch before it is sent out of the GIGAswitch. Cut-through is normally enabled.

Gigabox Objects

mgmtMemoryAvail

Subgroup: clockCard
Syntax: integer
Description: The number of unused bytes of management memory available on the clock card. The management memory is used for storing management parameters.

mgmtMemoryAction

Subgroup: clockCard
Syntax: integer: other(1), rewrite(2), rewriting(3)
Description: This object, when read, returns a value of other(1) unless it is in the middle of rewriting the management memory, when it will return rewriting(3). The state will revert to other(1) when the action is completed.

Setting this variable to rewrite(2) will cause the SCP to attempt to rewrite the management Memory on the clock card (an action which can increase available space by removing duplicate entries).

pscStatus

Subgroup: psc (power system controller)
Syntax: integer: notpresent(1), okay(2), fault(3)
Access: read-only
Description: The status of the power system controller (PSC).

pscFwRev

Subgroup: psc (power system controller)
Syntax: DisplayString
Access: read-only
Description: The firmware revision of the power system controller (PSC).

pscHwRev

Subgroup: psc (power system controller)
Syntax: DisplayString
Access: read-only
Status:
Description: The hardware revision of the power system controller.

keyswitchPosition

Subgroup: psc (power system controller)
Syntax: integer: fault(1), secure(2), local(3), remote(4), worldaccess(5)
Access: read-only

Description: Indicates the position of the keyswitch. The keyswitch position determines the type of access allowed to network management and to out-of-band management.

For SNMP access, secure means no SNMP access. Local means read-only SNMP access. Other values allow read-write SNMP access. All access is still subject to the community string, IP address, and privileged port restrictions.

pscFwImageStatus

Subgroup: psc (power system controller)
Syntax: integer: okay(1), downloadrequired(2)
Access: read-only
Description: The status of the PSC firmware image. If the value of this object is downloadrequired(2), the manager should download a new firmware image to the PSC, using objects in the gigaUpgradeSoftware group.

pscBackplaneStatus

Subgroup: psc (power system controller)
Syntax: integer: okay(1), fault(2)
Access: read-only
Description: Status of the GIGAswitch backplane.

cabinetTemperature

Subgroup: psc (power system controller)
Syntax: integer: normal(1), high(2), excessivelyhigh(3), low(4), excessivelylow(5)
Access: read-only
Description: The cabinet temperature.

temperatureWarning

Subgroup: psc (power system controller)
Syntax: integer: heedwarning(1), ignorewarning(2)
Description: This variable determines how the SCP responds to a cabinet temperature that is too high or too low. If the value is heedwarning(1), the system will be shut down when the temperature goes outside the designed limits. If the value is ignorewarning(2), the system will continue to operate (despite possible risk to the system). This object has a default value of heedwarning(1).

rightPowerStatus

Subgroup: powerSupply
Syntax: integer: notpresent(1), okay(2), fault(3)
Access: read-only
Description: The status of the right power unit.

rightPowerInputSource

Subgroup: powerSupply
Syntax: integer: ACline(1), DC48V(2), none(3)
Access: read-only

Description: The input source of the right power unit. A value of ACline(1) is returned if the power supply is connected to an AC line. A value of DC48V(2) is returned if the power supply is connected to a 48 V dc (telecommunications) line. A value of none(3) is returned if the power unit is not present or not powered on.

rightPowerOutputPower

Subgroup: powerSupply
Syntax: integer
Access: read-only
Description: Output power of right power unit, in watts.

leftPowerStatus

Subgroup: powerSupply
Syntax: integer: notpresent(1), okay(2), fault(3)
Access: read-only
Description: The status of the left power unit.

leftPowerInputSource

Subgroup: powerSupply
Syntax: integer: ACline(1), DC48V(2), none(3)
Access: read-only
Description: The input source of the left power unit. A value of ACline(1) is returned if the power supply is connected to an AC line. A value of DC48V(2) is returned if the power supply is connected to a 48 V dc (telecommunications) line. A value of none(3) is returned if the power unit is not present or not powered on.

leftPowerOutputPower

Subgroup: powerSupply
Syntax: integer
Access: read-only
Description: Output power of left power unit, in watts.

slotNumber

Subgroup: slot
Syntax: integer
Access: read-only
Description: The number of slots in the box.

scpSlot

Subgroup: slot
Syntax: integer
Access: read-only
Description: Identifies the slot the switch control processor (SCP) occupies. A value of 0 will be returned if an SCP has not been chosen.

slotTable

Subgroup: slot

Syntax: sequence of SlotEntry
Access: not-accessible
Description: Describes the current configuration of the box.

slotEntry

Subgroup: slot
Syntax: SlotEntry
Access: not-accessible
Description: The description and status of the card plugged into the slot. This sequence consists of five other objects: slotIndex, slotCardStatus, slotCardType, slotCardHwRev, and slotCardFwRev.

slotIndex

Subgroup: slot
Syntax: integer
Access: read-only
Description: The slot number. Varies from 1 to slotNumber.

slotCardStatus

Subgroup: slot
Syntax: integer: notpresent(1), powerDown(2), powerUp(3), powerDownThenUp(4), fault(5)
Description: The status of the card. Notpresent(1) indicates an empty slot. If the value is read as notpresent(1) and a card resides in the slot, then the entire entry should be considered invalid. Reading a value of powerDown(2) indicates a card is present but not poweredUp. Reading a value of powerUp(3) indicates a card is present and poweredUp. PowerDownThenUp(4) is used during firmware upgrades for the FGL.

Writing a value of powerDown(2) will cause the SCP to power down the card. Writing a value of powerUp(3) will cause the SCP to power up the card. The clock and crossbar cards cannot be powered up or down through this object.

slotCardType

Subgroup: slot
Syntax: integer: other(1), fgl2(2), CBS36(3), switchengine(4), clockcard(5)
Access: read-only
Description: The type of the card.

slotCardHwRev

Subgroup: slot
Syntax: DisplayString
Access: read-only
Description: The hardware revision number of the card.

slotCardFwRev

Subgroup: slot

Syntax: DisplayString
Access: read-only
Description: The firmware revision number of the card.

fanSpeed

Subgroup: fan
Syntax: integer: maximum(1), normal(2)
Description: The speed of the fans. Maximum(1) provides maximum cooling. Normal(2) causes fan speed to be under temperature control.

rightFanStatus

Subgroup: fan
Syntax: integer: notpresent(1), okay(2), fault(3)
Access: read-only
Description: The status of the right fan tray.

leftFanStatus

Subgroup: fan
Syntax: integer: notpresent(1), okay(2), fault(3)
Access: read-only
Description: The status of the left fan tray.

fppnTable

Subgroup: fppn
Syntax: sequence of FppnEntry
Access: not-accessible
Description: A list of front panel port number to MIB-II interface number mappings.

fppnEntry

Subgroup: fppn
Syntax: FppnEntry
Access: not-accessible
Description: Assigns the MIB-II interface ifIndex for a physical datalink (cable) connection to the box. This object consists of four other objects: fppnSlotNumber, fppnPortOfThatSlot, fppnIfIndex, and fppnBridgePortNumber.

fppnSlotNumber

Subgroup: fppn
Syntax: integer
Access: read-only
Description: The slot number of the module (1-14).

fppnPortOfThatSlot

Subgroup: fppn
Syntax: integer
Access: read-only
Description: The media connection to the module.

fppnIfIndex

Subgroup: fppn
Syntax: integer
Access: read-only
Description: The assigned MIB-II ifIndex.

fppnBridgePortNumber

Subgroup: fppn
Syntax: integer
Access: read-only
Description: The assigned bridge MIB dot1dBasePort.

floodUnknownUnicastRate

Subgroup: gigaBridge
Syntax: integer
Description: The maximum bytes-per-second bandwidth of packets multicast due to unknown destination address.

floodMulticastRate

Subgroup: gigaBridge
Syntax: integer
Description: The maximum bytes-per-second bandwidth of packets multicast because the destination address is a multicast address.

floodTable

Subgroup: gigaBridge
Syntax: sequence of FloodEntry
Access: not-accessible
Description: Detailed view of the bridge flooding process.

floodEntry

Subgroup: gigaBridge
Syntax: FloodEntry
Access: not-accessible
Description: Flooding state for a packet class and source. This object consists of 12 other objects: floodQuotaQualifier, floodQuotaClass, floodBytesSent, floodPacketsSent, floodGeezers, floodLosers, floodHogs, floodSinglePathDiscards, floodPacketsFiltered, floodPacketsPurged, floodLocalCopyPacketsDelivered, and floodLocalCopyPacketsDiscarded.

floodQuotaQualifier

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: TBD!

floodQuotaClass

Subgroup: gigaBridge

Syntax: integer
Access: read-only
Description: TBD!

floodBytesSent

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of bytes in flooded packets (each packet is only counted once). It does not include filtered packets or packets discarded due to buffer limitations.

floodPacketsSent

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: TBD!

floodGeezers

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets that could not be flooded because they had remained in the SCP or in the inbound linecard too long.

floodLosers

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets discarded by the SCP flooding software (at the interrupt level) due to insufficient buffering.

floodHogs

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets for which buffer quota conversion to flooding software optimistic quotas failed. This includes packets that were discarded as well as packets that were successfully flooded.

floodSinglePathDiscards

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets discarded to prevent packet misordering. Certain protocol types are considered single-path and may not be delivered out of order. If the Destination Address for a packet having a single-path

protocol type is learned while the packet is buffered by the flooding software, the packet is discarded.

floodPacketsFiltered

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets discarded by the flooding software because user-configured filtering resulted in no allowed outbound transmit ports.

floodPacketsPurged

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of packets discarded due to the incoming link leaving a forwarding state while they were buffered by flooding software.

floodBytesPurged

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the byte count in packets discarded due to the incoming link leaving a forwarding state while they were buffered by flooding software.

floodLocalCopyPacketsDelivered

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of multicast packets addressed to software modules in the SCP that have been successfully delivered to those modules.

floodLocalCopyPacketsDiscarded

Subgroup: gigaBridge
Syntax: integer
Access: read-only
Description: This object is the count of multicast packets addressed to software modules in the SCP that, because of buffer limitations, could not be delivered to those modules.

tftpDestination

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: IpAddress
Description: This object is the IP address of the host storing the file containing the software image. The object may not be set until the last requested transfer succeeds or fails. So, before setting this object the user should verify that transferStatus does not have the value 'requested' or 'inProgress'."

mopDestination

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: octet string (size 6)
Description: This object is the 802 48-bit address of the host storing the file containing the software image. The object may not be set until the last requested transfer succeeds or fails. Both version 3 and version 4 of MOP are supported.

If a unicast address is specified, the transfer will be from that host. If the MOP multicast address ab-00-00-01-00-00 is used, the transfer will be from the first MOP server to respond.

transferFileName

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: DisplayString
Description: This object is the name of the file containing the software image. The object may not be set until the last requested transfer succeeds or fails.

transferAction

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: integer: none(1), domop(2), dotftp(3)
Description: Set this object to initiate a transfer of a file from a server to the SCP. If transferFileName and mopDestination have been set to values, setting this object to domop(2) will cause a MOP 'load' to be attempted.

If transferFileName and tftpDestination have been set to values, setting this object to dotftp(3) will cause a TFTP 'Get' to be attempted.

