



# Simple Network Management Protocol Reference Guide

---

Sun Storage Fibre Channel Switch 5802

Firmware Version 7.4

Sun Microsystems, Inc.  
[www.sun.com](http://www.sun.com)

Part No. 820-4974-10  
September 2008, Revision A

Submit comments about this document at: <http://www.sun.com/hwdocs/feedback>

Copyright 2008 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

This document and the product to which it pertains are distributed under licenses restricting their use, copying, distribution, and decompilation. No part of the product or of this document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Enterprise Fabric Suite is a trademark of QLogic Corporation.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, AnswerBook2, docs.sun.com, StorageTek, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc., or its subsidiaries, in the U.S. and in other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and in other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

---

Copyright 2008 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, États-Unis. Tous droits réservés.

Sun Microsystems, Inc. possède les droits de propriété intellectuelle relatifs à la technologie décrite dans ce document. En particulier, et sans limitation, ces droits de propriété intellectuelle peuvent inclure un ou plusieurs brevets américains listés sur le site <http://www.sun.com/patents>, un ou les plusieurs brevets supplémentaires ainsi que les demandes de brevet en attente aux les États-Unis et dans d'autres pays.

Ce document et le produit auquel il se rapporte sont protégés par un copyright et distribués sous licences, celles-ci en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Tout logiciel tiers, sa technologie relative aux polices de caractères, comprise, est protégé par un copyright et licencié par des fournisseurs de Sun.

Enterprise Fabric Suite est de marque de fabrique de QLogic Corporation.

Des parties de ce produit peuvent dériver des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux États-Unis et dans d'autres pays, licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Java, AnswerBook2, docs.sun.com, StorageTek, et Solaris sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc., ou ses filiales, aux États-Unis et dans d'autres pays.

Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux États-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface utilisateur graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox dans la recherche et le développement du concept des interfaces utilisateur visuelles ou graphiques pour l'industrie informatique. Sun détient une licence non exclusive de Xerox sur l'interface utilisateur graphique Xerox, cette licence couvrant également les licenciés de Sun implémentant les interfaces utilisateur graphiques OPEN LOOK et se conforment en outre aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DÉCLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES DANS LA LIMITE DE LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE À LA QUALITÉ MARCHANDE, À L'APTITUDE À UNE UTILISATION PARTICULIÈRE OU À L'ABSENCE DE CONTREFAÇON.



# Contents

---

## **Preface   vii**

### **1.   SNMP Overview   1**

SNMP Interface Objectives   1

Manager and Agent   2

Traps   3

Management Information Bases   3

User Datagram Protocol   4

Numbering System Conventions   4

### **2.   Configuring a Switch   5**

System Specifications and Requirements   5

Configuring a Switch Using the Telnet Command Line Interface   6

Configuring a Switch Using the Enterprise Fabric Suite 2007 application   9

### **3.   MIB-II Objects   11**

Groups in MIB-II   11

System Group   12

The Interfaces Group   16

The Interfaces Table   17

The Address Translation Group   26

The IP Group	28
The IP Address Table	37
The IP Routing Table	39
The IP Address Translation Table	46
Additional IP Objects	49
The ICMP Group	49
The TCP Group	60
The TCP Connection Table	65
Additional TCP Objects	68
The UDP Group	69
The UDP Listener Table	70
The EGP Group	71
The EGP Neighbor Table	73
The Transmission Group	80
The SNMP Group	80

#### **4. Fibre Alliance MIB Objects 93**

FA MIB Definitions	93
Connectivity Unit Group	95
Connectivity Table	98
Revision Table	117
Sensor Table	120
Port Table	126
Event Table	144
Link Table	150
Zone Table	158
Zoning Alias Table	164
Port Statistics Table	168
Simple Name Server Table	196

	Platform Table	205
	Trap Table	211
	Related Traps	216
<b>5.</b>	<b>Fabric Element MIB Objects</b>	<b>221</b>
	Fibre Channel FE MIB Definitions	221
	Configuration Group	222
	Module Table	224
	FxPort Configuration Table	228
	The Status Group	235
	FxPort Physical Level Table	238
	Fx Port Fabric Login Table	241
	The Error Group	248
	Accounting Groups	254
	Class 2 Accounting Table	259
	Class 3 Accounting Table	263
	Capability Group	266
<b>6.</b>	<b>Custom MIB Objects</b>	<b>275</b>
	Custom MIB Definitions	275
	Related Traps	277
<b>7.</b>	<b>Firmware Download MIB Objects</b>	<b>279</b>
	Firmware Download MIB Definitions	279
	<b>Glossary</b>	<b>285</b>
	<b>Index</b>	<b>291</b>



# Preface

---

This guide describes the support for Simple Network Management Protocol (SNMP) and how to use SNMP to manage and monitor Fibre Channel switches. This guide is intended for users responsible for the support of SNMP and Fibre Channel switch configurations.

---

## How This Book Is Organized

- [Chapter 1](#) provides an overview of SNMP objectives, managers and agents, traps, Management Information Bases (MIB), and User Datagram Protocol.
- [Chapter 2](#) describes how to configure a switch using Telnet and the Enterprise Fabric Suite 2007™ application for Sun FC switches and directors.
- [Chapter 3](#) describes the Management Information Bases (MIB-II).
- [Chapter 4](#) describes the Fibre Alliance - Management Information Bases (FA-MIB version 4.0).
- [Chapter 5](#) describes the Fabric Element - Management Information Bases (FE-MIB).
- [Chapter 6](#) describes the Custom Management Information Bases for the Sun Storage Fibre Channel Switch 5802.
- [Chapter 7](#) describes the Firmware Download Management Information Bases (FD-MIB).

---

# Typographic Conventions

Typeface	Meaning	Examples
AaBbCc123	The names of commands, files, and directories; on-screen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. % You have mail.
<b>AaBbCc123</b>	What you type, when contrasted with on-screen computer output	% <b>su</b> Password:
<i>AaBbCc123</i>	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the <i>User's Guide</i> . These are called <i>class</i> options. You <i>must</i> be superuser to do this. To delete a file, type <code>rm filename</code> .

---

**Note** – Characters display differently depending on browser settings. If characters do not display correctly, change the character encoding in your browser to Unicode UTF-8.

---

---

## Related Documentation

The following table lists the documentation for this product. The online documentation is available at:

<http://docs.sun.com/app/docs/prod/switch.dir#hic>

Application	Title	Part Number	Format	Location
Regulatory and safety information	<i>Sun Storage Regulatory and Safety Compliance Manual</i>	820-5506-xx	PDF	Online
Hardware and software requirements	<i>Sun Storage Fibre Channel Switch 5802 Hardware Release Notes</i>	820-5539-xx	PDF	Online
Initial switch installation	<i>Sun Storage Fibre Channel Switch 5802 Setup</i>	820-4950-xx	Printed PDF	Shipping kit, Online



Application	Title	Part Number	Format	Location
Install the switch	<i>Sun Storage Fibre Channel Switch 5802 Installation Guide</i>	820-4969-xx	PDF	Online
Manage the switch	<i>Sun Storage Fibre Channel Switch 5802 QuickTools User Guide</i>	820-4972-xx	PDF	Online
Manage the switch	<i>Enterprise Fabric Suite 2007 User Guide</i>	820-4966-xx	PDF	Enterprise Fabric Suite 2007 CD Online
Manage the switch	<i>Sun Storage Fibre Channel Switch 5802 Command Line Interface Guide</i>	820-4960-xx	PDF	Online
Command line interface reference	<i>Command Line Interface Quick Reference Guide</i>	820-4962-xx	PDF	Online
Look up messages and correct problems	<i>Event Message Guide</i>	820-4971-xx	PDF	Online
Manage the switch	<i>CIM Agent Reference Guide</i>	820-4959-xx	PDF	Online

## Documentation, Support, and Training

Sun Function	URL
Documentation	<a href="http://www.sun.com/documentation/">http://www.sun.com/documentation/</a>
Support	<a href="http://www.sun.com/support/">http://www.sun.com/support/</a>
Training	<a href="http://www.sun.com/training/">http://www.sun.com/training/</a>
Service	<a href="http://www.sun.com/service/contacting/index.xml">http://www.sun.com/service/contacting/index.xml</a>

## Third-Party Web Sites

Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites

or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

---

## Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

<http://www.sun.com/hwdocs/feedback>

Please include the title and part number of your document with your feedback:

*Simple Network Management Protocol Reference Guide*, part number 820-4974-10.

## SNMP Overview

---

Simple Network Management Protocol is the protocol governing network management and monitoring of network devices. This Simple Network Management Protocol Reference Guide describes how to use SNMP to manage and monitor the Sun Storage Fibre Channel Switch 5802. Specifically, this guide describes the SNMP agent that resides on the switch.

The following topics are covered in this section:

- [SNMP Interface Objectives](#)
- [Manager and Agent](#)
- [Traps](#)
- [Management Information Bases \(MIBs\)](#)
- [User Datagram Protocol \(UDP\)](#)
- [Numbering System Conventions](#)

---

## SNMP Interface Objectives

The objectives of the SNMP Interface are as follows:

- Connect to the SNMP agent that resides on the switch using a management workstation.
- Support of *Fabric Element Management Information Bases* (FE-MIB) (rfc2837) and *Fibre Alliance Management Information Bases* (FA-MIB) draft.
- Support of version 1 and 2 traps.
- The SNMP agent supports SNMPv1 and SNMPv2c.

# Manager and Agent

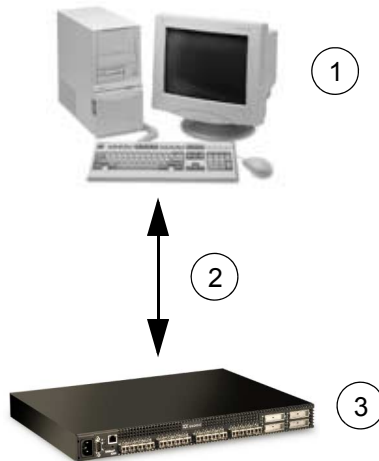
The two primary elements of SNMP are:

- **Manager** — the application that runs on the management workstation.
- **Agent** — the daemon application that runs on the switch.

The Manager is the application through which the network administrator performs network management functions. The SNMP agent is the direct interface on the switch for any SNMP manager connecting to the switch using the SNMP protocol, as shown in [FIGURE 1-1](#). The agent will be started by the script file(s) responsible for switch initialization when the switch powers up or when the switch is reset.

When an SNMP request arrives at the agent, the agent will compose a message and pass it on to Switch Management to process the message and provide a response to the agent. The agent then provides a response to the originator of the SNMP request. The SNMP agent does not have direct access to the internal database of the switch.

**FIGURE 1-1** SNMP Interface Architecture



**Figure Legend**

- 
- |   |   |
|---|---|
| 1 | Workstation with SNMP Manager Application       |
| 2 | Ethernet Connection                             |
| 3 | Switch with Agent and Common User Interface API |
-

---

# Traps

Traps are notification messages sent from the switch to a registered manager when a change of state occurs within the switch. A change of state can be an alarm condition or simply a configuration change.

The Fibre Alliance MIB defines a trap table configurable through SNMP. A trap table may have up to 5 entries, and can be configured using the SNMP Manager or Enterprise Fabric Suite 2007 graphical user interface. The same trap table information is available to both SNMP Manager and Enterprise Fabric Suite 2007.

A trap event is reported when the incoming error has a severity level less than or equal to the configured severity level. The trap event types and trap severity levels are listed in [TABLE 1-1](#). Refer to [TABLE 4-1](#) for information on specific traps.

**TABLE 1-1** Trap Severity Levels

Event Type	Severity Level
Unknown	1
Emergency	2
Alert	3
Critical	4
Error	5
Warning	6
Notify	7
Info	8
Debug	9
Mark	10

---

# Management Information Bases

Management information bases (MIBs) define the properties of the managed object within the device to be managed. Every managed device keeps a database of values for each definition written in the MIB. It is not the actual database itself; it is

implementation dependant. Definition of the MIB conforms to the Structure of Management Information (SMI) given in Request For Comment (RFC) 1155. The latest Internet MIB is given in RFC 1213, and is sometimes called MIB-II.

---

## User Datagram Protocol

The Sun Storage Fibre Channel Switch 5802 supports the following User Datagram Protocol (UDP) settings:

- Agents “listen” on UDP port 161.
  - Responses are sent back to the originating Network Management Station (NMS) port from a dynamic port, although many agents use port 161 also for this target.
  - The maximum SNMP message size is 65507 octets (maximum UDP message size).
  - The minimum receive packet size for SNMP implementations is 484 octets in length.
  - Agent and Network Monitoring Systems are responsible for determining error recovery.
- 

## Numbering System Conventions

The conventions for numbering systems in this guide are as follows:

- Decimal = 101
- Hexadecimal = 0x101
- Binary = 101b

## Configuring a Switch

---

This section describes how to configure a switch to support SNMP. The following topics are covered:

- [System Specifications and Requirements](#)
  - [Configuring a Switch Using the Telnet Command Line Interface](#)
  - [Configuring a Switch Using the Enterprise Fabric Suite 2007 application](#)
- 

### System Specifications and Requirements

- Sun Storage Fibre Channel Switch 5802 supports SNMPv1 and SNMPv2c.
- Version 1 and 2 traps are supported.
- Hardware — one out-of-band Ethernet connection is required.
- Software — one switch management software application allows you to:
  - Monitor and control the switch.
  - Read, write, and receive trap information, if supported.
- Ports on the switch reserved for SNMP:
  - Port 161 is not configurable, and is used for the standard SNMP commands.
  - Port 162 is configurable and is the default port used for traps.
- One or more in-band switches can be managed by an out-of-band switch acting as a proxy switch.
- A Sun Storage Fibre Channel Switch 5802 can only act as a proxy for other Sun Storage switches.
- The Sun Storage Fibre Channel Switch 5802 proxy capability can be disabled.

---

# Configuring a Switch Using the Telnet Command Line Interface

The Telnet command line interface offers a convenient way to change SNMP parameters. SNMP parameter defaults are preset during manufacturing. For security purposes, these default values should be changed. For specific information about SNMP parameters, refer to the SNMP Configuration section in the corresponding *Command Line Interface Guide*. To configure a switch using the command line interface, do the following.

Press the Enter key to accept the default value for each parameter.

```
SB5600.116.50 (admin) #> set setup snmp
A list of attributes with formatting and current values will
follow.
Enter a new value or simply press the ENTER key to accept the
current value.
If you wish to terminate this process before reaching the end of
the attributes for the category being processed, press 'q' or 'Q'
and the ENTER key to do so.
If you wish to terminate the configuration process completely,
press 'qq' or 'QQ' and the ENTER key to do so.

SNMP System Configuration - may optionally use 'set setup snmp
common' command.

Current Values:
  SnmpEnabled      True
  Contact          <sysContact undefined>
  Location         <solicitation undefined>
  Read Community   public
  Write Community  private
  AuthFailureTrap  False
  ProxyEnabled     True
  SNMPv3Enabled    False

New Value (press ENTER to not specify value, 'q' to quit):
  SnmpEnabled      (True / False)      :
  Contact          (string, max=64 chars):
  Location         (string, max=64 chars):
  ReadCommunity    (string, max=32 chars):
  WriteCommunity   (string, max=32 chars):
  AuthFailureTrap  (True / False)      :
  ProxyEnabled     (True / False)      :
  SNMPv3Enabled    (True / False)      :
```



SNMP Trap 1 Configuration - may optionally use 'set setup snmp trap 1' command.

Current Values:

```
Trap1Enabled      False
Trap1Address      10.0.0.254
Trap1Port         5001
Trap1Severity     info
Trap1Version      2
Trap1Community    public
```

New Value (press ENTER to not specify value, 'q' to quit):

```
Trap1Enabled      (True / False)      :
Trap1Address      (hostname, IPv4, or IPv6 Address):
Trap1Port         (decimal value, 1-65535)      :
Trap1Severity     (select a severity level)
                  1=unknown      6=warning
                  2=emergency    7=notify
                  3=alert        8=info
                  4=critical     9=debug
                  5=error        10=mark         :
Trap1Version      (1 / 2)              :
Trap1Community    (string, max=32 chars)      :
```

SNMP Trap 2 Configuration - may optionally use 'set setup snmp trap 2' command.

Current Values:

```
Trap2Enabled      False
Trap2Address      10.20.43.231
Trap2Port         162
Trap2Severity     info
Trap2Version      2
Trap2Community    public
```

New Value (press ENTER to not specify value, 'q' to quit):

```
Trap2Enabled      (True / False)      :
Trap2Address      (hostname, IPv4, or IPv6 Address):
Trap2Port         (decimal value, 1-65535)      :
Trap2Severity     (select a severity level)
                  1=unknown      6=warning
                  2=emergency    7=notify
                  3=alert        8=info
                  4=critical     9=debug
                  5=error        10=mark         :
Trap2Version      (1 / 2)              :
Trap2Community    (string, max=32 chars)      :
```

SNMP Trap 3 Configuration - may optionally use 'set setup snmp trap 3' command.

Current Values:

```
Trap3Enabled      False
Trap3Address      10.20.33.231
Trap3Port         162
Trap3Severity     warning
Trap3Version      2
Trap3Community    public
```

New Value (press ENTER to not specify value, 'q' to quit):

```
Trap3Enabled      (True / False)      :
Trap3Address      (hostname, IPv4, or IPv6 Address):
Trap3Port         (decimal value, 1-65535)      :
Trap3Severity     (select a severity level)
                  1=unknown      6=warning
                  2=emergency    7=notify
                  3=alert        8=info
                  4=critical      9=debug
                  5=error        10=mark        :
Trap3Version      (1 / 2)              :
Trap3Community    (string, max=32 chars)      :
```

SNMP Trap 4 Configuration - may optionally use 'set setup snmp trap 4' command.