This object may not be set until the last requested transfer succeeds or fails, and the last requested copy succeeds or fails.

transferStatus

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: integer: none(1), requested(2), inProgress(3), failed(4), success(5), and failedDueToCheckSum(6)
Description: Poll this variable to determine when the transfer has completed. Polling for completion allows other SNMP requests to be processed while the transfer is in progress.

transferSize

Subgroup: gigaUpgradeSoftware doTransfer
Syntax: integer
Description: Size in bytes of the transferred file. This object can be polled along with the transferStatus object to monitor the progress of the transfer.

copyToSlot

Subgroup: gigaUpgradeSoftware useTransfer
Syntax: integer

Description: The slot number of the card which is to be upgraded. The object may not be set until the last requested copy succeeds or fails.

copyType

Subgroup: gigaUpgradeSoftware useTransfer
Syntax: integer: none(1), elCamino(2), fgl2(3), clock(4), powerSystemController(5)
Description: The type of the image. The object is set after a transfer succeeds or fails.

copyAction

Subgroup: gigaUpgradeSoftware useTransfer
Syntax: integer: none(1), doupgrade(2)
Description: This object initiates a software upgrade of a card. If copyToSlot is non-zero, and copyType has a value other than 'none', and transferStatus has the value 'success', a software upgrade of the card is attempted. The object may not be set until the last requested transfer succeeds, and the last requested copy succeeds or fails. Reading this object always returns the value none(1).

copyStatus

Subgroup: gigaUpgradeSoftware useTransfer
Syntax: integer: none(1), requested(2), inProgress(3), failed(4), success(5), failedDueToCardType(6), failedDueToHwRev(7), failedDueToFwRev(8), and failedDueToBadImage(9)
Access: read-only
Description: This variable can be polled to determine when the card upgrade has completed. Polling for completion allows other SNMP requests to be processed while the copy is in progress.

deleteTransfer

Subgroup: gigaUpgradeSoftware
Syntax: integer: exists(1), notExist(2)
Description: Reading this object reveals whether the result of a file transfer is still present. Exists(1) indicates that the transferred file is present; the value notExist(2) indicates that no file has been transferred or that it has been deleted. Writing the value exists(1) is an error. Writing the value notExist(2) deletes the transferred file. The object may not be set until the last requested transfer succeeds or fails. It may not be set while a copy is in progress either.

arpTimeoutInSeconds

Subgroup: ArpTimingMechanism
Syntax: integer
Description: The maximum amount of time an IP to LAN address translation will be used if it cannot be re-verified.

arpPeriodBetweenRequests

Subgroup: ArpTimingMechanism
Syntax: integer
Description: The time, in seconds, between ARP requests that are used to verify or to discover an IP to LAN address translation.

arpRequestRetries

Subgroup: ArpTimingMechanism
Syntax: integer
Description: The number of times ARP requests are used to verify or to discover an IP to LAN address translation.

snmpDuplicateDiscardInterval

Subgroup: snmpParameters
Syntax: integer
Description: If duplicated snmp messages arrive within this interval, all but the first one will be discarded. The unit is a tenth of second.

B

MIB Object Identifiers

Object identifiers can be used interchangeably with object names. This appendix lists the object identifiers associated with all MIB objects in groups implemented by the GIGAswitch system. These objects are in the FDDI, MIB-II, Bridge, GIGAswitch, and DEC vendor MIBs. Use these tables for quick reference when specifying OIDs with your NMS. The following two entities reside over all of these OID:

- dec: 1.3.6.1.4.1.36
- ema: 1.3.6.1.4.1.36.2

Table B–1 GIGAswitch MIB Object Identifiers

Object	ID
sysobjid	1.3.6.1.4.1.36.2.15
bridges	1.3.6.1.4.1.36.2.15.3
gigaswitch	1.3.6.1.4.1.36.2.15.3.3
minimumGIGAswitch MIBVersionSupported	1.3.6.1.4.1.36.2.15.3.3.1.0
maximumGIGAswitch MIBVersionSupported	1.3.6.1.4.1.36.2.15.3.3.2.0
gigaversion1	1.3.6.1.4.1.36.2.15.3.3.3
gigaBox	1.3.6.1.4.1.36.2.15.3.3.3.1
clockCard	1.3.6.1.4.1.36.2.15.3.3.3.1.1
mgmtMemoryAvail	1.3.6.1.4.1.36.2.15.3.3.3.1.1.1.0
mgmtMemoryAction	1.3.6.1.4.1.36.2.15.3.3.3.1.1.2.0
psc	1.3.6.1.4.1.36.2.15.3.3.3.1.2
pscStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.2.1.0
pscFwRev	1.3.6.1.4.1.36.2.15.3.3.3.1.2.2.0
pscHwRev	1.3.6.1.4.1.36.2.15.3.3.3.1.2.3.0
keyswitchPosition	1.3.6.1.4.1.36.2.15.3.3.3.1.2.4.0
pscFwImageStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.2.5.0

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
pscBackplaneStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.2.6.0
cabinetTemperature	1.3.6.1.4.1.36.2.15.3.3.3.1.2.7.0
temperatureWarning	1.3.6.1.4.1.36.2.15.3.3.3.1.2.8.0
powerSupply	1.3.6.1.4.1.36.2.15.3.3.3.1.3
rightPowerStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.3.1.0
rightPowerInputSource	1.3.6.1.4.1.36.2.15.3.3.3.1.3.2.0
rightPowerOutputPower	1.3.6.1.4.1.36.2.15.3.3.3.1.3.3.0
leftPowerStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.3.4.0
leftPowerInputSource	1.3.6.1.4.1.36.2.15.3.3.3.1.3.5.0
leftPowerOutputPower	1.3.6.1.4.1.36.2.15.3.3.3.1.3.6.0
slot	1.3.6.1.4.1.36.2.15.3.3.3.1.4
slotNumber	1.3.6.1.4.1.36.2.15.3.3.3.1.4.1.0
scpSlot	1.3.6.1.4.1.36.2.15.3.3.3.1.4.2.0
slotTable	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3
slotEntry	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1
slotIndex	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1.1.n
slotCardStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1.2.n
slotCardType	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1.3.n
slotCardHwRev	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1.4.n
slotCardFwRev	1.3.6.1.4.1.36.2.15.3.3.3.1.4.3.1.5.n
fan	1.3.6.1.4.1.36.2.15.3.3.3.1.5
fanSpeed	1.3.6.1.4.1.36.2.15.3.3.3.1.5.1.0
rightFanStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.5.2.0
leftFanStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.5.3.0
battery	1.3.6.1.4.1.36.2.15.3.3.3.1.6
batteryStatus	1.3.6.1.4.1.36.2.15.3.3.3.1.6.1.0
batteryUsing	1.3.6.1.4.1.36.2.15.3.3.3.1.6.2.0
batteryCharge	1.3.6.1.4.1.36.2.15.3.3.3.1.6.3.0
batteryTest	1.3.6.1.4.1.36.2.15.3.3.3.1.6.4.0
fppn	1.3.6.1.4.1.36.2.15.3.3.3.1.7
fppnTable	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
fppnEntry	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1.1
fppnSlotNumber	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1.1.1.n
fppnPortOfThatSlot	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1.1.2.n
fppnIfIndex	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1.1.3.n
fppnBridgePortNumber	1.3.6.1.4.1.36.2.15.3.3.3.1.7.1.1.4.n
gigaBridge	1.3.6.1.4.1.36.2.15.3.3.3.2
filterByReferencedExpression	1.3.6.1.4.1.36.2.15.3.3.3.2.1
ebrNportMatrixNameTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.1
ebrNportMatrixNameEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.1.1
ebrNportMatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.1.1.1.n
ebrNportMatrixValue	1.3.6.1.4.1.36.2.15.3.3.3.2.1.1.1.2.n
ebrNportMatrixStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.1.1.1.3.n
ebrNportSapNameTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2
ebrNportSapNameEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1
ebrNportSapName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1.1.n
ebrNportSapNameSap	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1.2.n
ebrNportSapMatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1.3.n
ebrNportSapNameDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1.4.n
ebrNportSapNameStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.1.2.1.5.n
ebrNportSnapNameTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3
ebrNportSnapNameEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1
ebrNportSnapName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1.1.n
ebrNportSnapNameSnap	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1.2.n
ebrNportSnapMatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1.3.n
ebrNportSnapNameDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1.4.n
ebrNportSnapNameStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.1.3.1.5.n
ebrNportDANameTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4
ebrNportDANameEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1
ebrNportDAName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1.1.n
ebrNportDANameDA	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1.2.n
ebrNportDAMatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1.3.n

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
ebrNportDANameDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1.4.n
ebrNportDANameStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.1.4.1.5.n
ebrNportSANameTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5
ebrNportSANameEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1
ebrNportSAName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1.1.n
ebrNportSANameSA	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1.2.n
ebrNportSAMatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1.3.n
ebrNportSANameDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1.4.n
ebrNportSANameStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.1.5.1.5.n
ebrNportDefaultMatrixValue	1.3.6.1.4.1.36.2.15.3.3.3.2.1.6.0
ebrNportManualFilter	1.3.6.1.4.1.36.2.15.3.3.3.2.1.7.0
ebrNportMatrix NameRowTable	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8
ebrNportMatrix NameRowEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8.1
ebrNportmatrixName	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8.1.1.n
ebrNportMatrixReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8.1.2.n
ebrNportMatrixAllowed ToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8.1.3.n
ebrNportMatrixNameRow Status	1.3.6.1.4.1.36.2.15.3.3.3.2.1.8.1.4.n
filterByBitmapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.2
ebrNportSapProtoTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.1
ebrNportSapProtoEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.1.1
ebrNportSapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.2.1.1.1.n
ebrNportSapReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.2.1.1.2.n
ebrNportSapAllowedToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.2.1.1.3.n
ebrNportSwManualFilter	1.3.6.1.4.1.36.2.15.3.3.3.2.2.10.0
ebrNportSapFilter CharacteristicsTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.2
ebrNportSapFilter CharacteristicsEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.2.1

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
ebrNportSapFilter CharacteristicsSapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.2.2.1.1.n
ebrNportSapDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.2.2.1.2.n
ebrNportSapStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.2.2.1.3.n
ebrNportSnapProtoTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.3
ebrNportSnapProtoEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.3.1
ebrNportSnapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.2.3.1.1.n
ebrNportSnapReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.2.3.1.2.n
ebrNportSnapAllowedToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.2.3.1.3.n
ebrNportSnapFilter CharacteristicsTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.4
ebrNportSnapFilter CharacteristicsEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.4.1
ebrNportSnapFilter CharacteristicsSnapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.2.4.1.1.n
ebrNportSnapDisp	1.3.6.1.4.1.36.2.15.3.3.3.2.2.4.1.2.n
ebrNportSnapStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.2.4.1.3.n
ebrNportStaticDATable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.5
ebrNportStaticDAEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.5.1
ebrNportDAAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.2.5.1.1.n
ebrNportDAReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.2.5.1.2.n
ebrNportDAAllowedToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.2.5.1.3.n
ebrNportStaticDAFilter CharacteristicsTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.6
ebrNportStaticDAFilter CharacteristicsEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.6.1
ebrNportDestinationAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.2.6.1.1.n
ebrNportDADisp	1.3.6.1.4.1.36.2.15.3.3.3.2.2.6.1.2.n
ebrNportDAStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.2.6.1.3.n
ebrNportStaticSATable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.7
ebrNportStaticSAEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.7.1
ebrNportSAAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.2.7.1.1.n
ebrNportSAReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.2.7.1.2.n

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
ebrNportSAAllowedToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.2.7.1.3.n
ebrNportStaticSAFilter CharacteristicsTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.8
ebrNportStaticSAFilter CharacteristicsEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.8.1
ebrNportSourceAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.2.8.1.1.n
ebrNportSADisp	1.3.6.1.4.1.36.2.15.3.3.3.2.2.8.1.2.n
ebrNportSAStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.2.8.1.3.n
ebrNportSwTable	1.3.6.1.4.1.36.2.15.3.3.3.2.2.9
ebrNportSwEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.2.9.1
ebrNportSwReceivePort	1.3.6.1.4.1.36.2.15.3.3.3.2.2.9.1.1.n
ebrNportSwAllowedToGoTo	1.3.6.1.4.1.36.2.15.3.3.3.2.2.9.1.2.n
ebrNportPortNumTable	1.3.6.1.4.1.36.2.15.3.3.3.2.3
ebrNportPortNumEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.3.1
ebrNportPortNumAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.3.1.1.n
ebrNportPortNum	1.3.6.1.4.1.36.2.15.3.3.3.2.3.1.2.n
ebrNportPortNumStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.3.1.3.n
serviceClassAssignments	1.3.6.1.4.1.36.2.15.3.3.3.2.5
ebrNportSapSvcTable	1.3.6.1.4.1.36.2.15.3.3.3.2.5.1
ebrNportSapSvcEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.5.1.1
ebrNportSapSvcSapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.5.1.1.1.n
ebrNportSapSvc	1.3.6.1.4.1.36.2.15.3.3.3.2.5.1.1.2.n
ebrNportSapSvcStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.5.1.1.3.n
ebrNportSnapSvcTable	1.3.6.1.4.1.36.2.15.3.3.3.2.5.2
ebrNportSnapSvcEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.5.2.1
ebrNportSnapSvcSnapValue	1.3.6.1.4.1.36.2.15.3.3.3.2.5.2.1.1.n
ebrNportSnapSvc	1.3.6.1.4.1.36.2.15.3.3.3.2.5.2.1.2.n
ebrNportSnapSvcStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.5.2.1.3.n
ebrNportDASvcTable	1.3.6.1.4.1.36.2.15.3.3.3.2.5.3
ebrNportDASvcEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.5.3.1
ebrNportSvcAddress	1.3.6.1.4.1.36.2.15.3.3.3.2.5.3.1.1.n
ebrNportSvc	1.3.6.1.4.1.36.2.15.3.3.3.2.5.3.1.2.n

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
ebrNportSvcStatus	1.3.6.1.4.1.36.2.15.3.3.3.2.5.3.1.3.n
flooding	1.3.6.1.4.1.36.2.15.3.3.3.2.6
floodUnknownUnicastRate	1.3.6.1.4.1.36.2.15.3.3.3.2.6.1.0
floodMulticastRate	1.3.6.1.4.1.36.2.15.3.3.3.2.6.2.0
floodTable	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3
floodEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1
floodQuotaQualifier	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.1.n
floodPacketsPurged	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.10.n
floodBytesPurged	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.11.n
floodLocalCopyPackets Delivered	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.12.n
floodLocalCopyPackets Discarded	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.13.n
floodQuotaClass	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.2.n
floodBytesSent	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.3.n
floodPacketsSent	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.4.n
floodGeezers	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.5.n
floodLosers	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.6.n
floodHogs	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.7.n
floodSinglePathDiscards	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.8.n
floodPacketsFiltered	1.3.6.1.4.1.36.2.15.3.3.3.2.6.3.1.9.n
cutThrough	1.3.6.1.4.1.36.2.15.3.3.3.2.7
cutThroughTable	1.3.6.1.4.1.36.2.15.3.3.3.2.7.1
cutThroughEntry	1.3.6.1.4.1.36.2.15.3.3.3.2.7.1.1
cutThroughBridgePort	1.3.6.1.4.1.36.2.15.3.3.3.2.7.1.1.1.n
cutThroughInbound	1.3.6.1.4.1.36.2.15.3.3.3.2.7.1.1.2.n
cutThroughOutbound	1.3.6.1.4.1.36.2.15.3.3.3.2.7.1.1.3.n
gigaUpgradeSoftware	1.3.6.1.4.1.36.2.15.3.3.3.3
doTransfer	1.3.6.1.4.1.36.2.15.3.3.3.3.1
tftpDestination	1.3.6.1.4.1.36.2.15.3.3.3.3.1.1.0
mopDestination	1.3.6.1.4.1.36.2.15.3.3.3.3.1.2.0
transferFileName	1.3.6.1.4.1.36.2.15.3.3.3.3.1.3.0

(continued on next page)

Table B–1 (Cont.) GIGAswitch MIB Object Identifiers

Object	ID
transferAction	1.3.6.1.4.1.36.2.15.3.3.3.3.1.4.0
transferStatus	1.3.6.1.4.1.36.2.15.3.3.3.3.1.5.0
transferSize	1.3.6.1.4.1.36.2.15.3.3.3.3.1.6.0
useTransfer	1.3.6.1.4.1.36.2.15.3.3.3.3.2
copyToSlot	1.3.6.1.4.1.36.2.15.3.3.3.3.2.1.0
copyType	1.3.6.1.4.1.36.2.15.3.3.3.3.2.2.0
copyAction	1.3.6.1.4.1.36.2.15.3.3.3.3.2.3.0
copyStatus	1.3.6.1.4.1.36.2.15.3.3.3.3.2.4.0
deleteTransfer	1.3.6.1.4.1.36.2.15.3.3.3.3.3.0

Table B–2 MIB-II Object Identifiers

Object	ID
mib-2	1.3.6.1.2.1
system	1.3.6.1.2.1.1
sysDescr	1.3.6.1.2.1.1.1.0
sysObjectID	1.3.6.1.2.1.1.2.0
sysUpTime	1.3.6.1.2.1.1.3.0
sysContact	1.3.6.1.2.1.1.4.0
sysName	1.3.6.1.2.1.1.5.0
sysLocation	1.3.6.1.2.1.1.6.0
sysServices	1.3.6.1.2.1.1.7.0
transmission	1.3.6.1.2.1.10
snmp	1.3.6.1.2.1.11
snmpInPkts	1.3.6.1.2.1.11.1.0
snmpInBadValues	1.3.6.1.2.1.11.10.0
snmpInReadOnlys	1.3.6.1.2.1.11.11.0
snmpInGenErrs	1.3.6.1.2.1.11.12.0
snmpInTotalReqVars	1.3.6.1.2.1.11.13.0
snmpInTotalSetVars	1.3.6.1.2.1.11.14.0
snmpInGetRequests	1.3.6.1.2.1.11.15.0
snmpInGetNexts	1.3.6.1.2.1.11.16.0
snmpInSetRequests	1.3.6.1.2.1.11.17.0
snmpInGetResponses	1.3.6.1.2.1.11.18.0
snmpInTraps	1.3.6.1.2.1.11.19.0
snmpOutPkts	1.3.6.1.2.1.11.2.0
snmpOutTooBigs	1.3.6.1.2.1.11.20.0
snmpOutNoSuchNames	1.3.6.1.2.1.11.21.0
snmpOutBadValues	1.3.6.1.2.1.11.22.0
snmpOutReadOnlys	1.3.6.1.2.1.11.23.0
snmpOutGenErrs	1.3.6.1.2.1.11.24.0
snmpOutGetRequests	1.3.6.1.2.1.11.25.0
snmpOutGetNexts	1.3.6.1.2.1.11.26.0
snmpOutSetRequests	1.3.6.1.2.1.11.27.0

(continued on next page)

Table B–2 (Cont.) MIB-II Object Identifiers

Object	ID
snmpOutGetResponses	1.3.6.1.2.1.11.28.0
snmpOutTraps	1.3.6.1.2.1.11.29.0
snmpInBadVersions	1.3.6.1.2.1.11.3.0
snmpEnableAuthenTraps	1.3.6.1.2.1.11.30.0
snmpEnableAuthenTraps	1.3.6.1.2.1.11.30.0
snmpInBadCommunityNames	1.3.6.1.2.1.11.4.0
snmpInBadCommunityUses	1.3.6.1.2.1.11.5.0
snmpInASNParseErrs	1.3.6.1.2.1.11.6.0
snmpInBadTypes	1.3.6.1.2.1.11.7.0
snmpInTooBigs	1.3.6.1.2.1.11.8.0
snmpInNoSuchNames	1.3.6.1.2.1.11.9.0
interfaces	1.3.6.1.2.1.2
ifNumber	1.3.6.1.2.1.2.1.0
ifTable	1.3.6.1.2.1.2.2
ifEntry	1.3.6.1.2.1.2.2.1
ifIndex	1.3.6.1.2.1.2.2.1.1.n
ifInOctets	1.3.6.1.2.1.2.2.1.10.n
ifInUcastPkts	1.3.6.1.2.1.2.2.1.11.n
ifInNUcastPkts	1.3.6.1.2.1.2.2.1.12.n
ifInDiscards	1.3.6.1.2.1.2.2.1.13.n
ifInErrors	1.3.6.1.2.1.2.2.1.14.n
ifInUnknownProtos	1.3.6.1.2.1.2.2.1.15.n
ifOutOctets	1.3.6.1.2.1.2.2.1.16.n
ifOutUcastPkts	1.3.6.1.2.1.2.2.1.17.n
ifOutNUcastPkts	1.3.6.1.2.1.2.2.1.18.n
ifOutDiscards	1.3.6.1.2.1.2.2.1.19.n
ifDescr	1.3.6.1.2.1.2.2.1.2.n
ifOutErrors	1.3.6.1.2.1.2.2.1.20.n
ifOutQLen	1.3.6.1.2.1.2.2.1.21.n
ifSpecific	1.3.6.1.2.1.2.2.1.22.n
ifType	1.3.6.1.2.1.2.2.1.3.n

(continued on next page)

Table B–2 (Cont.) MIB-II Object Identifiers

Object	ID
ifMtu	1.3.6.1.2.1.2.2.1.4.n
ifSpeed	1.3.6.1.2.1.2.2.1.5.n
ifPhysAddress	1.3.6.1.2.1.2.2.1.6.n
ifAdminStatus	1.3.6.1.2.1.2.2.1.7.n
ifOperStatus	1.3.6.1.2.1.2.2.1.8.n
ifLastChange	1.3.6.1.2.1.2.2.1.9.n
at	1.3.6.1.2.1.3
atTable	1.3.6.1.2.1.3.1
atEntry	1.3.6.1.2.1.3.1.1
atIfIndex	1.3.6.1.2.1.3.1.1.1.n
atPhysAddress	1.3.6.1.2.1.3.1.1.2.n
atNetAddress	1.3.6.1.2.1.3.1.1.3.n
ip	1.3.6.1.2.1.4
ipForwarding	1.3.6.1.2.1.4.1.0
ipOutRequests	1.3.6.1.2.1.4.10.0
ipOutDiscards	1.3.6.1.2.1.4.11.0
ipOutNoRoutes	1.3.6.1.2.1.4.12.0
ipReasmTimeout	1.3.6.1.2.1.4.13.0
ipReasmReqds	1.3.6.1.2.1.4.14.0
ipReasmOKs	1.3.6.1.2.1.4.15.0
ipReasmFails	1.3.6.1.2.1.4.16.0
ipFragOKs	1.3.6.1.2.1.4.17.0
ipFragFails	1.3.6.1.2.1.4.18.0
ipFragCreates	1.3.6.1.2.1.4.19.0
ipDefaultTTL	1.3.6.1.2.1.4.2.0
ipAddrTable	1.3.6.1.2.1.4.20
ipAddrEntry	1.3.6.1.2.1.4.20.1
ipAdEntAddr	1.3.6.1.2.1.4.20.1.1.n
ipAdEntIfIndex	1.3.6.1.2.1.4.20.1.2.n
ipAdEntNetMask	1.3.6.1.2.1.4.20.1.3.n
ipAdEntBcastAddr	1.3.6.1.2.1.4.20.1.4.n

(continued on next page)