Current Values:

```
Trap4Enabled      False
Trap4Address      0.0.0.0
Trap4Port         162
Trap4Severity     warning
Trap4Version      2
Trap4Community    public
```

New Value (press ENTER to not specify value, 'q' to quit):

```
Trap4Enabled      (True / False)      :
Trap4Address      (hostname, IPv4, or IPv6 Address):
Trap4Port         (decimal value, 1-65535)      :
Trap4Severity     (select a severity level)
                  1=unknown      6=warning
                  2=emergency    7=notify
                  3=alert        8=info
                  4=critical      9=debug
                  5=error        10=mark        :
Trap4Version      (1 / 2)              :
Trap4Community    (string, max=32 chars)      :
```

```
SNMP Trap 5 Configuration - may optionally use 'set setup snmp
trap 5' command.
```

```
Current Values:
```

```
Trap5Enabled      False
Trap5Address      0.0.0.0
Trap5Port         162
Trap5Severity     warning
Trap5Version      2
Trap5Community    public
```

```
New Value (press ENTER to not specify value, 'q' to quit):
```

```
Trap5Enabled      (True / False)                :
Trap5Address      (hostname, IPv4, or IPv6 Address):
Trap5Port         (decimal value, 1-65535)       :
Trap5Severity     (select a severity level)
                  1=unknown      6=warning
                  2=emergency    7=notify
                  3=alert        8=info
                  4=critical     9=debug
                  5=error        10=mark          :
Trap5Version      (1 / 2)                        :
Trap5Community    (string, max=32 chars)         :
```

```
Do you want to save and activate this snmp setup? (y/n): [n]
```

```
SNMP setup NEITHER saved NOR activated.
```

```
SB5600.116.50 (admin) #>
```

---

## Configuring a Switch Using the Enterprise Fabric Suite 2007 application

To configure a switch using Enterprise Fabric Suite 2007, use the SNMP Properties, Switch Properties, and Network Properties dialogs. For specific information, refer to the corresponding *Enterprise Fabric Suite User Guide*.



## MIB-II Objects

---

This section covers the implementation details for the MIB-II on the switch. A MIB defines the properties of the managed object within the device to be managed. Every managed device keeps a database of values for each definition written in the MIB. It is not the actual database itself; it is implementation dependant. Definition of the MIB conforms to the SMI given in RFC 1155. The latest Internet MIB is given in RFC 1213, and is sometimes called MIB-II.

- [Groups in MIB-II](#)
- [System Group](#)
- [The Interfaces Group](#)
- [The Address Translation Group](#)
- [The IP Group](#)
- [The ICMP Group](#)
- [The TCP Group](#)
- [The UDP Group](#)
- [The EGP Group](#)
- [The Transmission Group](#)
- [The SNMP Group](#)

---

## Groups in MIB-II

Refer the [TABLE 3-1](#) for the syntax for MIB-II Groups.

**TABLE 3-1** MIB-II Groups

Group	Syntax
system	OBJECT IDENTIFIER::= { mib-2 1 }
interfaces	OBJECT IDENTIFIER::= {mib-2 2}
at	OBJECT IDENTIFIER::= {mib-2 3}
I	OBJECT IDENTIFIER::= { mib-2 4}
icmp	OBJECT IDENTIFIER::= {mib-2 5}
tcp	OBJECT IDENTIFIER::= {mib-2 6}
udp	OBJECT IDENTIFIER::= {mib-2 7}
snmp	OBJECT IDENTIFIER::= {mib-2 11}

# System Group

Implementation of the System group is mandatory for all systems. If an agent is not configured to have a value for any of these variables, a string of length 0 is returned.

## sysDescr (1.3.6.1.2.1.1.1)

A textual description of the entity. This value should include the full name and version identification of the system's hardware type, operating-system, and networking software. It is mandatory that this only contain printable American Standard Code for Information Interchange (ASCII) characters.

### Syntax

DisplayString (SIZE (0.255))

### Access

read-only

### Status

Mandatory

## Return Value

Sun Storage FC Switch 5802

## sysObjectID (1.3.6.1.2.1.1.2)

The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprise subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining 'what kind of box' is being managed. For example, if vendor 'Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its 'Fred Router'.

## Syntax

OBJECT IDENTIFIER

## Access

read-only

## Status

mandatory

## Return Value

1.3.6.1.4.1.42.2.209

## sysUpTime (1.3.6.1.2.1.1.3)

The time, in hundredths of a second, since the network management portion of the system was last re-initialized.

## Syntax

TimeTicks

## Access

read-only

## Status

mandatory

## Return Value

The time since the switch was powered on, or last reset (reset, hardreset, or hotreset) was executed. For example, 3 days 21 hours, 5 minutes, and 26.84 seconds. The value will roll over after approximately 497 days of continuous up time.

## sysContact (1.3.6.1.2.1.1.4)

The textual identification of the contact person for this managed Node, together with information on how to contact this person.

## Syntax

DisplayString (SIZE (0.255))

## Access

read-write

## Status

mandatory

## Return Value

The default is: <sysContact undefined>. The string size is limited to a maximum of 64.

## sysName (1.3.6.1.2.1.1.5)

An administratively-assigned name for this managed Node. By convention, this is the Node's fully-qualified domain name.

## Syntax

DisplayString (SIZE (0.255))

## Access



read-write

## Status

mandatory

## Return Value

Switch

## sysLocation (1.3.6.1.2.1.1.6)

The physical location of this Node, such as telephone closet and 3rd floor.

## Syntax

DisplayString (SIZE (0.255))

## Access

read-write

## Status

mandatory

## Return Value

The default is: <sysLocation undefined>. The string size is limited to a maximum of 64.

## sysServices (1.3.6.1.2.1.1.7)

A value that indicates the set of services that this entity primarily offers. The value is a sum. This sum initially takes the value zero. Then, for each layer L in the range 1 through 7 that this Node performs transactions for, 2 raised to (L - 1) is added to the sum. For example, a Node that performs primarily routing functions would have a value of 4 ( $2^{(3-1)}$ ). In contrast, a Node that is a host offering application services would have a value of 72 ( $2^{(4-1)} + 2^{(7-1)}$ ).

## Syntax

INTEGER (0.127)

#### Access

read-only

#### Status

mandatory

#### Return Value

The default is: 2

---

## The Interfaces Group

Implementation of the Interfaces group is mandatory for all systems.

### ifNumber (1.3.6.1.2.1.2.1)

The number of network interfaces (regardless of their current state) present on this system.

#### Syntax

INTEGER

#### Access

read-only

#### Status

mandatory

#### Return Value

The default is: 2

---

# The Interfaces Table

The Interfaces table contains information on the entity's interfaces. Each interface is thought of as being attached to a `subnet'. This term should not be confused with `subnet' which refers to an addressing partitioning scheme used in the Internet suite of protocols.

## ifIndex (1.3.6.1.2.1.2.2.1.1)

A unique value for each interface. Its value ranges between 1 and the value of ifNumber. The value for each interface must remain constant at least from one re-initialization of the entity's network management system to the next re-initialization.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

## ifDescr (1.3.6.1.2.1.2.2.1.2)

A textual string containing information about the interface. This string should include the name of the manufacturer, the product name, and the version of the hardware interface.

### Syntax

DisplayString (SIZE (0.255))

### Access

read-only

## Status

mandatory

## ifType (1.3.6.1.2.1.2.2.1.3)

The type of interface distinguished according to the physical/link protocol(s) immediately 'below' the network layer in the protocol stack.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## ifMtu (1.3.6.1.2.1.2.2.1.4)

The size of the largest datagram which can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## ifSpeed (1.3.6.1.2.1.2.2.1.5)

An estimate of the interface's current bandwidth in bits per second. For interfaces that do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth.

### Syntax

Gauge

### Access

read-only

### Status

mandatory

## ifPhysAddress (1.3.6.1.2.1.2.2.1.6)

The interface's address at the protocol layer immediately “below” the network layer in the protocol stack. For interfaces that do not have such an address, such as a serial line, this object should contain an octet string of zero length.

### Syntax

PhysAddress

### Access

read-only

### Status

mandatory

## ifAdminStatus (1.3.6.1.2.1.2.2.1.7)

The desired state of the interface. The testing(3) state indicates that no operational packets can be passed.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# ifOperStatus (1.3.6.1.2.1.2.2.1.8)

The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

# ifLastChange (1.3.6.1.2.1.2.2.1.9)

The value of `sysUpTime` at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.

## Syntax

TimeTicks

## Access

read-only

## Status

mandatory

# ifInOctets (1.3.6.1.2.1.2.2.1.10)

The total number of octets received on the interface, including framing characters.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# ifInUcastPkts (1.3.6.1.2.1.2.2.1.11)

The number of subnetwork-unicast packets delivered to a higher-layer protocol.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ifInNUcastPkts (1.3.6.1.2.1.2.2.1.12)

The number of non-unicast (that is, subnetwork- broadcast or subnetwork-multicast) packets delivered to a higher-layer protocol.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ifInDiscards (1.3.6.1.2.1.2.2.1.13)

The number of inbound packets that were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ifInErrors (1.3.6.1.2.1.2.2.1.14)

The number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.



## Syntax

Counter

## Access

read-only

## Status

mandatory

# ifInUnknownProtos (1.3.6.1.2.1.2.2.1.15)

The number of packets received from the interface that were discarded because of an unknown or unsupported protocol.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# ifOutOctets (1.3.6.1.2.1.2.2.1.16)

The total number of octets transmitted out of the interface, including framing characters.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ifOutUcastPkts (1.3.6.1.2.1.2.2.1.17)

The total number of packets that higher level protocols requested be transmitted to a subnetwork unicast address, including those that were discarded or not sent.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ifOutNUcastPkts (1.3.6.1.2.1.2.2.1.18)

The total number of packets that higher level protocols requested be transmitted to a non-unicast (subnetwork broadcast or subnetwork multicast) address, including those that were discarded or not sent.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ifOutDiscards (1.3.6.1.2.1.2.2.1.19)

The number of outbound packets that were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ifOutErrors (1.3.6.1.2.1.2.2.1.20)

The number of outbound packets that could not be transmitted because of errors.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ifOutQLen (1.3.6.1.2.1.2.2.1.21)

The length (in packets) of the output packet queue.

### Syntax

Gauge

## Access

read-only

## Status

mandatory

## ifSpecific (1.3.6.1.2.1.2.2.1.22)

A reference to MIB definitions specific to the particular media being used to realize the interface. For example, if the interface is realized by an Ethernet, then the value of this object refers to a document that defines objects specific to Ethernet. If this information is not present, its value should be set to the OBJECT IDENTIFIER {0 0}, which is a syntactically valid object identifier, and any conformant implementation of ASN.1 (Abstract Syntax Notation) and BER must be able to generate and recognize this value.

## Syntax

OBJECT IDENTIFIER

## Access

read-only

## Status

mandatory

---

# The Address Translation Group

Implementation of the Address Translation group is mandatory for all systems. However, this group is deprecated by MIB-II. That is, it is being included solely for compatibility with MIB-I Nodes, and will most likely be excluded from MIB-III Nodes. From MIB-II and onwards, each network protocol group contains its own address translation tables.

The Address Translation group contains one table which is the union across all interfaces of the translation tables for converting a `NetworkAddress` (for example, an IP address) into a subnetwork-specific address. For lack of a better term, this document refers to such a subnetwork-specific address as a 'physical' address.

Examples of such translation tables are for: broadcast media where ARP is in use, the translation table is equivalent to the ARP cache, or on an X.25 network where non-algorithmic translation to X.121 addresses is required. The translation table contains the `NetworkAddress` to X.121 address equivalences.

## atIfIndex (1.3.6.1.2.1.3.1.1.1)

The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.

### Syntax

INTEGER

### Access

read-write

### Status

deprecated

## atPhysAddress (1.3.6.1.2.1.3.1.1.2)

The media-dependent "physical" address. Setting this object to a null string (one of zero length) has the effect of invalidating the corresponding entry in the `atTable` object. That is, it effectively disassociates the interface identified with the entry from the mapping identified with the entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management workstations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant `atPhysAddress` object.

### Syntax

PhysAddress

## Access

read-write

## Status

deprecated

### atNetAddress (1.3.6.1.2.1.3.1.1.3)

The NetworkAddress corresponding to the media-dependent 'physical' address.

## Syntax

NetworkAddress

## Access

read-write

## Status

deprecated

---

## The IP Group

Implementation of the IP group is mandatory for all systems.

### ipForwarding (1.3.6.1.2.1.4.1)

The indication of whether this entity is acting as an IP Gateway with respect to the forwarding of datagrams received by, but not addressed to, this entity. IP Gateways forward datagrams; IP hosts do not (except those source-routed from the host).

For some managed Nodes, this object may take on only a subset of the values possible. Accordingly, it is appropriate for an agent to return a "badValue" response if a management station attempts to change this object to an inappropriate value.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Returns forwarding (1). Writes not supported.

## ipDefaultTTL (1.3.6.1.2.1.4.2)

The default value inserted into the `Time-To-Live` field of the IP header of datagrams originated at this entity whenever a TTL value is not supplied by the transport layer protocol.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Returns 64 (0x40). Writes not supported.

## ipInReceives (1.3.6.1.2.1.4.3)

The total number of input datagrams received from interfaces, including those received in error.

## Syntax

Counter

## Access

read-only

## Status

mandatory

### ipInHdrErrors (1.3.6.1.2.1.4.4)

The number of input datagrams discarded due to errors in their IP headers. These include bad checksums, version number mismatch, other format errors, time-to-live exceeded, and errors discovered in processing their IP options.

## Syntax

Counter

## Access

read-only

## Status

mandatory

### ipInAddrErrors (1.3.6.1.2.1.4.5)

The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (for example, 0.0.0.0) and addresses of unsupported Classes (for example, Class E). For entities which are not IP Gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.

## Syntax

Counter



## Access

read-only

## Status

mandatory

# ipForwDatagrams (1.3.6.1.2.1.4.6)

The number of input datagrams for which this entity was not their final IP destination. As a result, an attempt was made to find a route to forward them to that final destination. In entities that do not act as IP Gateways, this counter will include only those packets that were Source Routed from this entity, and the Source Route option processing was successful.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# ipInUnknownProtos (1.3.6.1.2.1.4.7)

The number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ipInDiscards (1.3.6.1.2.1.4.8)

The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (for example, for lack of buffer space). This counter does not include any datagrams discarded while awaiting reassembly.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ipInDelivers (1.3.6.1.2.1.4.9)

The total number of input datagrams successfully delivered to IP user protocols (including ICMP).

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ipOutRequests (1.3.6.1.2.1.4.10)

The total number of IP datagrams that local IP user protocols (including ICMP) supplied to IP in requests for transmission. This counter does not include any datagrams counted in ipForwDatagrams.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ipOutDiscards (1.3.6.1.2.1.4.11)

The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (for example, for lack of buffer space). This counter would include datagrams counted in ipForwDatagrams if any such packets met this (discretionary) discard criterion.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ipOutNoRoutes (1.3.6.1.2.1.4.12)

The number of IP datagrams discarded because no route could be found to transmit them to their destination. This counter includes any packets counted in `ipForwDatagrams` which meet this “no-route” criterion. This includes any datagrams that a host cannot route because all of its default gateways are down.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## ipReasmTimeout (1.3.6.1.2.1.4.13)

The maximum number of seconds which received fragments are held while they are awaiting reassembly at this entity.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

## ipReasmReqds (1.3.6.1.2.1.4.14)

The number of IP fragments received that needed to be reassembled at this entity.

### Syntax

Counter

Access

read-only

Status

mandatory

## ipReasmOKs (1.3.6.1.2.1.4.15)

The number of IP datagrams successfully reassembled.

Syntax

Counter

Access

read-only

Status

mandatory

## ipReasmFails (1.3.6.1.2.1.4.16)

The number of failures detected by the IP reassembly algorithm for example, timed out, errors). This is not necessarily a count of discarded IP fragments, since some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received.

Syntax

Counter

Access

read-only

## Status

mandatory

## ipFragOKs (1.3.6.1.2.1.4.17)

The number of IP datagrams that have been successfully fragmented at this entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ipFragFails (1.3.6.1.2.1.4.18)

The number of IP datagrams that have been discarded because they needed to be fragmented at this entity, but could not because their Don't Fragment flag was set.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## ipFragCreates (1.3.6.1.2.1.4.19)

The number of IP datagram fragments that have been generated as a result of fragmentation at this entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

---

# The IP Address Table

The IP address table contains this entity's IP addressing information.

## ipAdEntAddr (1.3.6.1.2.1.4.20.1.1)

The IP address to which this entry's addressing information pertains.

## Syntax

IpAddress

## Access

read-only

## Status

mandatory

## ipAdEntIfIndex (1.3.6.1.2.1.4.20.1.2)

The index value which uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

# ipAdEntNetMask (1.3.6.1.2.1.4.20.1.3)

The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.

## Syntax

IpAddress

## Access

read-only

## Status

mandatory

# ipAdEntBcastAddr (1.3.6.1.2.1.4.20.1.4)

The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface.

## Syntax

INTEGER

## Access



read-only

## Status

mandatory

## ipAdEntReasmMaxSize (1.3.6.1.2.1.4.20.1.5)

The size of the largest IP datagram which this entity can reassemble from incoming IP fragmented datagrams received on this interface.

## Syntax

INTEGER (0.65535)

## Access

read-only

## Status

mandatory

---

# The IP Routing Table

The IP routing table contains an entry for each route presently known to this entity.