Table B–2 (Cont.) MIB-II Object Identifiers

Object	ID
ipAdEntReasmMaxSiz	1.3.6.1.2.1.4.20.1.5.0
ipAdEntReasmMaxSize	1.3.6.1.2.1.4.20.1.5.n
ipRouteTable	1.3.6.1.2.1.4.21
ipRoutingTable	1.3.6.1.2.1.4.21.0
ipRouteEntry	1.3.6.1.2.1.4.21.1
ipRouteDest	1.3.6.1.2.1.4.21.1.1.n
ipRouteAge	1.3.6.1.2.1.4.21.1.10.n
ipRouteMask	1.3.6.1.2.1.4.21.1.11.n
ipRouteMetric5	1.3.6.1.2.1.4.21.1.12.n
ipRouteInfo	1.3.6.1.2.1.4.21.1.13.n
ipRouteIfIndex	1.3.6.1.2.1.4.21.1.2.n
ipRouteMetric1	1.3.6.1.2.1.4.21.1.3.n
ipRouteMetric2	1.3.6.1.2.1.4.21.1.4.n
ipRouteMetric3	1.3.6.1.2.1.4.21.1.5.n
ipRouteMetric4	1.3.6.1.2.1.4.21.1.6.n
ipRouteNextHop	1.3.6.1.2.1.4.21.1.7.n
ipRouteType	1.3.6.1.2.1.4.21.1.8.n
ipRouteProto	1.3.6.1.2.1.4.21.1.9.n
ipNetToMediaTable	1.3.6.1.2.1.4.22
ipNetToMediaEntry	1.3.6.1.2.1.4.22.1
ipNetToMediaIfIndex	1.3.6.1.2.1.4.22.1.1.n
ipNetToMediaPhysAddress	1.3.6.1.2.1.4.22.1.2.n
ipNetToMediaNetAddress	1.3.6.1.2.1.4.22.1.3.n
ipNetToMediaType	1.3.6.1.2.1.4.22.1.4.n
ipRoutingDiscards	1.3.6.1.2.1.4.23.0
ipInReceives	1.3.6.1.2.1.4.3.0
ipInHdrErrors	1.3.6.1.2.1.4.4.0
ipInAddrErrors	1.3.6.1.2.1.4.5.0
ipForwDatagrams	1.3.6.1.2.1.4.6.0
ipInUnknownProtos	1.3.6.1.2.1.4.7.0
ipInDiscards	1.3.6.1.2.1.4.8.0

(continued on next page)

Table B–2 (Cont.) MIB-II Object Identifiers

Object	ID
ipInDelivers	1.3.6.1.2.1.4.9.0
icmp	1.3.6.1.2.1.5
icmpInMsgs	1.3.6.1.2.1.5.1.0
icmpInTimestamps	1.3.6.1.2.1.5.10.0
icmpInTimestampReps	1.3.6.1.2.1.5.11.0
icmpInAddrMasks	1.3.6.1.2.1.5.12.0
icmpInAddrMaskReps	1.3.6.1.2.1.5.13.0
icmpOutMsgs	1.3.6.1.2.1.5.14.0
icmpOutErrors	1.3.6.1.2.1.5.15.0
icmpOutDestUnreachs	1.3.6.1.2.1.5.16.0
icmpOutTimeExcds	1.3.6.1.2.1.5.17.0
icmpOutParmProbs	1.3.6.1.2.1.5.18.0
icmpOutSrcQuenchs	1.3.6.1.2.1.5.19.0
icmpInErrors	1.3.6.1.2.1.5.2.0
icmpOutRedirects	1.3.6.1.2.1.5.20.0
icmpOutEchos	1.3.6.1.2.1.5.21.0
icmpOutEchoReps	1.3.6.1.2.1.5.22.0
icmpOutTimestamps	1.3.6.1.2.1.5.23.0
icmpOutTimestampReps	1.3.6.1.2.1.5.24.0
icmpOutAddrMasks	1.3.6.1.2.1.5.25.0
icmpOutAddrMaskReps	1.3.6.1.2.1.5.26.0
icmpInDestUnreachs	1.3.6.1.2.1.5.3.0
icmpInTimeExcds	1.3.6.1.2.1.5.4.0
icmpInParmProbs	1.3.6.1.2.1.5.5.0
icmpInSrcQuenchs	1.3.6.1.2.1.5.6.0
icmpInRedirects	1.3.6.1.2.1.5.7.0
icmpInEchos	1.3.6.1.2.1.5.8.0
icmpInEchoReps	1.3.6.1.2.1.5.9.0
udp	1.3.6.1.2.1.7
udpInDatagrams	1.3.6.1.2.1.7.1.0
udpNoPorts	1.3.6.1.2.1.7.2.0

(continued on next page)

Table B–2 (Cont.) MIB-II Object Identifiers

Object	ID
udpInErrors	1.3.6.1.2.1.7.3.0
udpOutDatagrams	1.3.6.1.2.1.7.4.0
udpTable	1.3.6.1.2.1.7.5
udpEntry	1.3.6.1.2.1.7.5.1
udpLocalAddress	1.3.6.1.2.1.7.5.1.1.n
udpLocalPort	1.3.6.1.2.1.7.5.1.2.n

Table B–3 Bridge MIB Object Identifiers

Object	ID
dot1dBridge	1.3.6.1.2.1.17
dot1dBase	1.3.6.1.2.1.17.1
dot1dBaseBridgeAddress	1.3.6.1.2.1.17.1.1.0
dot1dBaseNumPorts	1.3.6.1.2.1.17.1.2.0
dot1dBaseType	1.3.6.1.2.1.17.1.3.0
dot1dBasePortTable	1.3.6.1.2.1.17.1.4
dot1dBasePortEntry	1.3.6.1.2.1.17.1.4.1
dot1dBasePort	1.3.6.1.2.1.17.1.4.1.1.n
dot1dBasePortIfIndex	1.3.6.1.2.1.17.1.4.1.2.n
dot1dBasePortCircuit	1.3.6.1.2.1.17.1.4.1.3.n
dot1dBasePortDelay ExceededDiscards	1.3.6.1.2.1.17.1.4.1.4.n
dot1dBasePortMtu ExceededDiscards	1.3.6.1.2.1.17.1.4.1.5.n
dot1dStp	1.3.6.1.2.1.17.2
dot1dStpProtocolSpecification	1.3.6.1.2.1.17.2.1.0
dot1dStpHoldTime	1.3.6.1.2.1.17.2.10.0
dot1dStpForwardDelay	1.3.6.1.2.1.17.2.11.0
dot1dStpBridgeMaxAge	1.3.6.1.2.1.17.2.12.0
dot1dStpBridgeHelloTime	1.3.6.1.2.1.17.2.13.0
dot1dStpBridgeForwardDelay	1.3.6.1.2.1.17.2.14.0
dot1dStpPortTable	1.3.6.1.2.1.17.2.15
dot1dStpPortEntry	1.3.6.1.2.1.17.2.15.1
dot1dStpPort	1.3.6.1.2.1.17.2.15.1.1.n
dot1dStpPortForward Transitions	1.3.6.1.2.1.17.2.15.1.10.n
dot1dStpPortPriority	1.3.6.1.2.1.17.2.15.1.2.n
dot1dStpPortState	1.3.6.1.2.1.17.2.15.1.3.n
dot1dStpPortEnable	1.3.6.1.2.1.17.2.15.1.4.n
dot1dStpPortPathCost	1.3.6.1.2.1.17.2.15.1.5.n
dot1dStpPortDesignatedRoot	1.3.6.1.2.1.17.2.15.1.6.n
dot1dStpPortDesignatedCost	1.3.6.1.2.1.17.2.15.1.7.n

(continued on next page)

Table B–3 (Cont.) Bridge MIB Object Identifiers

Object	ID
dot1dStpPortDesignated Bridge	1.3.6.1.2.1.17.2.15.1.8.n
dot1dStpPortDesignatedPort	1.3.6.1.2.1.17.2.15.1.9.n
dot1dStpPriority	1.3.6.1.2.1.17.2.2.0
dot1dStpTimeSince TopologyChange	1.3.6.1.2.1.17.2.3.0
dot1dStpTopChanges	1.3.6.1.2.1.17.2.4.0
dot1dStpDesignatedRoot	1.3.6.1.2.1.17.2.5.0
dot1dStpRootCost	1.3.6.1.2.1.17.2.6.0
dot1dStpRootPort	1.3.6.1.2.1.17.2.7.0
dot1dStpMaxAge	1.3.6.1.2.1.17.2.8.0
dot1dStpHelloTime	1.3.6.1.2.1.17.2.9.0
dot1dTp	1.3.6.1.2.1.17.4
dot1dTpLearnedEntry Discards	1.3.6.1.2.1.17.4.1.0
dot1dTpAgingTime	1.3.6.1.2.1.17.4.2.0
dot1dTpFdbTable	1.3.6.1.2.1.17.4.3
dot1dTpFdbEntry	1.3.6.1.2.1.17.4.3.1
dot1dTpFdbAddress	1.3.6.1.2.1.17.4.3.1.1.n
dot1dTpFdbPort	1.3.6.1.2.1.17.4.3.1.2.n
dot1dTpFdbStatus	1.3.6.1.2.1.17.4.3.1.3.n
dot1dTpPortTable	1.3.6.1.2.1.17.4.4
dot1dTpPortEntry	1.3.6.1.2.1.17.4.4.1
dot1dTpPort	1.3.6.1.2.1.17.4.4.1.1.n
dot1dTpPortMaxInfo	1.3.6.1.2.1.17.4.4.1.2.n
dot1dTpPortInFrames	1.3.6.1.2.1.17.4.4.1.3.n
dot1dTpPortOutFrames	1.3.6.1.2.1.17.4.4.1.4.n
dot1dTpPortInDiscards	1.3.6.1.2.1.17.4.4.1.5.n
dot1dStatic	1.3.6.1.2.1.17.5
dot1dStaticTable	1.3.6.1.2.1.17.5.1
dot1dStaticEntry	1.3.6.1.2.1.17.5.1.1
dot1dStaticAddress	1.3.6.1.2.1.17.5.1.1.1.n

(continued on next page)

Table B–3 (Cont.) Bridge MIB Object Identifiers

Object	ID
dot1dStaticReceivePort	1.3.6.1.2.1.17.5.1.1.2.n
dot1dStaticAllowedToGoTo	1.3.6.1.2.1.17.5.1.1.3.n
dot1dStaticStatus	1.3.6.1.2.1.17.5.1.1.4.n

Table B–4 Dec_vendor MIB Object Identifiers

Object	ID
decMIBextension	1.3.6.1.4.1.36.2.18
elanext	1.3.6.1.4.1.36.2.18.1
efddi	1.3.6.1.4.1.36.2.18.1.1
efddiSMT	1.3.6.1.4.1.36.2.18.1.1.1
efddiSMTTable	1.3.6.1.4.1.36.2.18.1.1.1.1
efddiSMTEntry	1.3.6.1.4.1.36.2.18.1.1.1.1.1
eSMTIndex	1.3.6.1.4.1.36.2.18.1.1.1.1.1.1.n
eSMTStationType	1.3.6.1.4.1.36.2.18.1.1.1.1.1.2.n
eSMTTracesReceived	1.3.6.1.4.1.36.2.18.1.1.1.1.1.3.n
efddiMAC	1.3.6.1.4.1.36.2.18.1.1.2
efddiMACTable	1.3.6.1.4.1.36.2.18.1.1.2.1
efddiMACEntry	1.3.6.1.4.1.36.2.18.1.1.2.1.1
eMACSMTIndex	1.3.6.1.4.1.36.2.18.1.1.2.1.1.1.n
eMACRingErrorReason	1.3.6.1.4.1.36.2.18.1.1.2.1.1.10.n
eMACRingInitializations Initiated	1.3.6.1.4.1.36.2.18.1.1.2.1.1.11.n
eMACRingInitializations Received	1.3.6.1.4.1.36.2.18.1.1.2.1.1.12.n
eMACRingBeaconing Initiated	1.3.6.1.4.1.36.2.18.1.1.2.1.1.13.n
eMACDuplicateAddress TestFailures	1.3.6.1.4.1.36.2.18.1.1.2.1.1.14.n
eMACDuplicateTokens Detected	1.3.6.1.4.1.36.2.18.1.1.2.1.1.15.n
eMACUpstreamNbrDupl AddressFlag	1.3.6.1.4.1.36.2.18.1.1.2.1.1.16.n
eMACTracesInitiated	1.3.6.1.4.1.36.2.18.1.1.2.1.1.17.n
eMACRestrictedToken Timeout	1.3.6.1.4.1.36.2.18.1.1.2.1.1.18.n
eMACFrameStatusErrors	1.3.6.1.4.1.36.2.18.1.1.2.1.1.19.n
eMACIndex	1.3.6.1.4.1.36.2.18.1.1.2.1.1.2.n
eMACFrameAlignmentErrors	1.3.6.1.4.1.36.2.18.1.1.2.1.1.20.n
eMACTransmitUnderruns	1.3.6.1.4.1.36.2.18.1.1.2.1.1.21.n
eMACLinkIndex	1.3.6.1.4.1.36.2.18.1.1.2.1.1.3.n

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
eMACLinkState	1.3.6.1.4.1.36.2.18.1.1.2.1.1.4.n
eMACRingPurgerState	1.3.6.1.4.1.36.2.18.1.1.2.1.1.5.n
eMACRingPurgerEnable	1.3.6.1.4.1.36.2.18.1.1.2.1.1.6.n
eMACRingPurgeErrors	1.3.6.1.4.1.36.2.18.1.1.2.1.1.7.n
eMACFrameStripMode	1.3.6.1.4.1.36.2.18.1.1.2.1.1.8.n
eMACFCIStripErrors	1.3.6.1.4.1.36.2.18.1.1.2.1.1.9.n
efddiPORT	1.3.6.1.4.1.36.2.18.1.1.3
efddiPORTTable	1.3.6.1.4.1.36.2.18.1.1.3.1
efddiPORTEntry	1.3.6.1.4.1.36.2.18.1.1.3.1.1
ePORTSMTIndex	1.3.6.1.4.1.36.2.18.1.1.3.1.1.1.n
ePORTIndex	1.3.6.1.4.1.36.2.18.1.1.3.1.1.2.n
ePORTPHYIndex	1.3.6.1.4.1.36.2.18.1.1.3.1.1.3.n
ePORTPMDType	1.3.6.1.4.1.36.2.18.1.1.3.1.1.4.n
ePORTPHYState	1.3.6.1.4.1.36.2.18.1.1.3.1.1.5.n
ePORTRejectReason	1.3.6.1.4.1.36.2.18.1.1.3.1.1.6.n
ePORTConnections Completed	1.3.6.1.4.1.36.2.18.1.1.3.1.1.7.n
ePORTTNEExpRejects	1.3.6.1.4.1.36.2.18.1.1.3.1.1.8.n
ePORTElasticityBufferErrors	1.3.6.1.4.1.36.2.18.1.1.3.1.1.9.n
efddiFDX	1.3.6.1.4.1.36.2.18.1.1.4
efddiFDXTable	1.3.6.1.4.1.36.2.18.1.1.4.1
efddiFDXEntry	1.3.6.1.4.1.36.2.18.1.1.4.1.1
eFDXSMTIndex	1.3.6.1.4.1.36.2.18.1.1.4.1.1.1.n
eFDXMACIndex	1.3.6.1.4.1.36.2.18.1.1.4.1.1.2.n
eFDXEnable	1.3.6.1.4.1.36.2.18.1.1.4.1.1.3.n
eFDXOp	1.3.6.1.4.1.36.2.18.1.1.4.1.1.4.n
eFDXState	1.3.6.1.4.1.36.2.18.1.1.4.1.1.5.n
esystem	1.3.6.1.4.1.36.2.18.1.2
esysChar	1.3.6.1.4.1.36.2.18.1.2.1
esysRomVersion	1.3.6.1.4.1.36.2.18.1.2.1.1.0
esysInitSwitch	1.3.6.1.4.1.36.2.18.1.2.1.2.0

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
esysResetDefaultsSwitch	1.3.6.1.4.1.36.2.18.1.2.1.3.0
esysGatewayAddress	1.3.6.1.4.1.36.2.18.1.2.1.4.0
esysTrapAddressTable	1.3.6.1.4.1.36.2.18.1.2.1.5
esysTrapEntry	1.3.6.1.4.1.36.2.18.1.2.1.5.1
esysTrapAddress	1.3.6.1.4.1.36.2.18.1.2.1.5.1.1.n
esysUpdateSwitch	1.3.6.1.4.1.36.2.18.1.2.1.6.0
esysLastLoadHost	1.3.6.1.4.1.36.2.18.1.2.1.7.0
esysStatus	1.3.6.1.4.1.36.2.18.1.2.2
esysDeviceState	1.3.6.1.4.1.36.2.18.1.2.2.1.0
esysDeviceBrokenReason	1.3.6.1.4.1.36.2.18.1.2.2.2.0
esysNvramFailed	1.3.6.1.4.1.36.2.18.1.2.2.3.0
esysCounters	1.3.6.1.4.1.36.2.18.1.2.3
esysPowerups	1.3.6.1.4.1.36.2.18.1.2.3.1.0
esysMgmtResets	1.3.6.1.4.1.36.2.18.1.2.3.2.0
esysUnsolicitedResets	1.3.6.1.4.1.36.2.18.1.2.3.3.0
esysConcConfig	1.3.6.1.4.1.36.2.18.1.2.4
esysFRUConfigTable	1.3.6.1.4.1.36.2.18.1.2.4.1
esysFRUConfigEntry	1.3.6.1.4.1.36.2.18.1.2.4.1.1
esysFRUIndex	1.3.6.1.4.1.36.2.18.1.2.4.1.1.1.n
esysFRUSlot	1.3.6.1.4.1.36.2.18.1.2.4.1.1.2.n
esysFRUDesc	1.3.6.1.4.1.36.2.18.1.2.4.1.1.3.n
esysFRUType	1.3.6.1.4.1.36.2.18.1.2.4.1.1.4.n
esysFRURev	1.3.6.1.4.1.36.2.18.1.2.4.1.1.5.n
esysFRUState	1.3.6.1.4.1.36.2.18.1.2.4.1.1.6.n
esysFddiPortTrapSwitch	1.3.6.1.4.1.36.2.18.1.2.4.2.0
einterfaces	1.3.6.1.4.1.36.2.18.1.3
eifTable	1.3.6.1.4.1.36.2.18.1.3.1
eifEntry	1.3.6.1.4.1.36.2.18.1.3.1.1
eifIndex	1.3.6.1.4.1.36.2.18.1.3.1.1.1.n
eifBadFramesReceived	1.3.6.1.4.1.36.2.18.1.3.1.1.2.n
eifReceiveOverrun	1.3.6.1.4.1.36.2.18.1.3.1.1.3.n

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
eifOversizeFrames	1.3.6.1.4.1.36.2.18.1.3.1.1.4.n
eifTransmitFramesError	1.3.6.1.4.1.36.2.18.1.3.1.1.5.n
eifMgmtSetsAllowedSwitch	1.3.6.1.4.1.36.2.18.1.3.1.1.6.n
ebridge	1.3.6.1.4.1.36.2.18.1.4
ebrChar	1.3.6.1.4.1.36.2.18.1.4.1
ebrLB100SpanningTreeVer	1.3.6.1.4.1.36.2.18.1.4.1.1.0
ebrTopologyChangeTimer	1.3.6.1.4.1.36.2.18.1.4.1.10.0
ebrManualFilterSwitch	1.3.6.1.4.1.36.2.18.1.4.1.11.0
ebrFragmentationSwitch	1.3.6.1.4.1.36.2.18.1.4.1.12.0
ebrRemoveMgmtAddress	1.3.6.1.4.1.36.2.18.1.4.1.13.0
ebrRemoveMgmtProto	1.3.6.1.4.1.36.2.18.1.4.1.14.0
ebr802SpanningTreeVer	1.3.6.1.4.1.36.2.18.1.4.1.2.0
ebrMaxForwarding DBEntries	1.3.6.1.4.1.36.2.18.1.4.1.3.0
ebrMaxNVForwarding DBEntries	1.3.6.1.4.1.36.2.18.1.4.1.4.0
ebrMaxProtocolDBEntries	1.3.6.1.4.1.36.2.18.1.4.1.5.0
ebrMaxNVProtocolDBEntries	1.3.6.1.4.1.36.2.18.1.4.1.6.0
ebrForwardingDBPurge Threshold	1.3.6.1.4.1.36.2.18.1.4.1.7.0
ebrPortTestPassedThreshold	1.3.6.1.4.1.36.2.18.1.4.1.8.0
ebrPortTestInterval	1.3.6.1.4.1.36.2.18.1.4.1.9.0
ebrMultiFiltSw	1.3.6.1.4.1.36.2.18.1.4.10
ebrMultiSwTable	1.3.6.1.4.1.36.2.18.1.4.10.1
ebrMultiSwEntry	1.3.6.1.4.1.36.2.18.1.4.10.1.1
ebrMultiSwIndex	1.3.6.1.4.1.36.2.18.1.4.10.1.1.1.n
ebrMultiSwManualFilter	1.3.6.1.4.1.36.2.18.1.4.10.1.1.2.n
ebrMultiSwProtoEnetOther	1.3.6.1.4.1.36.2.18.1.4.10.1.1.3.n
ebrMultiSwProtoSapOther	1.3.6.1.4.1.36.2.18.1.4.10.1.1.4.n
ebrMultiSwProtoSnapOther	1.3.6.1.4.1.36.2.18.1.4.10.1.1.5.n
ebrNTP	1.3.6.1.4.1.36.2.18.1.4.11
ebrNTPTable	1.3.6.1.4.1.36.2.18.1.4.11.1

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
ebrNTPEntry	1.3.6.1.4.1.36.2.18.1.4.11.1.1
ebrNTPtype	1.3.6.1.4.1.36.2.18.1.4.11.1.1.1.n
ebrNTPStatus	1.3.6.1.4.1.36.2.18.1.4.11.1.1.2.n
esysIPXSwitch	1.3.6.1.4.1.36.2.18.1.4.11.2.0
ebrRateLimiting	1.3.6.1.4.1.36.2.18.1.4.12
ebrRateLimitSwitch	1.3.6.1.4.1.36.2.18.1.4.12.1.0
ebrRateLimit	1.3.6.1.4.1.36.2.18.1.4.12.2.0
ebrRateLimitCounterTable	1.3.6.1.4.1.36.2.18.1.4.12.3
ebrRateLimitCounterEntry	1.3.6.1.4.1.36.2.18.1.4.12.3.1
ebrRateLimitPort	1.3.6.1.4.1.36.2.18.1.4.12.3.1.1.n
ebrRateLimitAddressFrames	1.3.6.1.4.1.36.2.18.1.4.12.3.1.2.n
ebrRateLimitProtocolFrames	1.3.6.1.4.1.36.2.18.1.4.12.3.1.3.n
ebrStat	1.3.6.1.4.1.36.2.18.1.4.2
ebrCurrForwarding DBEntries	1.3.6.1.4.1.36.2.18.1.4.2.1.0
ebrCurrNVForwarding DBEntries	1.3.6.1.4.1.36.2.18.1.4.2.2.0
ebrCurrProtocolDBEntries	1.3.6.1.4.1.36.2.18.1.4.2.3.0
ebrCurrNVProtocol DBEntries	1.3.6.1.4.1.36.2.18.1.4.2.4.0
ebrMgmtHeardPort	1.3.6.1.4.1.36.2.18.1.4.2.5.0
ebrLB100BeingPolled	1.3.6.1.4.1.36.2.18.1.4.2.6.0
ebrInactiveForwarding DBEntries	1.3.6.1.4.1.36.2.18.1.4.2.7.0
ebrTimeSinceForwarding DBPurged	1.3.6.1.4.1.36.2.18.1.4.2.8.0
ebrTimeSinceLastHello	1.3.6.1.4.1.36.2.18.1.4.2.9.0
ebrCoun	1.3.6.1.4.1.36.2.18.1.4.3
ebrDeviceFramesLost	1.3.6.1.4.1.36.2.18.1.4.3.1.0
ebrSpanningTreeMode Changes	1.3.6.1.4.1.36.2.18.1.4.3.2.0
ebrSpan	1.3.6.1.4.1.36.2.18.1.4.4
ebrBestRootAge	1.3.6.1.4.1.36.2.18.1.4.4.1.0