## ipRouteDest (1.3.6.1.2.1.4.21.1.1)

The destination IP address of this route. An entry with a value of 0.0.0.0 is considered a default route. Multiple routes to a single destination can appear in the table, but access to such multiple entries is dependent on the table-access mechanisms defined by the network management protocol in use.

## Syntax

IpAddress

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# ipRouteIfIndex (1.3.6.1.2.1.4.21.1.2)

The index value which uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# ipRouteMetric1 (1.3.6.1.2.1.4.21.1.3)

The primary routing metric for this route. The semantics of this metric are determined by the routing protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.

## Syntax

INTEGER

#### Access

read-write

#### Status

mandatory

#### Return Value

Writes not supported.

### ipRouteMetric2 (1.3.6.1.2.1.4.21.1.4)

An alternate routing metric for this route. The semantics of this metric are determined by the routing protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to -1.

#### Syntax

INTEGER

#### Access

read-write

#### Status

mandatory

#### Return Value

Writes not supported.

### ipRouteMetric3 (1.3.6.1.2.1.4.21.1.5)

An alternate routing metric for this route. The semantics of this metric are determined by the routing protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to -1.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# ipRouteMetric4 (1.3.6.1.2.1.4.21.1.6)

An alternate routing metric for this route. The semantics of this metric are determined by the routing protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to -1.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

## ipRouteNextHop (1.3.6.1.2.1.4.21.1.7)

The IP address of the next hop of this route. In the case of a route bound to an interface which is realized from a broadcast media, the value of this field is the agent's IP address on that interface.

### Syntax

IpAddress

### Access

read-write

### Status

mandatory

### Return Value

Writes not supported.

## ipRouteType (1.3.6.1.2.1.4.21.1.8)

The type of route. The values `direct(3)` and `indirect(4)` refer to the notion of direct and indirect routing in the IP architecture. Setting this object to the value `invalid(2)` has the effect of invalidating the corresponding entry in the `ipRouteTable` object. That is, it effectively disassociates the destination identified with the entry from the route identified with the entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant `ipRouteType` object.

### Syntax

INTEGER

### Access

read-write

### Status

mandatory

## Return Value

Writes not supported.

## ipRouteProto (1.3.6.1.2.1.4.21.1.9)

The routing mechanism through which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## ipRouteAge (1.3.6.1.2.1.4.21.1.10)

The number of seconds since this route was last updated or otherwise determined to be correct. No semantics of 'too old' can be implied except through knowledge of the routing protocol by which the route was learned.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

## ipRouteMask (1.3.6.1.2.1.4.21.1.11)

Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the `ipRouteDest` field.

## Syntax

IpAddress

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

## ipRouteMetric5 (1.3.6.1.2.1.4.21.1.12)

An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to -1.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

## ipRouteInfo (1.3.6.1.2.1.4.21.1.13)

A reference to MIB definitions specific to the particular routing protocol which is responsible for this route, as determined by the value specified in the route's `ipRouteProto` value. If this information is not present, its value should be set to the OBJECT IDENTIFIER {0 0}, which is a syntactically valid object identifier. Any conformant implementation of ASN.1 and BER must be able to generate and recognize this value.

## Syntax

OBJECT IDENTIFIER

## Access

read-only

## Status

mandatory

---

# The IP Address Translation Table

The IP address translation table contain the `IpAddress` to 'physical' address equivalences. Some interfaces do not use translation tables for determining address equivalences (for example, DDN-X.25 has an algorithmic method). If all interfaces are of this type, then the Address Translation table is empty, that is, has zero entries.

## ipNetToMediaIfIndex (1.3.6.1.2.1.4.22.1.1)

The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.

## Syntax



INTEGER

Access

read-write

Status

mandatory

Return Value

Writes not supported.

## ipNetToMediaPhysAddress (1.3.6.1.2.1.4.22.1.2)

The media-dependent `physical' address.

Syntax

PhysAddress

Access

read-write

Status

mandatory

Return Value

Writes not supported.

## ipNetToMediaNetAddress (1.3.6.1.2.1.4.22.1.3)

The `IpAddress` corresponding to the media-dependent `physical' address.

Syntax

IpAddress

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# ipNetToMediaType (1.3.6.1.2.1.4.22.1.4)

The type of mapping. Setting this object to the value `invalid(2)` has the effect of invalidating the corresponding entry in the `ipNetToMediaTable`. That is, it effectively disassociates the interface identified with the entry from the mapping identified with the entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant `ipNetToMediaType` object.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

---

## Additional IP Objects

Following are the additional IP objects.

### ipRoutingDiscards (1.3.6.1.2.1.4.23)

The number of routing entries which were chosen to be discarded even though they are valid. One possible reason for discarding such an entry could be to free-up buffer space for other routing entries.

#### Syntax

Counter

#### Access

read-only

#### Status

mandatory

---

## The ICMP Group

Implementation of the ICMP group is mandatory for all systems.

### icmpInMsgs (1.3.6.1.2.1.5.1)

The total number of ICMP messages received by the entity. This counter includes all those counted by `icmpInErrors`.

#### Syntax

Counter

#### Access

read-only

## Status

mandatory

## icmpInErrors (1.3.6.1.2.1.5.2)

The number of ICMP messages received by the entity but were determined as having ICMP-specific errors (such as, bad ICMP checksums, bad length).

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpInDestUnreachs (1.3.6.1.2.1.5.3)

The number of ICMP Destination Unreachable messages received.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpInTimeExcds (1.3.6.1.2.1.5.4)

The number of ICMP Time Exceeded messages received.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpInParmProbs (1.3.6.1.2.1.5.5)

The number of ICMP Parameter Problem messages received.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpInSrcQuenchs (1.3.6.1.2.1.5.6)

The number of ICMP Source Quench messages received.

### Syntax

Counter

### Access

read-only

## Status

mandatory

## icmpInRedirects (1.3.6.1.2.1.5.7)

The number of ICMP Redirect messages received.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpInEchos (1.3.6.1.2.1.5.8)

The number of ICMP Echo (request) messages received.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpInEchoReps (1.3.6.1.2.1.5.9)

The number of ICMP Echo Reply messages received.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpInTimestamps (1.3.6.1.2.1.5.10)

The number of ICMP Timestamp (request) messages received.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpInTimestampReps (1.3.6.1.2.1.5.11)

The number of ICMP Timestamp Reply messages received.

### Syntax

Counter

### Access

read-only

## Status

mandatory

## icmpInAddrMasks (1.3.6.1.2.1.5.12)

The number of ICMP Address Mask Request messages received.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpInAddrMaskReps (1.3.6.1.2.1.5.13)

The number of ICMP Address Mask Reply messages received.

## Syntax

Counter

## Access

read-only

## Status

mandatory



## icmpOutMsgs (1.3.6.1.2.1.5.14)

The total number of ICMP messages which this entity attempted to send. This counter includes all those counted by `icmpOutErrors`.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutErrors (1.3.6.1.2.1.5.15)

The number of ICMP messages which this entity did not send due to problems discovered within ICMP, such as a lack of buffers. This value should not include errors discovered outside the ICMP layer such as the inability of IP to route the resultant datagram. In some implementations, there may be no types of errors which contribute to this counter's value.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutDestUnreachs (1.3.6.1.2.1.5.16)

The number of ICMP Destination Unreachable messages sent.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# icmpOutTimeExcds (1.3.6.1.2.1.5.17)

The number of ICMP Time Exceeded messages sent.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# icmpOutParmProbs (1.3.6.1.2.1.5.18)

The number of ICMP Parameter Problem messages sent.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## icmpOutSrcQuenchs (1.3.6.1.2.1.5.19)

The number of ICMP Source Quench messages sent.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutRedirects (1.3.6.1.2.1.5.20)

The number of ICMP Redirect messages sent. For a host, this object will always be zero, since hosts do not send redirects.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutEchos (1.3.6.1.2.1.5.21)

The number of ICMP Echo (request) messages sent.

### Syntax

Counter

Access

read-only

Status

mandatory

## icmpOutEchoReps (1.3.6.1.2.1.5.22)

The number of ICMP Echo Reply messages sent.

Syntax

Counter

Access

read-only

Status

mandatory

## icmpOutTimestamps (1.3.6.1.2.1.5.23)

The number of ICMP Timestamp (request) messages sent.

Syntax

Counter

Access

read-only

Status

mandatory

## icmpOutTimestampReps (1.3.6.1.2.1.5.24)

The number of ICMP Timestamp Reply messages sent.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutAddrMasks (1.3.6.1.2.1.5.25)

The number of ICMP Address Mask Request messages sent.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## icmpOutAddrMaskReps (1.3.6.1.2.1.5.26)

The number of ICMP Address Mask Reply messages sent.

### Syntax

Counter

### Access

read-only

## Status

mandatory

---

# The TCP Group

Implementation of the TCP group is mandatory for all systems that implement the TCP. Instances of object types that represent information about a particular TCP connection are transient; they persist only as long as the connection in question.

## tcpRtoAlgorithm (1.3.6.1.2.1.6.1)

The algorithm used to determine the timeout value used for retransmitting unacknowledged octets.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## tcpRtoMin (1.3.6.1.2.1.6.2)

The minimum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the LBOUND quantity described in RFC 793.

## Syntax

INTEGER

Access

read-only

Status

mandatory

## tcpRtoMax (1.3.6.1.2.1.6.3)

The maximum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the UBOUND quantity described in RFC 793.

Syntax

INTEGER

Access

read-only

Status

mandatory

## tcpMaxConn (1.3.6.1.2.1.6.4)

The limit on the total number of TCP connections the entity can support. In entities where the maximum number of connections is dynamic, this object should contain the value -1.

Syntax

INTEGER

Access

read-only

## Status

mandatory

## tcpActiveOpens (1.3.6.1.2.1.6.5)

The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## tcpPassiveOpens (1.3.6.1.2.1.6.6)

The number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.

## Syntax

Counter

## Access

read-only

## Status

mandatory



## tcpAttemptFails (1.3.6.1.2.1.6.7)

The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## tcpEstabResets (1.3.6.1.2.1.6.8)

The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## tcpCurrEstab (1.3.6.1.2.1.6.9)

The number of TCP connections for which the current state is either ESTABLISHED or CLOSE-WAIT.

## Syntax

Gauge

## Access

read-only

## Status

mandatory

# tcpInSegs (1.3.6.1.2.1.6.10)

The total number of segments received, including those received in error. This count includes segments received on currently established connections.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# tcpOutSegs (1.3.6.1.2.1.6.11)

The total number of segments sent including those on current connections, but excluding those containing only retransmitted octets.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## tcpRetransSegs (1.3.6.1.2.1.6.12)

The total number of segments retransmitted. That is, the number of TCP segments transmitted containing one or more previously transmitted octets.

## Syntax

Counter

## Access

read-only

## Status

mandatory

---

# The TCP Connection Table

The TCP connection table contains information about this entity's existing TCP connections.

## tcpConnState (1.3.6.1.2.1.6.13.1.1)

The state of this TCP connection. The only value which may be set by a management station is deleteTCB(12). Accordingly, it is appropriate for an agent to return a "badValue" response if a management station attempts to set this object to any other value.

If a management station sets this object to the value deleteTCB(12), then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed Node. The result is an immediate termination of the connection.

## Syntax

INTEGER

## Access

read-write

## Status

mandatory

## Return Value

Writes not supported.

# tcpConnLocalAddress (1.3.6.1.2.1.6.13.1.2)

The local IP address for this TCP connection. In the case of a connection in the listen state which is willing to accept connections for any IP interface associated with the Node, the value 0.0.0.0 is used.

## Syntax

IpAddress

## Access

read-only

## Status

mandatory

# tcpConnLocalPort (1.3.6.1.2.1.6.13.1.3)

The local port number for this TCP connection.

## Syntax

INTEGER (0.65535)

## Access

read-only

## Status

mandatory

## tcpConnRemAddress (1.3.6.1.2.1.6.13.1.4)

The remote IP address for this TCP connection.

## Syntax

IpAddress

## Access

read-only

## Status

mandatory

## tcpConnRemPort (1.3.6.1.2.1.6.13.1.5)

The remote port number for this TCP connection.

## Syntax

INTEGER (0.65535)

## Access

read-only

## Status

mandatory

---

## Additional TCP Objects

Following are the additional TCP objects.

### tcpInErrs (1.3.6.1.2.1.6.14)

The total number of segments received in error (for example, bad TCP checksums).

#### Syntax

Counter

#### Access

read-only

#### Status

mandatory

### tcpOutRsts (1.3.6.1.2.1.6.15)

The number of TCP segments sent containing the RST flag.

#### Syntax

Counter

#### Access

read-only

#### Status

mandatory

---

# The UDP Group

Implementation of the UDP group is mandatory for all systems which implement the UDP.

## udpInDatagrams (1.3.6.1.2.1.7.1)

The total number of UDP datagrams delivered to UDP users.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## udpNoPorts (1.3.6.1.2.1.7.2)

The total number of received UDP datagrams for which there was no application at the destination port.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## udpInErrors (1.3.6.1.2.1.7.3)

The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## udpOutDatagrams (1.3.6.1.2.1.7.4)

The total number of UDP datagrams sent from this entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

---

## The UDP Listener Table

The UDP listener table contains information about this entity's UDP end-points on which a local application is currently accepting datagrams.



## udpLocalAddress (1.3.6.1.2.1.7.5.1.1)

The local IP address for this UDP listener. In the case of a UDP listener which is willing to accept datagrams for any IP interface associated with the Node, the value 0.0.0.0 is used.

### Syntax

IpAddress

### Access

read-only

### Status

mandatory

## udpLocalPort (1.3.6.1.2.1.7.5.1.2)

The local port number for this UDP listener.

### Syntax

INTEGER (0.65535)

### Access

read-only

### Status

mandatory

---

## The EGP Group

Implementation of the EGP group is mandatory for all systems which implement the EGP.

## egpInMsgs (1.3.6.1.2.1.8.1)

The number of EGP messages received without error.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpInErrors (1.3.6.1.2.1.8.2)

The number of EGP messages received that proved to be in error.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpOutMsgs (1.3.6.1.2.1.8.3)

The total number of locally generated EGP messages.

### Syntax

Counter

### Access

read-only

## Status

mandatory

## egpOutErrors (1.3.6.1.2.1.8.4)

The number of locally generated EGP messages not sent due to resource limitations within an EGP entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

---

# The EGP Neighbor Table

The EGP neighbor table contains information about this entity's EGP neighbors.

## egpNeighState (1.3.6.1.2.1.8.5.1.1)

The EGP state of the local system with respect to this entry's EGP neighbor. Each EGP state is represented by a value that is one greater than the numerical value associated with the state in RFC 904.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## egpNeighAddr (1.3.6.1.2.1.8.5.1.2)

The IP address of this entry's EGP neighbor.

## Syntax

IpAddress

## Access

read-only

## Status

mandatory

## egpNeighAs (1.3.6.1.2.1.8.5.1.3)

The autonomous system of this EGP peer. Zero should be specified if the autonomous system number of the neighbor is not yet known.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## egpNeighInMsgs (1.3.6.1.2.1.8.5.1.4)

The number of EGP messages received without error from this EGP peer.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpNeighInErrs (1.3.6.1.2.1.8.5.1.5)

The number of EGP messages received from this EGP peer that proved to be in error (for example, bad EGP checksum).

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpNeighOutMsgs (1.3.6.1.2.1.8.5.1.6)

The number of locally generated EGP messages to this EGP peer.

### Syntax

Counter

## Access

read-only

## Status

mandatory

# egpNeighOutErrs (1.3.6.1.2.1.8.5.1.7)

The number of locally generated EGP messages not sent to this EGP peer due to resource limitations within an EGP entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# egpNeighInErrMsgs (1.3.6.1.2.1.8.5.1.8)

The number of EGP-defined error messages received from this EGP peer.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## egpNeighOutErrMsgs (1.3.6.1.2.1.8.5.1.9)

The number of EGP-defined error messages sent to this EGP peer.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpNeighStateUps (1.3.6.1.2.1.8.5.1.10)

The number of EGP state transitions to the UP state with this EGP peer.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## egpNeighStateDowns (1.3.6.1.2.1.8.5.1.11)

The number of EGP state transitions from the UP state to any other state with this EGP peer.

### Syntax

Counter

## Access

read-only

## Status

mandatory

# egpNeighIntervalHello (1.3.6.1.2.1.8.5.1.12)

The interval between EGP Hello command retransmissions, in hundredths of a second. This represents the t1 timer as defined in RFC 904.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

# egpNeighIntervalPoll (1.3.6.1.2.1.8.5.1.13)

The interval between EGP poll command retransmissions, in hundredths of a second. This represents the t3 timer as defined in RFC 904.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory



## egpNeighMode (1.3.6.1.2.1.8.5.1.14)

The polling mode of this EGP entity, either passive or active.

### Syntax

INTEGER {active(1), passive(2)}

### Access

read-only

### Status

mandatory

## egpNeighEventTrigger (1.3.6.1.2.1.8.5.1.15)

A control variable used to trigger operator-initiated *Start* and *Stop* events. When read, this variable always returns the most recent value that *egpNeighEventTrigger* was set to. If it has not been set since the last initialization of the network management subsystem on the Node, it returns a value of “stop”.

When set, this variable causes a *Start* or *Stop* event on the specified neighbor, as specified on pages 8-10 of RFC 904. Briefly, a *Start* event causes an *Idle* peer to begin neighbor acquisition and a non-*Idle* peer to re-initiate neighbor acquisition. A *stop* event causes a non-*Idle* peer to return to the *Idle* state until a *Start* event occurs, either by *egpNeighEventTrigger* or otherwise.