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
ebrLB100SpanningTree Compat	1.3.6.1.4.1.36.2.18.1.4.4.10.0
ebrTopologyChangeFlag	1.3.6.1.4.1.36.2.18.1.4.4.2.0
ebrTellParentFlag	1.3.6.1.4.1.36.2.18.1.4.4.3.0
ebrForwardingDBShort AgingTime	1.3.6.1.4.1.36.2.18.1.4.4.4.0
ebrBadHelloLimit	1.3.6.1.4.1.36.2.18.1.4.4.5.0
ebrBadHelloResetTimer	1.3.6.1.4.1.36.2.18.1.4.4.6.0
ebrNoFrameInterval	1.3.6.1.4.1.36.2.18.1.4.4.7.0
ebrLB100PollTime	1.3.6.1.4.1.36.2.18.1.4.4.8.0
ebrLB100ResponseTimeout	1.3.6.1.4.1.36.2.18.1.4.4.9.0
ebrInterfaces	1.3.6.1.4.1.36.2.18.1.4.5
ebrIfTable	1.3.6.1.4.1.36.2.18.1.4.5.1
ebrIfEntry	1.3.6.1.4.1.36.2.18.1.4.5.1.1
ebrIfIndex	1.3.6.1.4.1.36.2.18.1.4.5.1.1.1.n
ebrIfDeviceBytesSent	1.3.6.1.4.1.36.2.18.1.4.5.1.1.10.n
ebrIfDeviceBytesReceived	1.3.6.1.4.1.36.2.18.1.4.5.1.1.11.n
ebrIfDeviceFramesLost	1.3.6.1.4.1.36.2.18.1.4.5.1.1.12.n
ebrIfMultiBytesSent	1.3.6.1.4.1.36.2.18.1.4.5.1.1.13.n
ebrIfMultiBytesReceived	1.3.6.1.4.1.36.2.18.1.4.5.1.1.14.n
ebrIfMultiDeviceFramesSent	1.3.6.1.4.1.36.2.18.1.4.5.1.1.15.n
ebrIfMultiDeviceFrames Received	1.3.6.1.4.1.36.2.18.1.4.5.1.1.16.n
ebrIfMultiDeviceBytesSent	1.3.6.1.4.1.36.2.18.1.4.5.1.1.17.n
ebrIfMultiDeviceBytes Received	1.3.6.1.4.1.36.2.18.1.4.5.1.1.18.n
ebrIfBadBytesReceived	1.3.6.1.4.1.36.2.18.1.4.5.1.1.19.n
ebrIfLinkBrokenReason	1.3.6.1.4.1.36.2.18.1.4.5.1.1.2.n
ebrIfBadHelloLimitExceeded	1.3.6.1.4.1.36.2.18.1.4.5.1.1.20.n
ebrIfPortRestarts	1.3.6.1.4.1.36.2.18.1.4.5.1.1.3.n
ebrIfUnknownDAReceived	1.3.6.1.4.1.36.2.18.1.4.5.1.1.4.n
ebrIfFramesAddrFiltered	1.3.6.1.4.1.36.2.18.1.4.5.1.1.5.n

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
ebrIfMultiFramesFiltered	1.3.6.1.4.1.36.2.18.1.4.5.1.1.6.n
ebrIfFramesProtocolFiltered	1.3.6.1.4.1.36.2.18.1.4.5.1.1.7.n
ebrIfDeviceFramesSent	1.3.6.1.4.1.36.2.18.1.4.5.1.1.8.n
ebrIfDeviceFramesReceived	1.3.6.1.4.1.36.2.18.1.4.5.1.1.9.n
ebrIfEtherTable	1.3.6.1.4.1.36.2.18.1.4.5.2
ebrIfEtherEntry	1.3.6.1.4.1.36.2.18.1.4.5.2.1
ebrIfEthIndex	1.3.6.1.4.1.36.2.18.1.4.5.2.1.1.n
ebrIfEthPhysicalMedium Type	1.3.6.1.4.1.36.2.18.1.4.5.2.1.2.n
ebrIfEthCollisionPresence TestSwitch	1.3.6.1.4.1.36.2.18.1.4.5.2.1.3.n
ebrIfEthCollisionTestFailed	1.3.6.1.4.1.36.2.18.1.4.5.2.1.4.n
ebrIfEthFramingError	1.3.6.1.4.1.36.2.18.1.4.5.2.1.5.n
ebrIfEthLengthError	1.3.6.1.4.1.36.2.18.1.4.5.2.1.6.n
ebrIfEthTransmitMultiple Collisions	1.3.6.1.4.1.36.2.18.1.4.5.2.1.7.n
ebrIfEthCarrierLoss	1.3.6.1.4.1.36.2.18.1.4.5.2.1.8.n
ebrIfEthCollisionLimit Exceeded	1.3.6.1.4.1.36.2.18.1.4.5.2.1.9.n
ebrIfFddiTable	1.3.6.1.4.1.36.2.18.1.4.5.3
ebrIfFddiEntry	1.3.6.1.4.1.36.2.18.1.4.5.3.1
ebrIfFddiIndex	1.3.6.1.4.1.36.2.18.1.4.5.3.1.1.n
ebrIfFddiUnprocessed ErrorPackets	1.3.6.1.4.1.36.2.18.1.4.5.3.1.2.n
ebrIfFddiIpDatagrams Fragmented	1.3.6.1.4.1.36.2.18.1.4.5.3.1.3.n
ebrIfFddiIpDontFragment	1.3.6.1.4.1.36.2.18.1.4.5.3.1.4.n
ebrIfFddiIpIllegalHeader Length	1.3.6.1.4.1.36.2.18.1.4.5.3.1.5.n
ebrIfFddiIpIllegalSize	1.3.6.1.4.1.36.2.18.1.4.5.3.1.6.n
ebrIfSpanTable	1.3.6.1.4.1.36.2.18.1.4.5.4
ebrIfSpanEntry	1.3.6.1.4.1.36.2.18.1.4.5.4.1
ebrIfSpIndex	1.3.6.1.4.1.36.2.18.1.4.5.4.1.1.n

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
ebrIfSpDesigRootAge	1.3.6.1.4.1.36.2.18.1.4.5.4.1.2.n
ebrIfSpForwardDelayTimer	1.3.6.1.4.1.36.2.18.1.4.5.4.1.3.n
ebrIfSpBadHelloCount	1.3.6.1.4.1.36.2.18.1.4.5.4.1.4.n
ebrIfSpPossibleLoopFlag	1.3.6.1.4.1.36.2.18.1.4.5.4.1.5.n
ebrIfSpTopologyChange AckFlag	1.3.6.1.4.1.36.2.18.1.4.5.4.1.6.n
ebrTwoPortStatic	1.3.6.1.4.1.36.2.18.1.4.6
ebrTwoPortStaticTable	1.3.6.1.4.1.36.2.18.1.4.6.1
ebrTwoPortStaticEntry	1.3.6.1.4.1.36.2.18.1.4.6.1.1
ebrTwoPortAddress	1.3.6.1.4.1.36.2.18.1.4.6.1.1.1.n
ebrTwoPortPortNum	1.3.6.1.4.1.36.2.18.1.4.6.1.1.2.n
ebrTwoPortStatus	1.3.6.1.4.1.36.2.18.1.4.6.1.1.3.n
ebrMultiPortStatic	1.3.6.1.4.1.36.2.18.1.4.7
ebrMultiPortStaticTable	1.3.6.1.4.1.36.2.18.1.4.7.1
ebrMultiPortStaticEntry	1.3.6.1.4.1.36.2.18.1.4.7.1.1
ebrMultiPortAddress	1.3.6.1.4.1.36.2.18.1.4.7.1.1.1.n
ebrMultiPortReceivePort	1.3.6.1.4.1.36.2.18.1.4.7.1.1.2.n
ebrMultiPortAllowedToGoTo	1.3.6.1.4.1.36.2.18.1.4.7.1.1.3.n
ebrMultiPortPortNum	1.3.6.1.4.1.36.2.18.1.4.7.1.1.4.n
ebrMultiPortStatus	1.3.6.1.4.1.36.2.18.1.4.7.1.1.5.n
ebrTwoProtoFilt	1.3.6.1.4.1.36.2.18.1.4.8
ebrTwoProtoEnetFilterOther	1.3.6.1.4.1.36.2.18.1.4.8.1.0
ebrTwoProtoSapFilterOther	1.3.6.1.4.1.36.2.18.1.4.8.2.0
ebrTwoProtoSnapFilterOther	1.3.6.1.4.1.36.2.18.1.4.8.3.0
ebrTwoEnetProtoTable	1.3.6.1.4.1.36.2.18.1.4.8.4
ebrTwoEnetProtoEntry	1.3.6.1.4.1.36.2.18.1.4.8.4.1
ebrTwoEnetProtoType	1.3.6.1.4.1.36.2.18.1.4.8.4.1.1.n
ebrTwoEnetProtoStatus	1.3.6.1.4.1.36.2.18.1.4.8.4.1.2.n
ebrTwoSapProtoTable	1.3.6.1.4.1.36.2.18.1.4.8.5
ebrTwoSapProtoEntry	1.3.6.1.4.1.36.2.18.1.4.8.5.1
ebrTwoSapIndex	1.3.6.1.4.1.36.2.18.1.4.8.5.1.1.n