### Syntax

INTEGER {start(1), stop(2)}

### Access

read-write

### Status

mandatory

## egpAs (1.3.6.1.2.1.8.6)

The autonomous system number of this EGP entity.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

---

## The Transmission Group

Based on the transmission media underlying each interface on a system, the corresponding portion of the Transmission group is mandatory for that system.

When Internet-standard definitions for managing transmission media are defined, the transmission group is used to provide a prefix for the names of those objects.

Typically, such definitions reside in the experimental portion of the MIB until they are “proven”, then as a part of the Internet standardization process, the definitions are accordingly elevated and a new object identifier, under the transmission group is defined. By convention, the name assigned is:

type OBJECT IDENTIFIER ::= {transmission number}.

Where “type” is the symbolic value used for the media in the `ifType` column of the `ifTable` object, and “number” is the actual integer value corresponding to the symbol.

---

## The SNMP Group

Implementation of the SNMP group is mandatory for all systems which support an SNMP protocol entity. Some of the objects defined below will be zero-valued in those SNMP implementations that are optimized to support only those functions

specific to either a management agent or a management station. In particular, it should be observed that the objects below refer to an SNMP entity, and there may be several SNMP entities residing on a managed Node.

## snmpInPkts (1.3.6.1.2.1.11.1)

The total number of messages delivered to the SNMP entity from the transport service.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpOutPkts (1.3.6.1.2.1.11.2)

The total number of SNMP messages passed from the SNMP protocol entity to the transport service.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInBadVersions (1.3.6.1.2.1.11.3)

The total number of SNMP messages delivered to the SNMP protocol entity and were for an unsupported SNMP version.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInBadCommunityNames (1.3.6.1.2.1.11.4)

The total number of SNMP messages delivered to the SNMP protocol entity which used a SNMP community name not known to the entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInBadCommunityUses (1.3.6.1.2.1.11.5)

The total number of SNMP messages delivered to the SNMP protocol entity which represented an SNMP operation which was not allowed by the SNMP community named in the message.

### Syntax

Counter

Access

read-only

Status

mandatory

## snmpInASNParseErrs (1.3.6.1.2.1.11.6)

The total number of ASN.1 or BER errors encountered by the SNMP protocol entity when decoding received SNMP messages.

Syntax

Counter

Access

read-only

Status

mandatory

## snmpInTooBigs (1.3.6.1.2.1.11.8)

The total number of SNMP PDUs delivered to the SNMP protocol entity and for which the value of the error-status field is "tooBig".

Syntax

Counter

Access

read-only

Status

mandatory

## snmpInNoSuchNames (1.3.6.1.2.1.11.9)

The total number of SNMP PDUs delivered to the SNMP protocol entity and for which the value of the error-status field is “NoSuchName”.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInBadValues (1.3.6.1.2.1.11.10)

The total number of SNMP PDUs delivered to the SNMP protocol entity and for which the value of the error-status field is “badValue”.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInReadOnlys (1.3.6.1.2.1.11.11)

The total number valid SNMP PDUs delivered to the SNMP protocol entity and for which the value of the error-status field is "readOnly". It should be noted that it is a protocol error to generate an SNMP PDU which contains the value "readOnly" in the error-status field, as such, this object is provided as a means of detecting incorrect implementations of the SNMP.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInGenErrs (1.3.6.1.2.1.11.12)

The total number of SNMP PDUs delivered to the SNMP protocol entity and for which the value of the error-status field is "genErr".

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInTotalReqVars (1.3.6.1.2.1.11.13)

The total number of MIB objects retrieved successfully by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpInTotalSetVars (1.3.6.1.2.1.11.14)

The total number of MIB objects altered successfully by the SNMP protocol entity as the result of receiving valid SNMP `Set-Request` PDUs.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpInGetRequests (1.3.6.1.2.1.11.15)

The total number of SNMP `Get-Request` PDUs accepted and processed by the SNMP protocol entity.

## Syntax

Counter

## Access

read-only



## Status

mandatory

## snmpInGetNexts (1.3.6.1.2.1.11.16)

The total number of SNMP *Get-Next* PDUs accepted and processed by the SNMP protocol entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## snmpInSetRequests (1.3.6.1.2.1.11.17)

The total number of SNMP *Set-Request* PDUs accepted and processed by the SNMP protocol entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## snmpInGetResponses (1.3.6.1.2.1.11.18)

The total number of SNMP Get-Response PDUs accepted and processed by the SNMP protocol entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpInTraps (1.3.6.1.2.1.11.19)

The total number of SNMP Trap PDUs accepted and processed by the SNMP protocol entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpOutTooBigs (1.3.6.1.2.1.11.20)

The total number of SNMP PDUs generated by the SNMP protocol entity and for which the value of the error-status field is "tooBig"

### Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpOutNoSuchNames (1.3.6.1.2.1.11.21)

The total number of SNMP PDUs generated by the SNMP protocol entity and for which the value of the error-status is "NoSuchName".

## Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpOutBadValues (1.3.6.1.2.1.11.22)

The total number of SNMP PDUs generated by the SNMP protocol entity and for which the value of the error-status field is "badValue".

## Syntax

Counter

## Access

read-only

## Status

mandatory

## snmpOutGenErrs (1.3.6.1.2.1.11.24)

The total number of SNMP PDUs generated by the SNMP protocol entity and for which the value of the error-status field is "genErr".

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpOutGetRequests (1.3.6.1.2.1.11.25)

The total number of SNMP Get-Request PDUs generated by the SNMP protocol entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpOutGetNexts (1.3.6.1.2.1.11.26)

The total number of SNMP Get-Next PDUs generated by the SNMP protocol entity.

### Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpOutSetRequests (1.3.6.1.2.1.11.27)

The total number of SNMP Set-Request PDUs generated by the SNMP protocol entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

# snmpOutGetResponses (1.3.6.1.2.1.11.28)

The total number of SNMP Get-Response PDUs generated by the SNMP protocol entity.

## Syntax

Counter

## Access

read-only

## Status

mandatory

## snmpOutTraps (1.3.6.1.2.1.11.29)

The total number of SNMP Trap PDUs generated by the SNMP protocol entity.

### Syntax

Counter

### Access

read-only

### Status

mandatory

## snmpEnableAuthenTraps (1.3.6.1.2.1.11.30)

Indicates whether the SNMP agent process is permitted to generate authentication-failure traps. The value of this object overrides any configuration information; as such, it provides a means whereby all authentication-failure traps may be disabled.

It is strongly recommended that this object be stored in non-volatile memory so that it remains constant between re-initializations of the network management system.

### Syntax

INTEGER {enabled(1), disabled(2)}

### Access

read-write

### Status

mandatory

### Return Value

Read returns enabled (1) if AuthFailureTrap = True, otherwise disabled (2). Writes not supported.

# Fibre Alliance MIB Objects

---

This section covers the implementation details for the Fibre Alliance Management Information Bases (FA-MIB) version 6.0 on the switch.

---

## FA MIB Definitions

The FA-MIB version 4.0 is a collection of structured objects that resides on the workstation with the manager application. These objects define the syntax for information exchanged between the manager and the agent. The textual substitutions in [TABLE 4-1](#) are specific to the FA-MIB and can be used in place of primitive data types.

**TABLE 4-1** FA-MIB Textual Substitutions

Description	Syntax
FcNameId	OCTET STRING (SIZE(8))
FcGlobalId	OCTET STRING (SIZE(16))

**TABLE 4-1** FA-MIB Textual Substitutions (Continued)

Description	Syntax
FcAddressId	OCTET STRING (SIZE(3))
FcEventSeverity	INTEGER { unknown (1), emergency (2), alert (3), critical (4), error (5), warning (6), notify (7), info (8), debug (9), mark (10) - All messages logged }
FcUnitType	INTEGER { unknown(1) other(2) - none of the following hub(3) - passive connectivity unit supporting loop protocol. switch(4) - active connectivity unit supporting multiple protocols. gateway(5) - unit that converts not only the interface but also encapsulates the frame into another protocol. The assumption is that there is always two gateways connected together. For example, FC <-> ATM. converter(6) - unit that converts from one interface to another. For example, FC <-> SCSI. hba(7) - host bus adapter proxy-agent(8) - software proxy-agent storage-device(9) - disk, cd, tape, etc. host(10) - host computer storage-subsystem(11) - raid, library, etc. module(12) - subcomponent of a system swdriver(13) - software driver storage-access-device(14) - Provides storage management and access for heterogeneous hosts and heterogeneous devices wdm(15) - waveform division multiplexer ups(16) - uninterruptable power supply }

## revisionNumber

The revision number for this MIB. The format of the revision value is as follows:



- (0) = high order major revision number
- (1) = low order major revision number
- (2) = high order minor revision number
- (3) = low order minor revision number

The value will be stored as an ASCII value. The following is the current value of 04.00 for this object.

- (0) = '0'
- (1) = '4'
- (2) = '0'
- (3) = '0'

## Syntax

DisplayString (SIZE (4))

## Access

read-only

## Status

mandatory

## Return Value

A four digit ASCII value (for example, 0400 for MIB revision 4.0).

---

# Connectivity Unit Group

The objects described in this section are not in a table format. An example of how to access one of these objects is:

```
"snmpget localhost public fcmgmt.connSet.uNumber.0".
```

## uNumber (1.3.6.1.3.94.1.1)

The number of connectivity units present on this system (represented by this agent). May be a count of the boards in a chassis or the number of full boxes in a rack.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

The number of switches in fabric.

## systemURL (1.3.6.1.3.94.1.2)

The top-level URL of the system. If it does not exist, the value is an empty string. The URL format is implementation dependant and can have keywords embedded that are preceded by a percent sign (for example,%USER).

## Syntax

DisplayString

## Access

read-write

## Status

mandatory

## Return Value

The switch IP address. For example, http://10.0.0.1. Writes not supported, returns 'NoSuchName'.

## statusChangeTime (1.3.6.1.3.94.1.3)

The `sysUpTime` timestamp at which the last status change occurred for any members of the set, in centiseconds.

### Syntax

TimeTicks

### Access

read only

### Status

obsolete

### Return Value

The `sysUpTime` timestamp at which the last status change occurred.

## configurationChangeTime (1.3.6.1.3.94.1.4)

The `sysUpTime` timestamp at which the last configuration change occurred for any members of the set, in centiseconds. This represents a union of change information for `connUnitConfigurationChangeTime`.

### Syntax

TimeTicks

### Access

read only

### Status

obsolete

### Return Value

The `sysUpTime` timestamp at which the last configuration change occurred.

## connUnitTableChangeTime (1.3.6.1.3.94.1.5)

The `sysUpTime` timestamp at which the `connUnitTable` was updated (an entry was either added or deleted), in centiseconds.

### Syntax

TimeTicks

### Access

read only

### Status

obsolete

### Return Value

The `sysUpTime` timestamp at which the `connUnitTable` was updated.

---

## Connectivity Table

The objects described in this section are in a table format indexed by switch World Wide Name. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public
fcmgmt.connSet.connUnitTable.connUnitEntry.connUnitId..16.0.0.192.2
21.0.144.167.0.0.0.0.0.0.0.0.0"
```

## connUnitId (1.3.6.1.3.94.1.6.1.1)

The unique identification for this connectivity unit among those within this proxy domain. The value must be unique within the proxy domain because it is the index variable for `connUnitTable`. The value assigned to a given connectivity unit should be persistent across agent and unit resets. It should be the same as `connUnitGlobalId` if `connUnitGlobalId` is known and stable.

### Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

The World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

# connUnitGlobalId (1.3.6.1.3.94.1.6.1.2)

An optional global-scope identifier for this connectivity unit. It must be a WWN for this connectivity unit or 16 octets of value zero.

## Syntax

connUnitGlobalId

## Access

read-only

## Status

mandatory

## Return Value

The World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

# connUnitType (1.3.6.1.3.94.1.6.1.3)

The type of this connectivity unit.

## Syntax

FcUnitType

## Access

read-only

## Status

mandatory

## Return Value

switch (4)

# connUnitNumports (1.3.6.1.3.94.1.6.1.4)

Number of physical ports in the connectivity unit (internal/embedded, external).

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

The number of ports on the switch.

# connUnitState (1.3.6.1.3.94.1.6.1.5)

The operational state of the switch mapped. The overall state of connectivity unit.

## Syntax

INTEGER

Access

read-only

Status

mandatory

Return Value

Refer to [TABLE 4-2](#) for switch operational states.

**TABLE 4-2** Switch Operational States

Switch State	Return State
online	online (2)
offline	offline (3)
diagnostics	offline (3)
other	unknown (1)

# connUnitStatus (1.3.6.1.3.94.1.6.1.6)

Overall status of the connectivity unit. The goal of this object is to be the single poll point to check the status of the connunit. If there is any other component that has warning, then this should be set to warning. any of these values may occur with any of the ConnUnitState values.

Syntax

INTEGER

Access

read-only

Status

mandatory

## Return Value

Refer to [TABLE 4-3](#) for connectivity unit return values. Return value will be OK (3), unless one or more of the following occurs.

**TABLE 4-3** Sun Storage Fibre Channel Switch 5802 Connectivity Unit Return Values

Status	Return Value
If one power supply is reporting Bad and/or not installed	warning (4)
If both power supplies are reporting Bad and/or not installed	failed (5)
If one or more cooling fan failed	warning (4)
If all cooling fans failed	failed (5)
If temperature status = "Warm"	warning (4)
If temperature status = "Overheating"	failed (5)
If any port down	warning (4)
If POST failed	failed (5)
If switch Offline or in Diagnostics mode	warning (4)

## connUnitProduct (1.3.6.1.3.94.1.6.1.7)

The sml attribute `Config.Snmp.SysDescr`. This is the system description shown on the 'show version' telnet screen. It can also be read on the 'show setup snmp' screen and written using the 'set setup snmp' Telnet screen.

### Syntax

DisplayString (SIZE (0..79))

### Access

read-only

### Status

mandatory

## Return Value



## connUnitSn (1.3.6.1.3.94.1.6.1.8)

The serial number for this connectivity unit.

### Syntax

DisplayString (SIZE (0..79))

### Access

read-only

### Status

mandatory

### Return Value

The chassis serial number.

## connUnitUpTime (1.3.6.1.3.94.1.6.1.9)

The number of centiseconds since the last unit initialization.

### Syntax

TimeTicks

### Access

read-only

### Status

mandatory

### Return Value

The time interval since either `POST` or a `reset` (not including `hotreset` command for the NDCLA feature). `POST` (Power-On Self-Test) occurs during Power-On, or `hardreset`.

## `connUnitUrl` (1.3.6.1.3.94.1.6.1.10)

URL to launch a management application, if applicable. Otherwise, it's an empty string. In a standalone unit, this would be the same as the top-level URL. This has the same definition as `systemURL` for keywords. If `write` is not supported, then the return value is invalid. This value will be retained across boots.

### Syntax

`DisplayString`

### Access

read-write

### Status

mandatory

### Return Value

The switch IP address. For example, `http://10.0.0.1`. Writes not supported, returns `NoSuchName`.

## `connUnitDomainId` (1.3.6.1.3.94.1.6.1.11)

24 bit Fibre Channel address ID of this connectivity unit, right justified with leading zeros if required. This should be set to the Fibre Channel address ID, or if it is a switch, it would be set to the Domain Controller address. If this value is not applicable, return all bits set to one.

### Syntax

OCTET STRING (SIZE(3))

### Access

read-only

## Status

mandatory

## Return Value

The domain controller address. For example, FF FC 65.

## connUnitProxyMaster (1.3.6.1.3.94.1.6.1.12)

A value of “yes” means this is the proxy master unit for a set of managed units. For example, this could be the only unit with a management card in it for a set of units. A standalone unit should return “yes” for this object.

## Syntax

```
INTEGER {  
    unknown(1),  
    no(2),  
    yes(3)  
}
```

## Access

read-only

## Status

mandatory

## Return Value

If out-of-band switch, returns yes (3). If in-band switch, returns no (2).

## connUnitPrincipal (1.3.6.1.3.94.1.6.1.13)

Whether this connectivity unit is the principal unit within the group of fabric elements. If this value is not applicable, the return is unknown.

## Syntax

```
INTEGER {  
  unknown(1),  
  no(2),  
  yes(3)  
}
```

## Access

read-only

## Status

mandatory

## Return Value

For the principal switch, returns yes (3); otherwise returns no (2).

## connUnitNumSensors (1.3.6.1.3.94.1.6.1.14)

Number of sensors in the `connUnitSensorTable` elements. If this value is not applicable, return unknown.

## Syntax

```
INTEGER
```

## Access

read-only

## Status

mandatory

## Return Value

Returns the number of sensors listed in the `connUnitSensorTable`:

- Sun Storage FC Switch 5802 = 6

## connUnitStatusChangeTime (1.3.6.1.3.94.1.6.1.15)

The `sysUpTime` timestamp, in centiseconds, at which the last status change occurred.

### Syntax

TimeTicks

### Access

read-only

### Status

obsolete

### Return Value

This object is obsolete. Always returns error status "NoSuchName".

## connUnitConfigurationChangeTime (1.3.6.1.3.94.1.6.1.16)

The `sysUpTime` timestamp, in centiseconds, at which the last configuration change occurred.

### Syntax

TimeTicks

### Access

read-only

### Status

obsolete

### Return Value

This object is obsolete. Always returns error status "NoSuchName".

## connUnitNumRevs (1.3.6.1.3.94.1.6.1.17)

The number of revisions in the connUnitRevsTable.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

The number of entries in the revision table. The revision numbers of all components of the switch:

- Sun Storage FC Switch 5802 = 3

## connUnitNumZones (1.3.6.1.3.94.1.6.1.18)

Number of zones defined in connUnitZoneTable.

### Syntax

INTEGER

### Access

read-only

### Status

obsolete

### Return Value

This object is obsolete. Always returns error status "NoSuchName".

## connUnitModuleId (1.3.6.1.3.94.1.6.1.19)

This is a unique ID, persistent between boots, that can be used to group a set of connUnits together into a module. The intended use would be to create a connUnit with a connUnitType of “module” to represent a physical or logical group of connectivity units. Then, the value of the group would be set to the value of connUnitId for this “container” connUnit. connUnitModuleId should be zeros if this connUnit is not part of a module.

### Syntax

FcGlobalId

### Access

read-only

### Status

mandatory

### Return Value

The World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitName (1.3.6.1.3.94.1.6.1.20)

A display string containing a name for this connectivity unit. This object value should be persistent between boots.

### Syntax

DisplayString (SIZE(0..79))

### Access

read-write

### Status

mandatory

## Return Value

The SymbolicName of switch. The default is `Switch`.

## connUnitInfo (1.3.6.1.3.94.1.6.1.21)

A display string containing information about this connectivity unit. This object value should be persistent between boots.

## Syntax

DisplayString

## Access

read-write

## Status

mandatory

## Return Value

Returns the ConfigDescription field for the switch. The default is `Default Config`.

## connUnitControl (1.3.6.1.3.94.1.6.1.22)

This object is used to control the addressed connUnit. “Cold Start” and “Warm Start” are as defined in MIB-II and are not meant to be a factory reset.

- `resetConnUnitColdStart`: the addressed unit performs a “Cold Start” reset.
- `resetConnUnitWarmStart`: the addressed unit performs a “Warm Start” reset.
- `offlineConnUnit`: the addressed unit puts itself into an implementation dependant “offline” state. In general, if a unit is in an offline state, it cannot be used to perform meaningful Fibre Channel work.
- `onlineConnUnit`: the addressed unit puts itself into an implementation dependant “online” state. In general, if a unit is in an online state, it is capable of performing meaningful Fibre Channel work.

Each implementation may chose not to allow any or all of these values on a SET.



# Syntax

```
INTEGER {
  unknown(1),
  invalid(2),
  resetConnUnitColdStart(3),
  resetConnUnitWarmStart(4),
  offlineConnUnit(5),
  onlineConnUnit(6)
}
```

# Access

read-write

# Status

mandatory

# Return Value

Refer to the following tables for connUnitControl values.

**TABLE 4-4** connUnitControl Read Return Values

Switch Setting	Return Value
Online	Online (6)
Offline	Offline (5)
Diagnostics	Offline (5)
Other	Unknown (1)

**TABLE 4-5** connUnitControl Write Control Values

Control Value	Result
Cold Reset (3)	Reset
Offline (5)	Offline
Online (6)	Online
other	Not supported

## connUnitContact (1.3.6.1.3.94.1.6.1.23)

Contact information for this connectivity unit, and is persistent across boots.

### Syntax

DisplayString (SIZE (0..79))

### Access

read-write

### Status

mandatory

### Return Value

The default is: <sysContact undefined>. The string size is limited to a maximum of 64.

## connUnitLocation (1.3.6.1.3.94.1.6.1.24)

Location information for this connectivity unit, and is persistent across boots.

### Syntax

DisplayString (SIZE (0..79))

### Access

read-write

### Status

mandatory

### Return Value

The default is: <sysLocation undefined>. The string size is limited to a maximum of 64.

# connUnitEventFilter (1.3.6.1.3.94.1.6.1.25)

This value defines the event severity that will be logged by this connectivity unit. All events of severity less than or equal to connUnitEventFilter are logged in connUnitEventTable.

## Syntax

FcEventSeverity

## Access

read-write

## Status

mandatory

## Return Value

The switch log level setting. Refer to the following tables for connUnitEventFilter values.

**TABLE 4-6** connUnitEventFilter Read Return Values

Severity Levels	Return Value
Critical	Critical (4)
Warn	Warning (6)
Info	Info (8)
None	Unknown (1)

**TABLE 4-7** connUnitEventFilter Control Write Values

Control Value	Result
Emergency (2)	Critical
Alert (3)	Critical
Critical (4)	Critical
Error (5)	Warn
Warning (6)	Warn

**TABLE 4-7** connUnitEventFilter Control Write Values

Control Value	Result
Notify (7)	Info
Info (8)	Info
Debug (9)	Info
Mark (10)	Info
Unknown (1)	None

## connUnitNumEvents (1.3.6.1.3.94.1.6.1.26)

Number of events currently in the connUnitEventTable.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

An integer indicating the number of events in the event table.

## connUnitMaxEvents (1.3.6.1.3.94.1.6.1.27)

Maximum number of events that can be defined in connUnitEventTable.

### Syntax

INTEGER

### Access

read-only

## Status

mandatory

## Return Value

Always returns 30.

## connUnitEventCurrID (1.3.6.1.3.94.1.6.1.28)

The last used event ID (`connUnitEventIndex`).

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

The event ID of the last event.

## connUnitFabricID (1.3.6.1.3.94.1.6.1.29)

A globally unique value to identify the fabric that this `ConnUnit` belongs to, otherwise empty string. This would typically be equal to the `connUnitGlobalID` of the primary switch in a Fibre Channel fabric.

## Syntax

FcGlobalId

MaxAccess

read-only

## Status

mandatory

## Return Value

Returns the World Wide Name of the principal switch followed by 8 bytes of zeros.  
For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

## connUnitNumLinks (1.3.6.1.3.94.1.6.1.30)

The number of links in the link table.

## Syntax

INTEGER

MaxAccess

read-only

## Status

mandatory

## Return Value

Returns the number of link table entries for each switch.

## connUnitVendorId (1.3.6.1.3.94.1.6.1.31)

The connectivity unit vendor's name.

## Syntax

DisplayString (SIZE (0..79))

read-only

## Status

mandatory

## Return Value

"Sun"

---

# Revision Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The Table of revisions for hardware and software elements. There are four revision items in each switch. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public  
fcgmt.connSet.connUnitRevsTable.connUnitRevsEntry.connUnitRevsU  
nitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.0.1".
```

The number of entries in this table will be variable depending on which platform is being examined and the number of blades installed. SNMP first reports the firmware revision and flasher shell version. It then iterates through each of the installed blades reporting the PCB revision and ASIC version.

## connUnitRevsUnitId (1.3.6.1.3.94.1.7.1.1)

The connUnitId of the connectivity unit that contains this revision table.

## Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

Returns the World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

## connUnitRevsIndex (1.3.6.1.3.94.1.7.1.2)

A unique value among all `connUnitRevsEntry`s with the same value of `connUnitRevsUnitId`, in the range between 1 and `connUnitNumRevs[connUnitRevsUnitId]`.

### Syntax

INTEGER (1..2147483647)

### Access

read-only

### Status

mandatory

### Return Value

The revision table index.

## connUnitRevsRevId (1.3.6.1.3.94.1.7.1.3)

A vendor-specific string identifying a revision of a component of the `connUnit` indexed by `connUnitRevsUnitId`.

### Syntax

DisplayString

### Access

read-only

### Status

mandatory

### Return Value

Refer to [TABLE 4-8](#) for `connUnitRevsRevId` return values.



**TABLE 4-8** ConnUnitRevsRevId Return Values

Table Index	Return Value
1	Active Firmware Image
2	Flasher Shell Version
3	Hardware ASIC Version (1 per blade)

## connUnitRevsDescription (1.3.6.1.3.94.1.7.1.4)

Description of a component to which the revision corresponds.

### Syntax

DisplayString

### Access

read-only

### Status

mandatory

### Return Value

Refer to [TABLE 4-9](#) for connUnitRevsDescription return values.

**TABLE 4-9** ConnUnitRevsDescription Return Values

Table Index	Return Value
1	Active Firmware Version
2	Flasher Shell Version
3	Hardware ASIC Version

---

# Sensor Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The Index is the sensor number being interrogated. There are six sensor items in each switch. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public
fcmgmt.connSet.connUnitSensorTable.connUnitSensorEntry.connUnitS
ensorUnitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.0.1".
```

## connUnitSensorUnitId (1.3.6.1.3.94.1.8.1.1)

The connUnitId of the connectivity unit that contains this sensor table.

### Syntax

FcGlobalId

### Access

read-only

### Status

mandatory

### Return Value

Returns the World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitSensorIndex (1.3.6.1.3.94.1.8.1.2)

A unique value among all connUnitSensorEntrys with the same value of connUnitSensorUnitId, in the range between 1 and connUnitNumSensor[connUnitSensorUnitId].

### Syntax

INTEGER (1..2147483647)

Access

read-only

Status

mandatory

Return Value

The sensor table index.

connUnitSensorName (1.3.6.1.3.94.1.8.1.3)

A textual identification of the sensor intended primarily for operator use.

Syntax

DisplayString

Access

read-only

Status

mandatory

Return Value

Refer to [TABLE 4-10](#) for Sun Storage Fibre Channel Switch 5802 connUnitSensorName return values.

**TABLE 4-10** Sun Storage Fibre Channel Switch 5802 ConnUnitSensorName Return Values

Table Index	Return Value
1	Power Supply 1 Status
2	Power Supply 2 Status
3	Fan 1 Status

**TABLE 4-10** Sun Storage Fibre Channel Switch 5802 ConnUnitSensorName Return Values

Table Index	Return Value
4	Fan 2 Status
5	Temperature Status
6	Temperature Sensor 1 Value

# connUnitSensorStatus (1.3.6.1.3.94.1.8.1.4)

The status indicated by the sensor.

## Syntax

```
INTEGER {  
  unknown(1)  
  other(2) - the sensor indicates other than ok (warning or failure).  
  ok(3) - the sensor indicates ok  
  warning(4) - the sensor indicates a warning  
  failed(5) - the sensor indicates failure  
}
```

## Access

read-only

## Status

mandatory

## Return Value

Refer to the following tables for connUnitSensorStatus return values.

**TABLE 4-11** ConnUnitSensorStatus Return Values for Board Temperature

Switch Value	Return Value
Normal	OK (3)
Warm	Warning (4)
Overheating	Failed (5)
Other	Unknown (1)

**TABLE 4-12** ConnUnitSensorStatus Return Values for Fan Status

Switch Value	Return Value
Good	OK (3)
Bad	Failed (5)
Other	Unknown (1)

**TABLE 4-13** ConnUnitSensorStatus Return Values for Voltage Status

Switch Value	Return Value
Good	OK (3)
Bad	Failed (5)
Other	Unknown (1)

## connUnitSensorInfo (1.3.6.1.3.94.1.8.1.5)

Miscellaneous static information about the sensor, such as its serial number.

### Syntax

DisplayString

### Access

read-only

### Status

mandatory

### Return Value

Always returns an empty string.

# connUnitSensorMessage (1.3.6.1.3.94.1.8.1.6)

This describes the status of the sensor as a message. It may also provide more resolution on the sensor indication. For example, “Cover temperature 1503K, above nominal operating range” ::= { connUnitSensorEntry 6 }.

## Syntax

DisplayString

## Access

read-only

## Status

mandatory

## Return Value

Refer to [TABLE 4-14](#) for connUnitSensorMessage values.

**TABLE 4-14** ConnUnitSensorMessage Values

Sensor	Value
Power Supply	Good/Bad/NotInstalled
Fan	Good/Bad/NotInstalled
Temperature Status	Normal/Warm/Overheating/NotInstalled
Temperature Value	Degrees in C

# connUnitSensorType (1.3.6.1.3.94.1.8.1.7)

The type of component being monitored by this sensor.

## Syntax

INTEGER {  
unknown(1),  
other(2),  
battery(3),  
fan(4),  
power-supply(5),

```
transmitter(6),
enclosure(7),
board(8),
receiver(9)
}
```

Access

read-only

Status

mandatory

Return Value

Refer to [TABLE 4-15](#) for connUnitSensorType return values.

**TABLE 4-15** ConnUnitSensorType Return Values

Sensor	Value
Temperature	Board (8)
Fan	Fan (4)
Power Supply	Power Supply (5)
Voltage	Board (8)

connUnitSensorCharacteristic (1.3.6.1.3.94.1.8.1.8)

The characteristics being monitored by this sensor.

Syntax

```
INTEGER {
unknown(1),
other(2),
temperature(3),
pressure(4),
emf(5),
currentValue(6), - current is a keyword
airflow(7),
frequency(8),
```

```
power(9),
door(10)
}
```

Access

read-only

Status

mandatory

Return Value

Refer to [TABLE 4-16](#) for connUnitSensorCharacteristic values.

**TABLE 4-16** ConnUnitSensorCharacteristic Values

Sensor	Value
Temperature Value	Temperature (3)
Temperature Status	Temperature (3)
Fan	Airflow (7)
Power Supply	Power (9)

# Port Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The Index is the port number being interrogated. There may be different numbers of ports in each switch so the agent must determine the maximum allowable index on a switch by switch basis. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public
fcmgmt.connSet.connUnitPortTable.connUnitPortUnitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.1".
```

## connUnitPortUnitId (1.3.6.1.3.94.1.10.1.1)

The connUnitId of the connectivity unit that contains this port.



## Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

Returns the World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

## connUnitPortIndex (1.3.6.1.3.94.1.10.1.2)

A unique value among all `connUnitPortEntrys` on this connectivity unit, between 1 and `connUnitNumPort[connUnitPortUnitId]`.

## Syntax

INTEGER (1..2147483647)

## Access

read-only

## Status

mandatory

## Return Value

The index for each port on the switch. Sun Storage Fibre Channel Switch 5802 = 1 - 8,12,16,20 (varies depending on number of licensed ports)

## connUnitPortType (1.3.6.1.3.94.1.10.1.3)

The port type.

## Syntax

```
INTEGER {
  unknown(1),
  other(2),
  not-present(3),
  hub-port(4),
  n-port(5), - end port for fabric
  nl-port(6), - end port for loop
  fl-port(7), - public loop
  f-port(8), - fabric port
  e-port(9), - fabric expansion port
  g-port(10), - generic fabric port
  domain-ctl(11), - domain controller
  hub-controller(12),
  scsi(13), - parallel SCSI port
  escon(14),
  lan(15),
  wan(16),
  ac(17), - AC power line
  dc(18), - DC power line
  ssa(19) - serial storage architecture
  wdm(20),-- optical wave division multiplex
  ib 21), - Infiniband
  ipstore(22) - IP storage
}
```

## Access

read-only

## Status

mandatory

## Return Value

Refer to [TABLE 4-17](#) for connUnitPortType return values.

**TABLE 4-17** ConnUnitPortType Return Values

Switch Port Type	Return Value
G	g-port (10)
FL	fl-port (7)
F	f-port (8)

**TABLE 4-17** ConnUnitPortType Return Values

Switch Port Type	Return Value
E	e-port (9)
Donor	other (2)
other	unknown (1)

## connUnitPortFCClassCap (1.3.6.1.3.94.1.10.1.4)

Bit mask that specifies the classes of service capability of this port. If this is not applicable, return all bits set to zero.

The bits have the following definition:

unknown - 0

class-f - 1

class-one - 2

class-two - 4

class-three - 8

class-four - 16

class-five - 32

class-six - 64

### Syntax

OCTET STRING (SIZE (2))

### Access

read-only

### Status

mandatory

### Return Value

Always returns 0x0d (Class f, Class 2, and Class 3).

## connUnitPortFCClassOp (1.3.6.1.3.94.1.10.1.5)

Bit mask that specifies the classes of service that are currently operational. If this is not applicable, return all bits set to zero. This object has the same definition as connUnitPortFCClassCap" ::= { connUnitPortEntry 5 }.

### Syntax

OCTET STRING (SIZE (2))

### Access

read-only

### Status

mandatory

### Return Value

If F or FL, returns 0x0c (Class 2, and Class 3), else returns 0x0d (Class f, Class 2, and Class 3).

## connUnitPortState (1.3.6.1.3.94.1.10.1.6)

The user selected state of the port hardware.

### Syntax

```
INTEGER {  
  unknown(1),  
  online(2), - available for meaningful work  
  offline(3), - not available for meaningful work  
  bypassed(4), - no longer used (4/12/00)  
  diagnostics(5)  
}
```

### Access

read-only

### Status

mandatory

## Return Value

Refer to [TABLE 4-18](#) for connUnitPortState return values.

**TABLE 4-18** ConnUnitPortState Return Values

Port Value	Return Value
Online	online (2)
Offline	offline (3)
Downed	offline (3)
Diagnostics	diagnostics (5)
other	unknown (1)

## connUnitPortStatus (1.3.6.1.3.94.1.10.1.7)

An overall protocol status for the port. This value of connUnitPortState is not online, then this is reported Unknown.

## Syntax

```
INTEGER {  
    unknown(1),  
    unused(2), - device cannot report this status  
    ready(3), - FCAL Loop or FCPH Link reset protocol; initialization complete  
    warning(4), - do not use (4/12/00)  
    failure(5), - do not use (4/12/00)  
    notparticipating(6), - loop not participating and does not have a loop address  
    initializing(7), - protocol is proceeding  
    bypass(8), - do not use (4/12/00)  
    ols(9) - FCP offline status  
    other(10) - status not described above  
}
```

## Access

read-only

## Status

mandatory

## Return Value

Always returns unused (2).

# connUnitPortTransmitterType (1.3.6.1.3.94.1.10.1.8)

The technology of the port transceiver.

## Syntax

```
INTEGER {  
    unknown(1),  
    other(2),  
    unused(3),  
    shortwave(4),  
    longwave(5),  
    copper(6),  
    scsi(7),  
    longwaveNoOFC(8),  
    shortwaveNoOFC(9),  
    longwaveLED(10),  
    ssa(11)  
}
```

## Access

read-only

# Status

mandatory

# Return Value

Refer to [TABLE 4-19](#) for connUnitPortTransmitterType return values.