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
ebrTwoSapValue	1.3.6.1.4.1.36.2.18.1.4.8.5.1.2.n
ebrTwoSapStatus	1.3.6.1.4.1.36.2.18.1.4.8.5.1.3.n
ebrTwoSnapProtoTable	1.3.6.1.4.1.36.2.18.1.4.8.6
ebrTwoSnapProtoEntry	1.3.6.1.4.1.36.2.18.1.4.8.6.1
ebrTwoSnapIndex	1.3.6.1.4.1.36.2.18.1.4.8.6.1.1.n
ebrTwoSnapValue	1.3.6.1.4.1.36.2.18.1.4.8.6.1.2.n
ebrTwoSnapStatus	1.3.6.1.4.1.36.2.18.1.4.8.6.1.3.n
ebrMultiProtoFilt	1.3.6.1.4.1.36.2.18.1.4.9
ebrMultiEnetProtoTable	1.3.6.1.4.1.36.2.18.1.4.9.1
ebrMultiEnetProtoEntry	1.3.6.1.4.1.36.2.18.1.4.9.1.1
ebrMultiEnetProtoType	1.3.6.1.4.1.36.2.18.1.4.9.1.1.1.n
ebrMultiEnetReceivePort	1.3.6.1.4.1.36.2.18.1.4.9.1.1.2.n
ebrMultiEnetAllowedToGoTo	1.3.6.1.4.1.36.2.18.1.4.9.1.1.3.n
ebrMultiEnetStatus	1.3.6.1.4.1.36.2.18.1.4.9.1.1.4.n
ebrMultiSapProtoTable	1.3.6.1.4.1.36.2.18.1.4.9.2
ebrMultiSapProtoEntry	1.3.6.1.4.1.36.2.18.1.4.9.2.1
ebrMultiSapValue	1.3.6.1.4.1.36.2.18.1.4.9.2.1.1.n
ebrMultiSapReceivePort	1.3.6.1.4.1.36.2.18.1.4.9.2.1.2.n
ebrMultiSapAllowedToGoTo	1.3.6.1.4.1.36.2.18.1.4.9.2.1.3.n
ebrMultiSapStatus	1.3.6.1.4.1.36.2.18.1.4.9.2.1.4.n
ebrMultiSnapProtoTable	1.3.6.1.4.1.36.2.18.1.4.9.3
ebrMultiSnapProtoEntry	1.3.6.1.4.1.36.2.18.1.4.9.3.1
ebrMultiSnapValue	1.3.6.1.4.1.36.2.18.1.4.9.3.1.1.n
ebrMultiSnapReceivePort	1.3.6.1.4.1.36.2.18.1.4.9.3.1.2.n
ebrMultiSnapAllowedToGoTo	1.3.6.1.4.1.36.2.18.1.4.9.3.1.3.n
ebrMultiSnapStatus	1.3.6.1.4.1.36.2.18.1.4.9.3.1.4.n
eauth	1.3.6.1.4.1.36.2.18.1.5
eauth1	1.3.6.1.4.1.36.2.18.1.5.1
eauthTrapCommunity	1.3.6.1.4.1.36.2.18.1.5.1.1.0
eauthTrapUserTable	1.3.6.1.4.1.36.2.18.1.5.1.2
eauthTrapUserEntry	1.3.6.1.4.1.36.2.18.1.5.1.2.1

(continued on next page)

Table B–4 (Cont.) Dec_vendor MIB Object Identifiers

Object	ID
eauthTrapUserAddr	1.3.6.1.4.1.36.2.18.1.5.1.2.1.1.n
eauthTrapUserStatus	1.3.6.1.4.1.36.2.18.1.5.1.2.1.2.n
eauthReadOnlyCommunity	1.3.6.1.4.1.36.2.18.1.5.1.3.0
eauthReadOnlyUserTable	1.3.6.1.4.1.36.2.18.1.5.1.4
eauthReadOnlyUserEntry	1.3.6.1.4.1.36.2.18.1.5.1.4.1
eauthReadOnlyUserAddr	1.3.6.1.4.1.36.2.18.1.5.1.4.1.1.n
eauthReadOnlyUserMask	1.3.6.1.4.1.36.2.18.1.5.1.4.1.2.n
eauthReadOnlyUserStatus	1.3.6.1.4.1.36.2.18.1.5.1.4.1.3.n
eauthReadWriteCommunity	1.3.6.1.4.1.36.2.18.1.5.1.5.0
eauthReadWriteUserTable	1.3.6.1.4.1.36.2.18.1.5.1.6
eauthReadWriteUserEntry	1.3.6.1.4.1.36.2.18.1.5.1.6.1
eauthReadWriteUserAddr	1.3.6.1.4.1.36.2.18.1.5.1.6.1.1.n
eauthReadWriteUserMask	1.3.6.1.4.1.36.2.18.1.5.1.6.1.2.n
eauthReadWriteUserStatus	1.3.6.1.4.1.36.2.18.1.5.1.6.1.3.n

Table B–5 FDDI MIB Object Identifiers

Object	ID
fddi	1.3.6.1.2.1.10.15
snmpFddiSMT	1.3.6.1.2.1.10.15.1
snmpFddiSMTNumber	1.3.6.1.2.1.10.15.1.1.0
snmpFddiSMTTable	1.3.6.1.2.1.10.15.1.2
snmpFddiSMTEntry	1.3.6.1.2.1.10.15.1.2.1
snmpFddiSMTIndex	1.3.6.1.2.1.10.15.1.2.1.1.n
snmpFddiSMTConfig Capabilities	1.3.6.1.2.1.10.15.1.2.1.10.n
snmpFddiSMTConfigPolicy	1.3.6.1.2.1.10.15.1.2.1.11.n
snmpFddiSMTConnection Policy	1.3.6.1.2.1.10.15.1.2.1.12.n
snmpFddiSMTTNotify	1.3.6.1.2.1.10.15.1.2.1.13.n
snmpFddiSMTStatus Reporting	1.3.6.1.2.1.10.15.1.2.1.14.n
snmpFddiSMTECMState	1.3.6.1.2.1.10.15.1.2.1.15.n
snmpFddiSMTCFState	1.3.6.1.2.1.10.15.1.2.1.16.n
snmpFddiSMTHoldState	1.3.6.1.2.1.10.15.1.2.1.17.n
snmpFddiSMTRemote DisconnectFlag	1.3.6.1.2.1.10.15.1.2.1.18.n
snmpFddiSMTStationAction	1.3.6.1.2.1.10.15.1.2.1.19.n
snmpFddiSMTStationId	1.3.6.1.2.1.10.15.1.2.1.2.n
snmpFddiSMTOpVersionId	1.3.6.1.2.1.10.15.1.2.1.3.n
snmpFddiSMTHiVersionId	1.3.6.1.2.1.10.15.1.2.1.4.n
snmpFddiSMTLoVersionId	1.3.6.1.2.1.10.15.1.2.1.5.n
snmpFddiSMTMACCt	1.3.6.1.2.1.10.15.1.2.1.6.n
snmpFddiSMTNonMasterCt	1.3.6.1.2.1.10.15.1.2.1.7.n
snmpFddiSMTMasterCt	1.3.6.1.2.1.10.15.1.2.1.8.n
snmpFddiSMTPaths Available	1.3.6.1.2.1.10.15.1.2.1.9.n
snmpFddiMAC	1.3.6.1.2.1.10.15.2
snmpFddiMACNumber	1.3.6.1.2.1.10.15.2.1.0
snmpFddiMACTable	1.3.6.1.2.1.10.15.2.2
snmpFddiMACEntry	1.3.6.1.2.1.10.15.2.2.1

(continued on next page)

Table B–5 (Cont.) FDDI MIB Object Identifiers

Object	ID
snmpFddiMACSMTIndex	1.3.6.1.2.1.10.15.2.2.1.1.n
snmpFddiMACDupAddrTest	1.3.6.1.2.1.10.15.2.2.1.10.n
snmpFddiMACPaths Requested	1.3.6.1.2.1.10.15.2.2.1.11.n
snmpFddiMACDownstream PORTType	1.3.6.1.2.1.10.15.2.2.1.12.n
snmpFddiMACSMTAddress	1.3.6.1.2.1.10.15.2.2.1.13.n
snmpFddiMACTReq	1.3.6.1.2.1.10.15.2.2.1.14.n
snmpFddiMACTNeg	1.3.6.1.2.1.10.15.2.2.1.15.n
snmpFddiMACTMax	1.3.6.1.2.1.10.15.2.2.1.16.n
snmpFddiMACTvxValue	1.3.6.1.2.1.10.15.2.2.1.17.n
snmpFddiMACTMin	1.3.6.1.2.1.10.15.2.2.1.18.n
snmpFddiMACCurrent FrameStatus	1.3.6.1.2.1.10.15.2.2.1.19.n
snmpFddiMACIndex	1.3.6.1.2.1.10.15.2.2.1.2.n
snmpFddiMACFrameCts	1.3.6.1.2.1.10.15.2.2.1.20.n
snmpFddiMACErrorCts	1.3.6.1.2.1.10.15.2.2.1.21.n
snmpFddiMACLostCts	1.3.6.1.2.1.10.15.2.2.1.22.n
snmpFddiMACFrameError Threshold	1.3.6.1.2.1.10.15.2.2.1.23.n
snmpFddiMACFrameError Ratio	1.3.6.1.2.1.10.15.2.2.1.24.n
snmpFddiMACRMTState	1.3.6.1.2.1.10.15.2.2.1.25.n
snmpFddiMACDaFlag	1.3.6.1.2.1.10.15.2.2.1.26.n
snmpFddiMACUnaDaFlag	1.3.6.1.2.1.10.15.2.2.1.27.n
snmpFddiMACFrame Condition	1.3.6.1.2.1.10.15.2.2.1.28.n
snmpFddiMACChipSet	1.3.6.1.2.1.10.15.2.2.1.29.n
snmpFddiMACFrame StatusCapabilities	1.3.6.1.2.1.10.15.2.2.1.3.n
snmpFddiMACAction	1.3.6.1.2.1.10.15.2.2.1.30.n
snmpFddiMACTMax GreatestLowerBound	1.3.6.1.2.1.10.15.2.2.1.4.n

(continued on next page)

Table B–5 (Cont.) FDDI MIB Object Identifiers

Object	ID
snmpFddiMACTVX GreatestLowerBound	1.3.6.1.2.1.10.15.2.2.1.5.n
snmpFddiMACPaths Available	1.3.6.1.2.1.10.15.2.2.1.6.n
snmpFddiMACCurrentPath	1.3.6.1.2.1.10.15.2.2.1.7.n
snmpFddiMACUpstreamNbr	1.3.6.1.2.1.10.15.2.2.1.8.n
snmpFddiMACOld UpstreamNbr	1.3.6.1.2.1.10.15.2.2.1.9.n
snmpFddiPORT	1.3.6.1.2.1.10.15.4
snmpFddiPORTNumber	1.3.6.1.2.1.10.15.4.1.0
snmpFddiPORTTable	1.3.6.1.2.1.10.15.4.2
snmpFddiPORTEntry	1.3.6.1.2.1.10.15.4.2.1
snmpFddiPORTSMTIndex	1.3.6.1.2.1.10.15.4.2.1.1.n
snmpFddiPORTAvailable Paths	1.3.6.1.2.1.10.15.4.2.1.10.n
snmpFddiPORTMACLoop Time	1.3.6.1.2.1.10.15.4.2.1.11.n
snmpFddiPORTTBMax	1.3.6.1.2.1.10.15.4.2.1.12.n
snmpFddiPORTBSFlag	1.3.6.1.2.1.10.15.4.2.1.13.n
snmpFddiPORTLCTFailCts	1.3.6.1.2.1.10.15.4.2.1.14.n
snmpFddiPORTLerEstimate	1.3.6.1.2.1.10.15.4.2.1.15.n
snmpFddiPORTLem RejectCts	1.3.6.1.2.1.10.15.4.2.1.16.n
snmpFddiPORTLemCts	1.3.6.1.2.1.10.15.4.2.1.17.n
snmpFddiPORTLerCutoff	1.3.6.1.2.1.10.15.4.2.1.18.n
snmpFddiPORTLerAlarm	1.3.6.1.2.1.10.15.4.2.1.19.n
snmpFddiPORTIndex	1.3.6.1.2.1.10.15.4.2.1.2.n
snmpFddiPORTConnectState	1.3.6.1.2.1.10.15.4.2.1.20.n
snmpFddiPORTPCMState	1.3.6.1.2.1.10.15.4.2.1.21.n
snmpFddiPORTPCWithhold	1.3.6.1.2.1.10.15.4.2.1.22.n
snmpFddiPORTLerCondition	1.3.6.1.2.1.10.15.4.2.1.23.n
snmpFddiPORTChipSet	1.3.6.1.2.1.10.15.4.2.1.24.n
snmpFddiPORTAction	1.3.6.1.2.1.10.15.4.2.1.25.n

(continued on next page)