**TABLE 4-19** ConnUnitPortTransmitterType Return Values

SFP Transmitter Type	Return Value
Not Installed	Unused (3)
SL	Shortwave (4)
LL	Longwave (5)
LC	LongwaveNoOFC (8)
SN	ShortwaveNoOFC (9)
EL	Copper (6)
Other	Unknown (1)

# connUnitPortModuleType (1.3.6.1.3.94.1.10.1.9)

The module type of the port connector.

# Syntax

INTEGER {  
unknown(1),  
other(2),  
gbic(3),  
embedded(4), - fixed (oneXnine)  
glm(5),  
gbicSerialId(6),  
gbicNoSerialId(7),  
gbicNotInstalled(8),  
smallFormFactor(9) - this is generically a small form factor connector.

}

Access

read-only

Status

mandatory

Return Value

Refer to [TABLE 4-20](#) for connUnitPortModuleType return values.

**TABLE 4-20** ConnUnitPortModuleType Return Values

Type	Value
1 Gb/2Gb Ports	smallFormFactor(9)
10 Gb Ports	Other (2)

connUnitPortWwn (1.3.6.1.3.94.1.10.1.10)

The World Wide Name of the port, if applicable, otherwise returns all zeros.

Syntax

FcGlobalId

Access

read-only

Status

mandatory

Return Value

Returns the Port World Wide Name followed by 8 bytes of zeros. For example, the return value for port #2 would be 20 02 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00, and the return value for port #14 would be 20 0E 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00. If a port is configured as a Donor, return value = 0.



## connUnitPortFCId (1.3.6.1.3.94.1.10.1.11)

This is the assigned Fibre Channel ID of this port. This value is expected to be a Big Endian value of 24 bits. If this is a loop, then it is the ALPA that is connected. If this is an E\_Port, then it will only contain the domain ID left justified, zero filled. If this port does not have a Fibre Channel address, returns all bits set to 1.

### Syntax

FcAddressId

### Access

read-only

### Status

mandatory

### Return Value

The address for each port based on Domain, Area, and ALPA. For example, port #15 would be equal to 640F00 (Domain = 0x64, Area = 0x0F, ALPA = 0x00).

## connUnitPortSn (1.3.6.1.3.94.1.10.1.12)

The serial number of the unit. If not applicable, returns an empty string.

### Syntax

DisplayString (SIZE(0..79))

### Access

read-only

### Status

unsupported

### Return Value

Return Value = The media part number of the SFP (or `MediaPartNumber`) if installed.

## connUnitPortRevision (1.3.6.1.3.94.1.10.1.13)

The port revision. For example, for a GBIC.

### Syntax

`DisplayString (SIZE(0..79))`

### Access

read-only

### Status

unsupported

### Return Value

Return Value = The media revision of the SFP (or `MediaRevision`) if installed.

## connUnitPortVendor (1.3.6.1.3.94.1.10.1.14)

The port vendor. For example, for a GBIC.

### Syntax

`DisplayString (SIZE(0..79))`

### Access

read-only

### Status

unsupported

### Return Value

The port vendor as reported by the SFP (if supported).

## connUnitPortSpeed (1.3.6.1.3.94.1.10.1.15)

The speed of the port in kilobytes per second.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

The operational speed, otherwise returns the administrative speed setting.

If 1 Gbps, returns 106250.

If 2 Gbps, returns 212500.

If 4 Gbps, returns 425000.

If 10 Gbps, returns 1062500.

## connUnitPortControl (1.3.6.1.3.94.1.10.1.16)

This object is used to control the addressed connUnit's port.

- `resetConnUnitPort`: If the addressed connUnit allows this operation to be performed to this port, the addressed port performs a vendor-specific "reset" operation. Examples of these operations are: the Link Reset protocol, the Loop Initialization protocol, or a re-synchronization occurring between the transceiver in the addressed port to the transceiver that the port is connected to.
- `bypassConnUnitPort`: If the addressed connUnit allows this operation to be performed to this port, the addressed port performs a vendor-specific "bypass" operation. Examples of these operations are transitioning from online to offline, a request (non-participating) command to the Loop Port state machine, or removal of the port from an arbitrated loop by a hub.
- `unbypassConnUnitPort`: If the addressed connUnit allows this operation to be performed to this port, the addressed port performs a vendor-specific "unbypass" operation. Examples of these operations are the Link Failure protocol, a request (participating) command to the Loop Port state machine, or addition of the port to an arbitrated loop by a hub.

- `offlineConnUnitPort`: If the addressed `connUnit` allows this operation to be performed to this port, the addressed port performs a vendor-specific “offline” operation. Examples of these operations are disabling a port's transceiver, the Link Failure protocol, request (non-participating) command to the Loop Port state machine, or removal of the port from an arbitrated loop by a hub.
- `onlineConnUnitPort`: If the addressed `connUnit` allows this operation to be performed to this port, the addressed port performs a vendor-specific “online” operation. Examples of these operations are enabling a port's transceiver, the Link Failure protocol, request (participating) command to the Loop Port state machine, or addition of the port from an arbitrated loop by a hub.
- `resetConnUnitPortCounters`: If the addressed `connUnit` allows this operation to be performed to this port, the addressed port statistics table counters will be set to zero.

Each implementation may choose not to allow any or all of these values on a SET. On a read, if you do not support write, then return invalid. Otherwise, return the last control operation attempted.

## Syntax

```
INTEGER {
    unknown(1),
    invalid(2),
    resetConnUnitPort(3),
    bypassConnUnitPort(4),
    unbypassConnUnitPort(5),
    offlineConnUnitPort(6),
    onlineConnUnitPort(7),
    resetConnUnitPortCounters(8)
}
```

## Access

read-write

## Status

mandatory

## Return Value

Refer to [TABLE 4-21](#) for connUnitPortControl read return values.

**TABLE 4-21** ConnUnitPortControl Read Return Values

Port Value	Return Value
Online	online (7)
Offline	offline (6)
Diagnostics	offline (6)
other	unknown (1)

Refer to [TABLE 4-22](#) for connUnitPortControl write command values.

**TABLE 4-22** ConnUnitPortControl Write Command Values

Control Value	Command Sent
Online (7)	online
Offline (6)	offline
ResetCounters (8)	clear counters
other	error returned

## connUnitPortName (1.3.6.1.3.94.1.10.1.17)

A user-defined name for this port. This means that up to `DisplayString` characters may be supported. If less than, then the name will be truncated in the connunit.

### Syntax

INTEGER

### Access

read-write

### Status

mandatory

## Return Value

The symbolic port name. A 1G or 2G only capable port, would return port followed by the port number. 10G ports would return 10G followed by the port number. For example, a 1G/2G port#2 would return 'Port2' and a 10G port#18 would return '10G-18' by default.

## connUnitPortPhysicalNumber (1.3.6.1.3.94.1.10.1.18)

This is the internal port number this port is known by. In many implementations, this should be the same as `connUnitPortIndex`. Some implementations may have an internal port representation not compatible with the rules for table indexes. In that case, provide the internal representation of this port in this object. This value may also be used in the `connUnitLinkPortNumberX` or `connUnitLinkPortNumberY` objects of the `connUnitLinkTable`.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

The physical port number.

## connUnitPortStatObject (1.3.6.1.3.94.1.10.1.19)

This contains the OID of the first object of the table that contains the statistics for this particular port. If this has a value of zero, then there are no statistics available for this port. The port type information will help identify the statistics objects that will be found in the table.

## Syntax

## OBJECT IDENTIFIER

### Access

read-only

### Status

deprecated

### Return Value

The port object ID (1.2.6.1.3.94.4.5.1.1).

## connUnitPortProtocolCap (1.3.6.1.3.94.1.10.1.20)

Bit mask that specifies the driver level protocol capability of this port. If this is not applicable, returns all bits set to zero.

The bits have the following definitions:

unknown - 0

Loop - 1

Fabric - 2

SCSI - 4

TCP/IP - 8

VI - 16

FICON - 32

### Syntax

OCTET STRING (SIZE (2))

### Access

read-only

### Status

mandatory

## Return Value

Always returns 0x03 (Loop, Fabric).

## connUnitPortProtocolOp (1.3.6.1.3.94.1.10.1.21)

Bit mask that specifies the driver level protocol(s) that are currently operational. If not applicable, return all bits set to zero. This object has the same definition as connUnitPortProtocolCap.

## Syntax

OCTET STRING (SIZE (2))

## Access

read-only

## Status

unsupported

## Return Value

Always returns error status "NoSuchName".

## connUnitPortNodeWwn (1.3.6.1.3.94.1.10.1.22)

The Node World Wide Name of the port if applicable, otherwise all zeros. This should have the same value for a group of related ports. The container is defined as the largest physical entity. For example, all ports on HBAs on a host will have the same Node WWN. All ports on the same storage subsystem will have the same Node WWN." ::= { connUnitPortEntry 22 }.

## Syntax

FcNameId

## Access

read-only



## Status

mandatory

## Return Value

Returns the World Wide Node Name of the switch. For example: 10 00 00 C0 DD 00 71 C9.

# connUnitPortHWState (1.3.6.1.3.94.1.10.1.23)

The hardware detected state of the port.

## Syntax

```
INTEGER {  
    unknown(1),  
    failed(2), - port failed diagnostics  
    bypassed(3), - FCAL bypass, loop only  
    active(4), - connected to a device  
    loopback(5), - Port in external loopback  
    txfault(6), - Transmitter fault  
    noMedia(7), - media not installed linkDown  
    (8) - waiting for activity (rx sync)  
}
```

## Access

read-only

## Status

mandatory

## Return Value

Refer to [TABLE 4-23](#) for connUnitPortHWState port state return values.

**TABLE 4-23** ConnUnitPortHWState Port State Return Values

Port State	Return Value
If DiagStatus = Failed	Failed (2)
If SFP = Not Installed	NoMedia (7)
If SyncStatus = SyncAcquired	Active (4)
If SyncStatus = SyncLost	LinkDown (8)
Other	Unknown (1)

# Event Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The maximum index is determined based on the number of events in the table. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public
fcmgmt.connSet.connUnitEventTable.connUnitEventEntry.connUnitEventUnitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.0.1".
```

## connUnitEventUnitId (1.3.6.1.3.94.1.11.1.1)

The connUnitId of the connectivity unit that contains this event table.

### Syntax

FcGlobalId

### Access

read-only

### Status

mandatory

### Return Value

The World Wide Name of the switch followed by 8 bytes of zeros. For example, 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitEventIndex (1.3.6.1.3.94.1.11.1.2)

Each connectivity unit has its own event buffer. As it wraps, it may write over previous events. This object is an index into the buffer. It is recommended that this table be read using getNext's to retrieve the initial table. The management application should read the event table at periodic intervals and then determine if any new entries were added by comparing the last known index value with the current highest index value. The management application should then update its copy of the event table. If the read interval is too long, it is possible that there may be events that may not be contained in the agent's internal event buffer. For example, an agent may read events 50-75. At the next read interval, connUnitEventCurrID is 189. If the management application tries to read event index 76, and the agent's internal buffer is 100 entries max, event index 76 will no longer be available.

The index value is an incrementing integer starting from one every time there is a table reset. On table reset, all contents are emptied and all indexes are set to zero. When an event is added to the table, the event is assigned the next higher integer value than the last item entered into the table. If the index value reaches its maximum value, the next item entered will cause the index value to roll over and start at one again.

### Syntax

INTEGER (1..2147483647)

### Access

read-only

### Status

mandatory

### Return Value

The table index.

## connUnitEventId (1.3.6.1.3.94.1.11.1.3)

The internal event ID. Incremented for each event, ranging between 1 and `connUnitMaxEvents`. Not used as table index to simplify the agent implementation. When this reaches the end of the range specified by `connUnitMaxEvents`, the ID will roll over to start at one. This value will be set back to one at reset. The relationship of this value to the index is that internal event ID may represent a smaller number than a 32 bit integer (for example, maximum 100 entries) and would only have a value range up to `connUnitMaxEvents`.

### Syntax

INTEGER

### Access

read-only

### Status

deprecated

### Return Value

Unsupported. Always returns error status "NoSuchName".

## connUnitREventTime (1.3.6.1.3.94.1.11.1.4)

The real time when the event occurred. It has the following format.

DDMMYYYY HHMMSS

DD=day number

MM=month number

YYYY=year number

HH=hour number

MM=minute number

SS=seconds number

If not applicable, return either a NULL string or "00000000 000000".

## Syntax

DisplayString (SIZE (0..15))

## Access

read-only

## Status

mandatory

## Return Value

The timestamp of the event.

# connUnitSEventTime (1.3.6.1.3.94.1.11.1.5)

This is the sysUpTime timestamp when the event occurred.

## Syntax

connUnitSEventTime

## Access

read-only

## Status

mandatory

## Return Value

Always returns error status "NoSuchName".

# connUnitEventSeverity (1.3.6.1.3.94.1.11.1.6)

The event severity level.

## Syntax

FcEventSeverity

## Access

read-only

## Status

mandatory

## Return Value

Always returns error status "NoSuchName".

# connUnitEventType (1.3.6.1.3.94.1.11.1.7)

The type of this event.

## Syntax

```
INTEGER {  
    unknown(1),  
    other(2),  
    status(3),  
    configuration(4),  
    topology(5)  
}
```

## Access

read-only

## Status

mandatory

## Return Value

Always returns 3 (Status).

## connUnitEventObject (1.3.6.1.3.94.1.11.1.8)

This is used with the `connUnitEventType` to identify which object the event refers to. Examples include `connUnitPortStatus.connUnitId.connUnitPortIndex` and `connUnitStatus.connUnitId`.

### Syntax

OBJECT IDENTIFIER

### Access

read-only

### Status

mandatory

### Return Value

Always returns error status "NoSuchName".

## connUnitEventDescr (1.3.6.1.3.94.1.11.1.9)

The description of the event.

### Syntax

DisplayString

### Access

read-only

### Status

mandatory

### Return Value

The event description in the form: "[*Id*][*timestamp*][*severity*][*module*][*Description*]"

---

# Link Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The index is an index into the link table for the switch. There may be as many link entries as there are ports. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public  
fcmgmt.connSet.connUnitLinkTable.connUnitLinkEntry.connUnitLinkU  
nitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.0.1".
```

If the agent is able to discover links which do not directly attach to members of its agency and its discovery algorithm gives some assurance the links are recently valid, it *may* include these links. Link information entered by administrative action *may* be included even if not validated directly if the link has at least one endpoint in this agency, but *should not* be included otherwise.

A connectivity unit should fill the table in as best it can. One of the methods to fill this in would be to use the RNID ELS (ANSI document 99-422v0). This allows one to query a port for the information needed for the link table.

This table is accessed either directly if the management software has an index value or via GetNexts. The value of the indexes are not required to be contiguous. Each entry created in this table will be assigned an index. This relationship is kept persistent until the entry is removed from the table or the system is reset. The total number of entries are defined by the size of the table

## connUnitLinkId (1.3.6.1.3.94.1.12.1.1)

The connUnitId of the connectivity unit that contains this link table.

### Syntax

connUnitLinkId

### Access

read-only

### Status

mandatory



## Return Value

The World Wide Name of the switch followed by 8 bytes of zeros. For example, 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitLinkIndex (1.3.6.1.3.94.1.12.1.2)

This index is used to create a unique value for each entry in the link table with the same connUnitLinkId. The value can only be reused if it is not currently in use and the value is the next candidate to be used. This value wraps at the highest value represented by the size of INTEGER. This value is reset to zero when the system is reset, and the first value to be used is one.

## Syntax

INTEGER (1..2147483647)

## Access

read-only

## Status

mandatory

## Return Value

The table index.

## connUnitLinkNodeIdX (1.3.6.1.3.94.1.12.1.3)

The Node WWN of the unit at one end of the link. If the Node WWN is unknown and the Node is a connUnit in the responding agent, then the value of this object must be equal to its connUnitID.

## Syntax

OCTET STRING (SIZE(16))

## Access

read-only

## Status

mandatory

## Return Value

The World Wide Name of the local switch for each entry in the link table. For example, 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitLinkPortNumberX (1.3.6.1.3.94.1.12.1.4)

The port number on the unit specified by `connUnitLinkNodeIdx` if known, otherwise -1. If the value is non-negative, then it will be equal to `connUnitPortPhysicalNumber`.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

The local port number for each entry in the link table.

## connUnitLinkPortWwnX (1.3.6.1.3.94.1.12.1.5)

The port WWN of the unit specified by `connUnitLinkNodeIdx` if known, otherwise 16 octets of binary 0" ::= { `connUnitLinkEntry` 5 }.

## Syntax

`connUnitLinkPortWwnX`

## Access

read-only

## Status

mandatory

## Return Value

The local World Wide port number for each entry in the link table.

## connUnitLinkNodeIdY (1.3.6.1.3.94.1.12.1.6)

The Node WWN of the unit at the other end of the link. If the Node WWN is unknown and the Node is a connUnit in the responding SNMP agency, then the value of this object must be equal to its connUnitID.

## Syntax

OCTET STRING (SIZE(16))

## Access

read-only

## Status

mandatory

## Return Value

The remote World Wide Node number for each entry in the link table.

## connUnitLinkPortNumberY (1.3.6.1.3.94.1.12.1.7)

The port number on the unit specified by connUnitLinkNodeIdY if known, otherwise -1. If the value is non-negative, then it will be equal to connUnitPortPhysicalNumber.

## Syntax

OCTET STRING (SIZE(16))

## Access

read-only

## Status

mandatory

## Return Value

The remote port number for inter-switch link, if known. Otherwise, -1 (0xFFFFFFFF).