Table B–5 (Cont.) FDDI MIB Object Identifiers

Object	ID
snmpFddiPORTPCType	1.3.6.1.2.1.10.15.4.2.1.3.n
snmpFddiPORTPCNeighbor	1.3.6.1.2.1.10.15.4.2.1.4.n
snmpFddiPORTConnection Policies	1.3.6.1.2.1.10.15.4.2.1.5.n
snmpFddiPORTRemote MACIndicated	1.3.6.1.2.1.10.15.4.2.1.6.n
snmpFddiPORTCEState	1.3.6.1.2.1.10.15.4.2.1.7.n
snmpFddiPORTPaths Requested	1.3.6.1.2.1.10.15.4.2.1.8.n
snmpFddiPORTMAC Placement	1.3.6.1.2.1.10.15.4.2.1.9.n
snmpFddiATTACHMENT	1.3.6.1.2.1.10.15.5
snmpFddiATTACHMENT Number	1.3.6.1.2.1.10.15.5.1.0
snmpFddiATTACHMENT Table	1.3.6.1.2.1.10.15.5.2
snmpFddiATTACHMENT Entry	1.3.6.1.2.1.10.15.5.2.1
snmpFddiATTACHMENT SMTIndex	1.3.6.1.2.1.10.15.5.2.1.1.n
snmpFddiATTACHMENT Index	1.3.6.1.2.1.10.15.5.2.1.2.n
snmpFddiATTACHMENT Class	1.3.6.1.2.1.10.15.5.2.1.3.n
snmpFddiATTACHMENT OpticalBypassPresent	1.3.6.1.2.1.10.15.5.2.1.4.n
snmpFddiATTACHMENT IMaxExpiration	1.3.6.1.2.1.10.15.5.2.1.5.n
snmpFddiATTACHMENT InsertedStatus	1.3.6.1.2.1.10.15.5.2.1.6.n
snmpFddiATTACHMENT InsertPolicy	1.3.6.1.2.1.10.15.5.2.1.7.n
snmpFddiChipSets	1.3.6.1.2.1.10.15.6
snmpFddiPHYChipSets	1.3.6.1.2.1.10.15.6.1
snmpFddiMACChipSets	1.3.6.1.2.1.10.15.6.2
snmpFddiPHYMACChipSets	1.3.6.1.2.1.10.15.6.3

Glossary of GIGAswitch FDDI Terms

Address Resolution Protocol

See ARP.

agent

In the client-server model, the part of the system that prepares and exchanges information on behalf of a client or server application.

alarm

A message sent to operator terminals that are enabled or defined by management software. Alarms are set using the network management station (NMS). *See also NMS.*

American National Standards Institute

See ANSI.

ANSI

American National Standards Institute. A national standards organization with members from computer manufacturers and users in the United States. It is the U.S. member body of ISO and is involved with the development of standards around the OSI Reference Model. ANSI proposes, compiles, and publishes standards for programming languages, databases, telecommunications, and other products.

ARP

Address Resolution Protocol. A protocol that maps a high-level Internet address with a low-level physical hardware address. Limited to networks that support hardware broadcast.

backup

A network device or circuit that is used if the primary device or circuit becomes unavailable. The spanning tree algorithm can put bridges or network branches in backup mode if they are redundant with others and might create loops in the network. *See also spanning tree.*

BOOTP

Boot protocol. A protocol that determines a diskless host Internet address at startup, so that the host can operate in an Internet network. *See also protocol.*

BPDU

An IEEE 802.1d Bridge Protocol Data Unit.

broadcast

Simultaneous transmission of data to more than one destination in a network, so that all broadcast addresses receive the same message.

CBS

Crossbar switch. The switching module that forms the heart of the GIGAswitch FDDI System.

community name

SNMP password for primitive security. *See also SNMP.*

crossbar switch

See CBS.

cutthrough

A proces that enables the GIGAswitch FDDI system to start forwarding a packet out of a port before the entire packet is received. Inbound cutthrough begins packet transmission through the crossbar switch before it is fully received from the inbound port. Outbound cutthrough begins packet transmission on the outbound port before it is fully received from the crossbar switch.

DA

Destination Address. A unique network address identifying a target system. For filter purposes, this is the 48-bit MAC address. Packets are filtered based on the destination address of the packet.

destination address

See DA.

DAS

Dual Attachment Station. An FDDI station that offers two connections to the dual counter-rotating ring. *See also FDDI.*

dotted decimal notation

The representation for a 32-bit integer that consists of four 8-bit numbers written in base 10 with decimals separating them. This is used to represent IP addresses on the Internet.

dual attachment station

See DAS.

dual homing

An FDDI method of cabling concentrators and stations that enables an alternate or backup path to the ring if the primary connection fails. *See also FDDI.*

FEU

Front end unit. Power supply for the GIGAswitch FDDI System.

FDDI

Fiber Distributed Data Interface. A set of ANSI/ISO standards that define a high-bandwidth (100 megabits per second), general-purpose LAN connection between computers and peripheral equipment in a timed-token passing, dual ring of trees configuration.

Fiber Distributed Data Interface

See FDDI.

FGL-2

Fiber GIGAswitch FDDI Linecard, 2-port.

FGL-4

Fiber GIGAswitch FDDI Linecard, 4-port. Can configure only as a single attachment station, it cannot be configured as a dual attachment station.

filtering

The process where a bridge evaluates incoming messages and selects those it needs to process, and those which it blocks from delivery. Filters can be set using management station commands.

forwarding

The ability of a bridge or router to accept messages from one local area network (LAN) segment and retransmit those messages to another LAN segment. *See also LAN.*

frame

A data transmission unit containing data or control information, address information, and a frame check sequence.

front-end unit

See FEU.

full-duplex

Pertaining to a type of data communications system capable of providing simultaneous, independent transmission and reception in both directions.

get

An SNMP request for data command. *See also SNMP.*

get-next

An SNMP command that gets the next data item in the MIB object tree. *See also SNMP, MIB.*

hotswap

The ability to remove and insert a component without powering down the GIGAswitch FDDI system. This procedure does not interrupt normal operation.

in-band management

A technique for carrying control signals within the same bandwidth as data being carried. In-band management for the GIGAswitch FDDI system is performed using a network management station (NMS). *See also NMS*.

Internal Protocol

See IP.

IP

Internet Protocol. The network layer protocol for the Internet protocol suite that provides the basis for the connectionless, best-effort packet delivery service. IP includes the Internet Control Message Protocol (ICMP) as an integral part. The Internet protocol suite is referred to as TCP/IP because IP is one of the two most fundamental protocols.

LAN

Local area network. A self-contained group of computers and communications devices (such as modems, routers, servers, and repeaters) that offer a high-speed, reliable communications channel. LANs span a limited distance, such as a building or cluster of buildings, but can be connected to wide area networks (WANs) with bridges or routers.

learning

The process by which a bridge discovers and remembers which ports network devices are connected to.

local area network

See LAN.

Maintenance Operation Protocol

See MOP.

Management Information Base

See MIB.

Management Station for UNIX

See MSU.

MIB

Management Information Base. A collection of objects that can be accessed with a network management protocol.

MMF

Multi mode fiber. Used in FDDI networks to support network station connections up to 2 kilometers

MOP

Maintenance Operations Protocol. A network management protocol within DECnet software that handles tasks such as downline loading, upline dumping, and circuit testing.

multicast

A special form of broadcast transmission where copies of the packet are only delivered to a subset of all destinations.

network management station

See NMS.

NMS

Network management station. The system responsible for managing a network. The NMS talks to network management agents, which reside in the managed nodes, using a network management protocol (such as SNMP). *See also SNMP*.

OBM

Out-of-band management. In network management, a technique for carrying control signals over a separate channel rather than within the main signal bandwidth. Out of band management for the GIGAswitch FDDI system is performed with a local terminal connected directly to the system with an RS-232 cable.

out-of-band management

See OBM.

PDU

Protocol data unit. The data units (messages or blocks of data) passed between peer entities on different open systems. PDUs consist of both Protocol Control Information (PCI) and user data.

Physical media device

See PMD.

PAID

Process identification. A binary value that uniquely identifies a process. Each process has a process identification and a process name.

PID

Protocol ID (*not process ID*).

PM

Presentation module. An interaction method for use with DECmcc.

PMD

Physical layer media dependent. The GIGAswitch FDDI system supports two types of PMDs: multimode fiber and single mode fiber.

POLYCENTER

POLYCENTER network management software monitors, controls, and tests entities in the DECnet, DECnet/OSI, and multivendor distributed environment. The GIGAswitch FDDI system can be managed by the POLYCENTER Network Manager (formerly DECmcc) with the SNMP access module, or by the POLYCENTER SNMP Manager (formerly DECmcc Management Station for ULTRIX). *See also SNMP*.

port

An individual connector on the GIGAswitch FDDI system that connects a LAN to the GIGAswitch FDDI system. *See also LAN*.

presentation Module

See PM.

privileged port

A port that can perform SNMP set operations on a secure GIGAswitch FDDI system. Privileged ports are defined by network management. *See also SNMP*.

process identification

See PI.

protocol

A set of rules for the implementation of a network communication system. Protocols cover options such as signaling methods, coding, packaging of messages, and methods of preventing and correcting errors.

protocol data unit

See PDU.

PSA

Power system assembly.

PSC

Power system controller.

rate limiting

Limits imposed on multicast traffic and traffic with unknown destination addresses (DAs). This reduces the risk of overloading ports with traffic. *See also DA>*.

SA

Source address. The unique network address indicating the originator of a message.

Simple Network Managing Protocol

See SNMP.

SAP

Service access point. The point at which an entity provides a service to a user entity in the layer above it. The SAP is named according to the layer providing the services (transport services are provided at a transport SAP, or TSAP, at the top of the transport layer).

SAS

Single attachment station. An FDDI station that offers one S port for attachment to the FDDI ring, usually via a concentrator. *See also FDDI.*

SCP

Switch control processor.

service access point

See SAP.

SET

An SNMP command that can set (alter) an SNMP object. *See also SNMP.*

single attachment station

See SAS.

slot

A groove where a module or card can be installed.

SMF

Single Mode Fiber. Used in FDDI networks to support network station connections up to 40k.

SNAP

Subnetwork Access Protocol. Used in protocol ID PID filtering. *See also PID.*

SNMP

Simple network management protocol. A protocol for monitoring and controlling hosts, bridges, routers, and terminal servers on TCP/IP networks with network management applications, such as DECmcc.

source address

See SA.

spanning tree

The logical arrangement created by bridges in an extended LAN in which all LANs are connected and there are no loops. *See also LAN*.

Subnet mask

Address mask. A bit mask used to select bits from an Internet address for subnet addressing. The mask is 32 bits long and selects the network portion of the IP address and one or more bits of the local portion.

TFTP

Trivial file transfer protocol. An Internet facility for transferring electronic files in a TCP/IP environment. TFTP allows authorized users to transfer files over the network.

transparent bridging

The IEEE 802.1d bridging scheme used to interconnect LANs based upon a spanning tree algorithm. The bridge provides all necessary functionality, including address learning and address filtering. Transparent bridging is protocol independent, performs automatic learning and forwarding, and ensures a loop-free topology in a large network, as only one active bridge connects any two LANs. *See also LAN, learning, spanning tree.*

trap

An unsolicited SNMP message sent by an SNMP manageable device to one or more network management stations (NMS). *See also NMS, SNMP*.

Trivial File Transfer Protocol

See TFTP.

WAN

Wide area network. Two or more standard or extended LANs that are joined by routers, gateways, or packet-switched interface (PST) software.

Wide Area Network

See WAN.

UTP

Unshielded twisted pair. Used in networks to support network station connections up to 100 meters.