## connUnitLinkPortWwnY (1.3.6.1.3.94.1.12.1.8)

The port WWN on the unit specified by `connUnitLinkNodeIdY` if known, otherwise 16 octets of binary 0" ::= { `connUnitLinkEntry` 8 }.

## Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

The remote Port World Wide Name for each entry in the link table, if known.

## connUnitLinkAgentAddressY (1.3.6.1.3.94.1.12.1.9)

The address of an FCMGMT MIB agent for the Node identified by `connUnitLinkNodeIdY`, if known. Otherwise 16 octets of binary 0" ::= {`connUnitLinkEntry` 9}.

## Syntax

OCTET STRING (SIZE(16))

#### Access

read-only

#### Status

mandatory

#### Return Value

The remote IP address of the remote switch, if known. Otherwise, returns sixteen zeroes.

### connUnitLinkAgentAddressTypeY (1.3.6.1.3.94.1.12.1.10)

If connUnitLinkAgentAddressY is nonzero, it is a protocol address. ConnUnitLinkAgentAddressTypeY is the “address family number” assigned by IANA to identify the address format.

#### Syntax

INTEGER

#### Access

read-only

#### Status

mandatory

#### Return Value

Always returns 1 (Ipv4).

## connUnitLinkAgentPortY (1.3.6.1.3.94.1.12.1.11)

The IP port number for the agent. This is provided in case the agent is at a non-standard SNMP port.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

Returns value of 0.

## connUnitLinkUnitTypeY (1.3.6.1.3.94.1.12.1.12)

Type of the Fibre Channel connectivity unit as defined in connUnitType.

### Syntax

FcUnitType

### Access

read-only

### Status

mandatory

### Return Value

The type of remote device in the link table. For example, switch (4).

## connUnitLinkConnIdY (1.3.6.1.3.94.1.12.1.13)

This is the Fibre Channel ID of this port. If the connectivity unit is a switch, this is expected to be a Big Endian value of 24 bits. If this is loop, then it is the ALPA that is connected. If this is an E\_Port, then it will only contain the domain ID. If not any of those, unknown or cascaded loop, returns all bits set to 1.

### Syntax

OCTET STRING (SIZE(3))

### Access

read-only

### Status

mandatory

### Return Value

The remote Fibre Channel address of each entry in the link table.

## connUnitLinkCurrIndex (1.3.6.1.3.94.1.12.1.14)

The last used link index.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

The last used link table index number.

---

# Zone Table

The objects described in this section are in a table format indexed Zone number and Index. The zones are numbered 1 to `connUnitZoneSetNumZones`, the index represents the members within the zones.

An example of how to access one of these objects:

```
fcmgmt.connSet.connUnitZoneTable.connUnitZoneEntry.connUnitZoneIndex.1.1
```

## `connUnitZoneIndex` (1.3.6.1.3.94.1.13.1.1)

Unique table index for each zone. Valid values are between 1 and `connUnitZoneSetNumZones`.

### Syntax

INTEGER (1..2147483647)

### Access

read-only

### Status

mandatory

### Return Value

Returns index number for each zone within the active zoneset.

## `connUnitZoneMemberIndex` (1.3.6.1.3.94.1.13.1.2)

Unique table index for each zone member. Valid values are between 1 and `connUnitZoneNumMembers`.

### Syntax

INTEGER (1..2147483647)



## Access

read-only

## Status

mandatory

## Return Value

Returns index number for each member within a zone.

## connUnitZoneSetName (1.3.6.1.3.94.1.13.1.3)

Name of the active zone set to which the zone and zone member belong.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Returns the zone set name.

## connUnitZoneSetNumZones (1.3.6.1.3.94.1.13.1.4)

The number of zones in the active zone set.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

Returns the number of zones within the active zoneset.

## connUnitZoneName (1.3.6.1.3.94.1.13.1.5)

Name of the zone.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Returns the name of the zone.

## connUnitZoneCapabilities (1.3.6.1.3.94.1.13.1.6)

1-byte bit mask that specifies the zoning capabilities supported by the fabric.

Bit 7 - Soft zones supported.

Bit 6 - Hard zones supported.

Bits 5-0 - Reserved.

## Syntax

OCTET STRING (SIZE(1))

## Access

read-only

## Status

mandatory

## Return Value

Always returns 0xC0.

# connUnitZoneEnforcementState (1.3.6.1.3.94.1.13.1.7)

1-byte bit mask that specifies the current enforcement of the Zone Set.

Bit 7 - Soft zone set enforced.

Bit 6 - Hard zone set enforced.

Bits 5-0 - Reserved.

## Syntax

OCTET STRING (SIZE(1))

## Access

read-only

## Status

mandatory

## Return Value

Returns the zone type. Mapped as follows:

Soft.....0x80

Hard.....0x40

## connUnitZoneAttributeBlock (1.3.6.1.3.94.1.13.1.8)

A variable length structure that contains extended zone attributes defined in the FC-GS-4 enhanced zone server. See FC-GS-4 draft standard for details and format of the structure. Support of this object is optional.

### Syntax

OCTET STRING (SIZE(80))

### Access

read-only

### Status

mandatory

### Return Value

Not supported. Always returns SNMP error NoSuchName.

## connUnitZoneNumMembers (1.3.6.1.3.94.1.13.1.9)

Number of zone members in the zone: connUnitZoneName.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

Returns total number of members in a zone.

## connUnitZoneMemberIdType (1.3.6.1.3.94.1.13.1.10)

Type of zone member ID:

- 1- Port WWN
- 2- Domain & Port ID
- 3- FC Address
- 4- Node WWN
- 5- Alias Name
- 6-'FF'h - Vendor specified.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

Retrieves the member ID type.

WWN.....0x01 // Port WWN

Domain/Port....0x02 // Domain & Port ID

FCaddress.....0x03 // FC Address

[other].....0xff // Vendor specific

## connUnitZoneMemberID (1.3.6.1.3.94.1.13.1.11)

ID of the zone member based on connUnitZoneMemberIdType.

### Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

Returns the zone member name as a 16 8-bit octets. Mapped as follows:

WWN member - WWN (8 bytes) followed by 8 bytes of zeros.

FC address - FC address (3 bytes) followed by 13 bytes of zeros.

Domain/Port - Domain/Port address (2 bytes) followed by 14 bytes of zeros.

---

# Zoning Alias Table

The objects described in this section are in a table format indexed by Alias Number and Index. The aliases are numbered 1 to connUnitZoningAliasNumAliases, the index represents the members within the alias. An example of how to access one of these objects:

```
"fcmgmt.connSet.connUnitZoneTable.connUnitZoneEntry.connUnitZoningAliasIndex.1.1"
```

## connUnitZoningAliasIndex (1.3.6.1.3.94.1.14.1.1)

Unique table index for each alias. Valid values are between 1 and connUnitZoningAliasNumAliases.

## Syntax

INTEGER (1..2147483647)

## Access

read-only

## Status

mandatory

## Return Value

Returns the alias index.

# connUnitZoningAliasMemberIndex (1.3.6.1.3.94.1.14.1.2)

Unique table index for each alias member. Valid values are between 1 and connUnitZoningAliasNumMembers.

## Syntax

INTEGER (1..2147483647)

## Access

read-only

## Status

mandatory

## Return Value

Returns the alias member index.

# connUnitZoningAliasNumAliases (1.3.6.1.3.94.1.14.1.3)

The number of aliases defined in this table.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

Returns number of aliases defined.

# connUnitZoningAliasName (1.3.6.1.3.94.1.14.1.4)

The alias name.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Returns Alias name.

# connUnitZoningAliasNumMembers (1.3.6.1.3.94.1.14.1.5)

Number of members in the alias: connUnitZoningAliasName.

## Syntax

INTEGER

## Access



read-only

## Status

mandatory

## Return Value

Returns number of members in a defined Alias zone.

# connUnitZoningAliasMemberIdType (1.3.6.1.3.94.1.14.1.6)

Type of alias member ID:

1- Port WWN

2- Domain & Port ID

3- FC Address

Others: reserved.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

Returns the alias member Id type. Mapped as follows:

WWN..... 0x01 // Port WWN

DomainPort..... 0x02 // Domain & Port ID

FC Address..... 0x03 // FC Address

[other]..... 0xff // Vendor specific

## connUnitZoningAliasMemberID (1.3.6.1.3.94.1.14.1.7)

ID of the alias member based on connUnitZoningAliasMemberIdType.

### Syntax

FcGlobalId

### Access

read-only

### Status

mandatory

### Return Value

Returns the alias zone member name as 16 8-bit octets. Mapped as follows:

WWN member - WWN (8 bytes) followed by 8 bytes of zeros.

FC address - FC address (3 bytes) followed by 13 bytes of zeros.

Domain/Port - Domain/Port address (2 bytes) followed by 14 bytes of zeros.

---

## Port Statistics Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The index represents the port number to interrogate. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public  
fcmgmt.statSet.connUnitPortStatTable.connUnitPortStatEntry.connU  
nitPortStatUnitId.16.0.0.192.221.0.144.167.0.0.0.0.0.0.0.0.1".
```

There is one and only one statistics table for each individual port. For all objects in statistics table, if the object is not supported by the conn unit then the high order bit is set to 1 with all other bits set to zero. The high order bit is reserved to indicate if the object is supported or not. All objects start at a value of zero at hardware initialization and continue incrementing till end of 63 bits and then wrap to zero.

## connUnitPortStatUnitId (1.3.6.1.3.94.4.5.1.1)

A unique value among all entries in this table having the same `connUnitPortStatUnitId`, between 1 and `connUnitNumPort` [`connUnitPortStatUnitId`].

### Syntax

FcGlobalId

### Access

read-only

### Status

mandatory

### Return Value

Returns the World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00.

## connUnitPortStatIndex (1.3.6.1.3.94.4.5.1.2)

A unique value among all entries in this table, between 0 and `connUnitNumPort` [`connUnitPortUnitId`].

### Syntax

INTEGER (0..2147483647)

### Access

read-only

## Status

mandatory

## Return Value

The port table index.

# connUnitPortStatCountError (1.3.6.1.3.94.4.5.1.3)

A count of the errors that have occurred on this port.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal value indicating the total number of errors for a port.

# connUnitPortStatCountTxObjects (1.3.6.1.3.94.4.5.1.4)

The number of frames/packets/IOs/etc transmitted by this port. A Fibre Channel frame starts with SOF and ends with EOF. Fibre Channel loop devices should not count frames passed through. This value represents the sum total for all other Tx objects.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal value indicating the total number of bytes transmitted by a port.

# connUnitPortStatCountRxObjects (1.3.6.1.3.94.4.5.1.5)

The number of frames/packets/IOs/etc received by this port. A Fibre Channel frame starts with SOF and ends with EOF. Fibre Channel loop devices should not count frames passed through. This value represents the sum total for all other Rx objects.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal value indicating the total number of bytes received by a port.

# connUnitPortStatCountTxElements (1.3.6.1.3.94.4.5.1.6)

The number of octets or bytes that have been transmitted by this port. One second periodic polling of the port. This value is saved and compared with the next polled value to compute net throughput. For Fibre Channel, ordered sets are not included in the count.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal value indicating the total number of bytes transmitted by a port.

# connUnitPortStatCountRxElements (1.3.6.1.3.94.4.5.1.7)

The number of octets or bytes that have been received by this port. One second periodic polling of the port. This value is saved and compared with the next polled value to compute net throughput. For Fibre Channel, ordered sets are not included in the count.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal value indicating the total number of bytes received by a port.

## connUnitPortStatCountBBCreditZero (1.3.6.1.3.94.4.5.1.8)

Count of transitions in/out of BBcredit zero state. The other side is not providing any credit. This is a Fibre Channel statistic only.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountInputBuffersFull (1.3.6.1.3.94.4.5.1.9)

Count of occurrences when all input buffers of a port were full and outbound buffer-to-buffer credit transitioned to zero. There is no credit to provide to other side. This is a Fibre Channel statistic only.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountFBSYFrames (1.3.6.1.3.94.4.5.1.10)

Count of times that FBSY was returned to this port as a result of a frame that could not be delivered to the other end of the link. This occurs if either the fabric or the destination port is temporarily busy. Port can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic. This is the sum of all classes. If you cannot keep the by-class counters, then keep the sum counters.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

A hexadecimal number indicating the total number of FBusy on a port.

## connUnitPortStatCountPBSYFrames (1.3.6.1.3.94.4.5.1.11)

Count of times that PBSY was returned to this port as a result of a frame that could not be delivered to the other end of the link. This occurs if the destination port is temporarily busy. PBSY can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic. This is the sum of all classes. If you cannot keep the by-class counters, then keep the sum counters.

### Syntax

OCTET STRING (SIZE (8))



## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit set to 1 with remaining bits set to zero.

## connUnitPortStatCountFRJTFrames (1.3.6.1.3.94.4.5.1.12)

Count of times that FRJT was returned to this port as a result of a frame that was rejected by the fabric. This is the total for all classes and is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

A hexadecimal number indicating the total number of Frame Rejects on a port.

## connUnitPortStatCountPRJTFrames (1.3.6.1.3.94.4.5.1.13)

Count of times that FRJT was returned to this port as a result of a frame that was rejected at the destination N\_Port. This is the total for all classes and is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountClass1RxFrames (1.3.6.1.3.94.4.5.1.14)

Count of Class 1 frames received at this port. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass1TxFrames (1.3.6.1.3.94.4.5.1.15)

Count of Class 1 frames transmitted out this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass1FBSYFrames (1.3.6.1.3.94.4.5.1.16)

Count of times that FBSY was returned to this port as a result of a Class 1 frame that could not be delivered to the other end of the link. This occurs if either the fabric or the destination port is temporarily busy. FBSY can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

### connUnitPortStatCountClass1PBSYFrames (1.3.6.1.3.94.4.5.1.17)

Count of times that PBSY was returned to this port as a result of a Class 1 frame that could not be delivered to the other end of the link. This occurs if the destination N\_Port is temporarily busy. PBSY can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

### connUnitPortStatCountClass1FRJTFrames (1.3.6.1.3.94.4.5.1.18)

Count of times that FRJT was returned to this port as a result of a Class 1 frame that was rejected by the fabric. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass1PRJTFrames (1.3.6.1.3.94.4.5.1.19)

Count of times that FRJT was returned to this port as a result of a Class 1 frame that was rejected at the destination N\_Port. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass2RxFrames (1.3.6.1.3.94.4.5.1.20)

Count of Class 2 frames received at this port. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of Class 2 frames received by a port.

## connUnitPortStatCountClass2TxFrames (1.3.6.1.3.94.4.5.1.21)

Count of Class 2 frames transmitted out this port. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of Class 2 frames transmitted by a port.

## connUnitPortStatCountClass2FBSYFrames (1.3.6.1.3.94.4.5.1.22)

Count of times that FBSY was returned to this port as a result of a Class 2 frame that could not be delivered to the other end of the link. This occurs if either the fabric or the destination port is temporarily busy. FBSY can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountClass2PBSYFrames (1.3.6.1.3.94.4.5.1.23)

Count of times that PBSY was returned to this port as a result of a Class 2 frame that could not be delivered to the other end of the link. This occurs if the destination N\_Port is temporarily busy. PBSY can only occur on SOFc1 frames (the frames that establish a connection). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass2FRJTFrames (1.3.6.1.3.94.4.5.1.24)

Count of times that FRJT was returned to this port as a result of a Class 2 frame that was rejected by the fabric. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass2PRJTFrames (1.3.6.1.3.94.4.5.1.25)

Count of times that FRJT was returned to this port as a result of a Class 2 frame that was rejected at the destination N\_Port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value



Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountClass3RxFrames (1.3.6.1.3.94.4.5.1.26)

Count of Class 3 frames received at this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

The total number of Class 3 frames received by a port.

## connUnitPortStatCountClass3TxFrames (1.3.6.1.3.94.4.5.1.27)

Count of Class 3 frames transmitted out this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

## Return Value

The total number of Class 3 frames transmitted by a port.

## connUnitPortStatCountClass3Discards (1.3.6.1.3.94.4.5.1.28)

Count of Class 3 frames that were discarded upon reception at this port. There is no FBSY or FRJT generated for Class 3 frames. They are simply discarded if they cannot be delivered. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of Class3Toss frames for a port.

## connUnitPortStatCountRxMulticastObjects (1.3.6.1.3.94.4.5.1.29)

Count of Multicast frames or packets received at this port.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountTxMulticastObjects (1.3.6.1.3.94.4.5.1.30)

Count of Multicast frames or packets transmitted out this port.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountRxBroadcastObjects (1.3.6.1.3.94.4.5.1.31)

Count of Broadcast frames or packets received at this port.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountTxBroadcastObjects (1.3.6.1.3.94.4.5.1.32)

Count of Broadcast frames or packets transmitted out this port. On a Fibre Channel loop, count only OPNr frames generated.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountRxLinkResets (1.3.6.1.3.94.4.5.1.33)

Count of link resets. This is the number of LRs received. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of RxLinkResets received by a port.

# connUnitPortStatCountTxLinkResets (1.3.6.1.3.94.4.5.1.34)

Count of link resets. The number of LRs transmitted. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of TxLinkResets transmitted by a port.

# connUnitPortStatCountNumberLinkResets (1.3.6.1.3.94.4.5.1.35)

Count of link resets and LIPs detected at this port. The number of times the reset link protocol is initiated. These are the count of the logical resets, and a count of the number of primitives. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of TotalLinkResets for a port.

# connUnitPortStatCountRxOfflineSequences (1.3.6.1.3.94.4.5.1.36)

Count of offline primitive OLSs received at this port. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of RxOfflineSeqs received by a port.

## connUnitPortStatCountTxOfflineSequences (1.3.6.1.3.94.4.5.1.37)

Count of offline primitive OLSs transmitted by this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

The total number of TxOfflineSeqs transmitted by a port.

## connUnitPortStatCountNumberOfflineSequences (1.3.6.1.3.94.4.5.1.38)

Count of offline primitive sequences received at this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

The total number of TotalOfflineSeqs received by a port.

## connUnitPortStatCountLinkFailures (1.3.6.1.3.94.4.5.1.39)

Count of link failures. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

The total number of LinkFailures for a port.

## connUnitPortStatCountInvalidCRC (1.3.6.1.3.94.4.5.1.40)

Count of frames received with invalid CRC. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). Loop ports should not count CRC errors passing through when monitoring. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status



mandatory

## Return Value

The total number of InvalidCRCs received by a port.

## connUnitPortStatCountInvalidTxWords (1.3.6.1.3.94.4.5.1.41)

Count of invalid transmission words received at this port. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of DecodeErrors for a port.

## connUnitPortStatCountPrimitiveSequenceProtocol Errors (1.3.6.1.3.94.4.5.1.42)

Count of primitive sequence protocol errors detected at this port. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number of `PrimSeqErrors` for a port.

# connUnitPortStatCountLossofSignal (1.3.6.1.3.94.4.5.1.43)

Count of instances of signal loss detected at port. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountLossofSynchronization (1.3.6.1.3.94.4.5.1.44)

Count of instances of synchronization loss detected at port. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

The total number `LossOfSyncs` detected by this port.

# connUnitPortStatCountInvalidOrderedSets (1.3.6.1.3.94.4.5.1.45)

Count of invalid ordered sets received at port. This count is part of the Link Error Status Block (LESB). (FC-PH 29.8). This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountFramesTooLong (1.3.6.1.3.94.4.5.1.46)

Count of frames received at this port where the frame length was greater than what was agreed to in FLOGI/PLOGI. This could be caused by losing the end of frame delimiter. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

# connUnitPortStatCountFramesTruncated (1.3.6.1.3.94.4.5.1.47)

Count of frames received at this port where the frame length was less than the minimum indicated by the frame header (normally 24 bytes). It could be more if the DFCTL field indicates an optional header should have been present. This is a Fibre Channel-only statistic.

## Syntax

OCTET STRING (SIZE (8))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountAddressErrors (1.3.6.1.3.94.4.5.1.48)

Count of frames received with unknown addressing.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

The total number of InvalidDestAddr frames received by a port.

## connUnitPortStatCountDelimiterErrors (1.3.6.1.3.94.4.5.1.49)

Count of invalid frame delimiters received at this port. An example is a frame with a Class 2 start and a Class 3 at the end. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

## connUnitPortStatCountEncodingDisparityErrors (1.3.6.1.3.94.4.5.1.50)

Count of disparity errors received at this port. This is a Fibre Channel-only statistic.

### Syntax

OCTET STRING (SIZE (8))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns high order bit to 1 with all other bits set to zero.

---

## Simple Name Server Table

The objects described in this section are in a table format indexed by World Wide Name and Index. The index represents the table index. An example of how to access one of these objects given a WWN of 100000c0dd0090a7 is:

```
"snmpget localhost public  
fcmgmt.connUnitServiceSet.connUnitServiceTables.connUnitSnsTable  
.connUnitSnsEntry.connUnitSnsId.16.0.0.192.221.0.144.167.0.0.0.0  
.0.0.0.0.1".
```

The Fibre Channel Simple Name Server table contains an entry for each device presently known to this connUnit. There will not be any version on this since FC-GS3 does not define a version today.

This table is accessed either directly if the management software has an index value or using `GetNexts`. The value of the indexes are not required to be contiguous. Each entry created in this table will be assigned an index. This relationship is kept persistent until the entry is removed from the table or the system is reset. The total number of entries are defined by the size of the table.

## connUnitSnsMaxEntry (1.3.6.1.3.94.5.1.1)

The current number of entries in the table.

### Syntax

INTEGER

### MaxAccess

read-only

### Status

mandatory

### Return Value

Returns the number of entries registered in the Simple Name Server for all switches.

## connUnitSnsId (1.3.6.1.3.94.5.2.1.1.1)

The connUnitId of the connectivity unit that contains this Name Server table.

### Syntax

OCTET STRING (SIZE (16))

### Access

read-only

### Status

mandatory

### Return Value

Returns the World Wide Name of the switch followed by 8 bytes of zeros. For example: 10 00 00 C0 DD 00 71 C9 00 00 00 00 00 00 00 00.

## connUnitSnsPortIndex (1.3.6.1.3.94.5.2.1.1.2)

The physical port number of this SNS table entry. Each physical port has an SNS table with 1-n entries indexed by ConnUnitSnsPortIdentifier (port address).

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

The name server table index.

## connUnitSnsPortIdentifier (1.3.6.1.3.94.5.2.1.1.3)

The port identifier for this entry in the SNS table.

### Syntax

FcAddressId

### Access

read-only

### Status

mandatory

### Return Value

The 24-bit Fibre Channel address for each entry in the name server table based on Domain, Area, and ALPA.



## connUnitSnsPortName (1.3.6.1.3.94.5.2.1.1.4)

The Port World Wide Name for this entry in the SNS table.

### Syntax

FcNameId

### Access

read-only

### Status

mandatory

### Return Value

The Port World Wide Name of the device in the name server table.

## connUnitSnsNodeName (1.3.6.1.3.94.5.2.1.1.5)

The Node name for this entry in the SNS table.

### Syntax

FcNameId

### Access

read-only

### Status

mandatory

### Return Value

The Node World Wide Name of the device in the name server table.

## connUnitSnsClassOfSvc (1.3.6.1.3.94.5.2.1.1.6)

The classes of service offered by this entry in the SNS table.

### Syntax

OCTET STRING (SIZE (1))

### Access

read-only

### Status

mandatory

### Return Value

A value indicating the first registered class of service for an entry in the name server table. This is a bit mask where each bit that represents the class of service is set to a value of one if the class is supported. Class 1 is bit zero.

## connUnitSnsNodeIPAddress (1.3.6.1.3.94.5.2.1.1.7)

The IPv6 formatted address of the Node for this entry in the SNS table.

### Syntax

OCTET STRING (SIZE (16))

### Access

read-only

### Status

mandatory

### Return Value

The switch IP address in IPv6 format.

## connUnitSnsProcAssoc (1.3.6.1.3.94.5.2.1.1.8)

The process associator for this entry in the SNS table.

### Syntax

OCTET STRING (SIZE (16))

### Access

read-only

### Status

mandatory

### Return Value

Unsupported. Always returns error status "NoSuchName".

## connUnitSnsFC4Type (1.3.6.1.3.94.5.2.1.1.9)

The FC-4 types supported by this entry in the SNS table.

### Syntax

OCTET STRING (SIZE (32))

### Access

read-only

### Status

mandatory

### Return Value

A value indicating the FC-4 Types registered for the device in the name server table. This is a 32 byte field with each bit uniquely identifying the FC-4 Type registered as defined in FC-GS-3 specification. Example: SCSI FCP (bit 8) = 00 00 01 00.

# connUnitSnsPortType (1.3.6.1.3.94.5.2.1.1.10)

The port type of this entry in the SNS table.

## Syntax

OCTET STRING (SIZE (1))

## Access

read-only

## Status

mandatory

## Return Value

A value indicating the PortType for the entry in the name server table. Refer to [TABLE 4-24](#) for connUnitPortType port type return values.

**TABLE 4-24** ConnUnitPortType State Return Values

Port Type	Return Value (hexadecimal)
N	1
NL	2
F/NL	3
NX	7F
F	8
FL	82
E	84
B	85

# connUnitSnsPortIPAddress (1.3.6.1.3.94.5.2.1.1.11)

The IPv6 formatted address of this entry in the SNS table.

## Syntax

OCTET STRING (SIZE (16))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName".

## connUnitSnsFabricPortName (1.3.6.1.3.94.5.2.1.1.12)

The fabric port name of this entry in the SNS table.

## Syntax

FcNameId

## Access

read-only

## Status

mandatory

## Return Value

The switch port Port World Wide Name for the device in the name server table.

## connUnitSnsHardAddress (1.3.6.1.3.94.5.2.1.1.13)

The hard ALPA of this entry in the SNS table.

## Syntax

FcAddressId

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName".

# connUnitSnsSymbolicPortName (1.3.6.1.3.94.5.2.1.1.14)

The symbolic port name of this entry in the SNS table.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

The symbolic Port Name registered by the device in the name server table. If not registered, returns (NULL).

# connUnitSnsSymbolicNodeName (1.3.6.1.3.94.5.2.1.1.15)

The symbolic Node name of this entry in the SNS table.

## Syntax

DisplayString (SIZE (0..79))

#### Access

read-only

#### Status

mandatory

#### Return Value

The symbolic Node Name registered by the device in the name server table. If not registered, returns (NULL).

---

## Platform Table

The Platform Table is a simple, read-only view of platform registration entries. Platform registry is a service hosted by the connectivity unit, in a very similar manner as the SNS table. The platform table is contained by the connectivity unit. A platform can register its attributes and platform nodes with the registry service.

The platform table is a flat, double-indexed MIB table. To keep the table simple, only one platform management URL is exposed. If a platform registers more than one management URL, the first one is reported in this table. This table is based on the fabric configuration server defined in the FC-GS-3 standard and enhanced platform attributes proposed for FC-GS-4. Note that the information contained in this table may only contain the platforms that this connUnit can see or it may contain a fabric wide view of the platforms.

### connUnitPlatformMaxEntry (1.3.6.1.3.94.5.1.2)

The maximum number of entries in the platform table.

#### Syntax

INTEGER

#### Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

# connUnitPlatformIndex (1.3.6.1.3.94.5.2.2.1.1)

Unique table index for each platform. Valid values are between 1 and connUnitPlatformsMaxEntry.

## Syntax

INTEGER (1..2147483647)

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

# connUnitPlatformNodeIndex (1.3.6.1.3.94.5.2.2.1.2)

Unique table index for each platform node. Valid values are between 1 and connUnitPlatformsNumNodes.

## Syntax

INTEGER (1..2147483647)

## Access

read-only



## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformUnitID (1.3.6.1.3.94.5.2.2.1.3)

The connUnitId of the connectivity unit that contains this Platform table.

## Syntax

FcGlobalId

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformName (1.3.6.1.3.94.5.2.2.1.4)

The platform name. May be either a readable string or a unique ID format as specified in the FC-GS-4 draft standard.

## Syntax

OCTET STRING (SIZE(79))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformType (1.3.6.1.3.94.5.2.2.1.6)

The platform type.

## Syntax

FcUnitType

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformLabel (1.3.6.1.3.94.5.2.2.1.7)

An administratively assigned symbolic name for the platform. The Platform Label shall only contain print-able ASCII characters.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformDescription (1.3.6.1.3.94.5.2.2.1.8)

A textual description of the platform. This value should include the full name and version identification of the platform's hardware type and software operating system. The Platform Description shall only contain printable ASCII characters.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformLocation (1.3.6.1.3.94.5.2.2.1.9)

The physical location of the platform (e.g., telephone closet, 3rd floor). The Platform Location shall only contain printable ASCII characters.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformManagementUrl (1.3.6.1.3.94.5.2.2.1.10)

Primary management URL for the platform. If the platform registers more than one URL, then this URL is equal to the first in the list.

## Syntax

DisplayString (SIZE (0..79))

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformNumNodes (1.3.6.1.3.94.5.2.2.1.11)

Number of nodes contained in the platform.

## Syntax

INTEGER

## Access

read-only

## Status

mandatory

## Return Value

Unsupported. Always returns error status "NoSuchName"

## connUnitPlatformNodeName (1.3.6.1.3.94.5.2.2.1.12)

The name (WWN - world wide name) of the node contained by the platform.

## Syntax

FcGlobalId

## Access

read-only

## Status

read-only

## Return Value

Unsupported. Always returns error status "NoSuchName"

---

# Trap Table

Traps are asynchronous messages sent from the agent (residing on the switch) to the manager (residing on the workstation) to identify significant events.

There can be up to 5 trap addresses within the trap table. All trap information is stored within the switch and is accessible to Telnet and the SNMP agent, and is persistent between boots. An example of how to access one of these objects given an IP address of 10.32.165.4 is:

```
"snmpget localhost public
fcmgmt.trapReg.trapRegTable.trapRegEntry.trapRegFilter.10.32.165
.4.162".
```

A trap event is reported when the incoming error has a severity level less than or equal to the configured severity level. The trap event types and trap severity levels are listed in [TABLE 4-25](#).

**TABLE 4-25** Trap Severity Levels

Event Type	Severity Level
Unknown	1
Emergency	2
Alert	3
Critical	4
Error	5
Warning	6
Notify	7
Info	8
Debug	9
Mark	10

## trapMaxClients (1.3.6.1.3.94.2.1)

The maximum number of SNMP trap recipients supported by the connectivity unit.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

Always returns 5.

## trapClientCount (1.3.6.1.3.94.2.2)

The current number of rows in the trap table.

### Syntax

INTEGER

### Access

read-only

### Status

mandatory

### Return Value

A value (1-5) indicating number of configured trap clients.

## trapRegIpAddress (1.3.6.1.3.94.2.3.1.1)

The IP address of a client registered for traps.

### Syntax

IpAddress

### Access

read-only

### Status

mandatory

### Return Value

The IP addresses (as defined in the trap table) of where to send traps when they occur.

## trapRegPort (1.3.6.1.3.94.2.3.1.2)

The UDP port to send traps to for this host. Normally this would be the standard trap port (162). This object is an index and must be specified to create a row in this table.

### Syntax

INTEGER (1..2147483647)

### Access

read-only

### Status

mandatory

### Return Value

The configured port number of where to send traps when they occur. The port number can be configured in the switch SNMP setup parameters. Default is 162.

## trapRegFilter (1.3.6.1.3.94.2.3.1.3)

This value defines the trap severity filter for this trap host. The connUnit will send traps to this host that have a severity level less than or equal to this value. The default value of this object is "warning".

### Syntax

FcEventSeverity

### Access

read-write

### Status

mandatory

### Return Value



A value indicating the trap severity level. Refer to [TABLE 4-25](#) for trap severity levels.

## trapRegRowState (1.3.6.1.3.94.2.3.1.4)

Specifies the state of the row.

- rowDestroy
  - READ: Can never happen.
  - WRITE: Remove this row from the table.
- rowInactive
  - READ: Indicates that this row does exist, but that traps are not enabled to be sent to the target.
  - WRITE: If the row does not exist, and the agent allows writes to the trap table, then a new row is created. The values of the optional columns will be set to default values. Traps are not enabled to be sent to the target. If the row already existed, then traps are disabled from being sent to the target.
- rowActive
  - READ: Indicates that this row exists, and that traps are enabled to be sent to the target.
  - WRITE: If the row does not exist, and the agent allows writes to the trap table, then a new row is created. The values of the optional columns will be set to default values. Traps are enabled to be sent to the target. If the row already exists, then traps are enabled to be sent to the target.

A value of “rowActive” or “rowInactive” must be specified to create a row in the table.

### Syntax

```
INTEGER {  
    rowDestroy(1), - Remove row from table.  
    rowInactive(2), - Row exists, but traps disabled  
    rowActive(3) - Row exists and is enabled for sending traps  
}
```

### Access

read-write

## Status

mandatory

## Return Value

Returns rowActive (3), if valid entry in trap table.

---

# Related Traps

The following traps contain the trap information being sent from the agent to the manager.

## connUnitStatusChange (1.3.6.1.3.94.0.1)

The overall status of the connectivity unit has changed. The recommended severity level (for filtering) is “alert”. Sent whenever a `Switch.OperChange` or `Switch.StateChange` event occurs.

Variables: { *connUnitStatus*, *connUnitState* }

## connUnitDeletedTrap (1.3.6.1.3.94.0.3)

A `connUnit` has been deleted from this agent. The recommended severity level (for filtering) is “warning”. Sent whenever an `Eport.OperChange` event occurs and the `connUnitTable` is smaller than previously noted (A `connUnit` has gone away).

Variables: { *connUnitId* }

## connUnitEventTrap (1.3.6.1.3.94.0.4)

An event has been generated by the connectivity unit. The recommended severity level (for filtering) is “info”. Sent when a change notification occurs that does not fit into any other specific category.

Variables:

{ *connUnitEventId*, *connUnitEventType*, *connUnitEventObject*, *connUnitEventDescr* }

FIGURE 4-1 provides the standard format of the `connUnitEventDescr` variable. Chassis, Blade, and Port are always 0.

FIGURE 4-1 `connUnitEventDescr` Variable Format

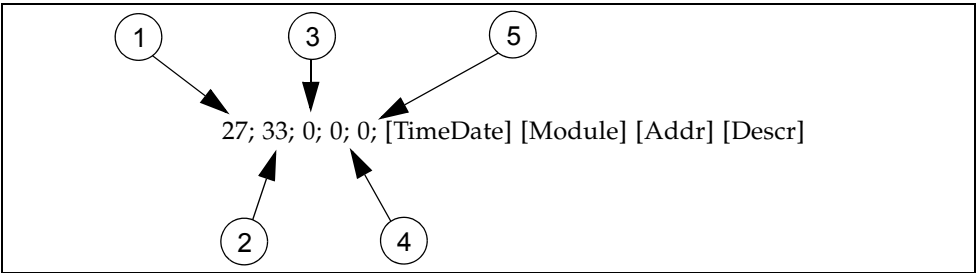


Figure Legend

1	Tag #
2	Event #
3	Chassis
4	Blade
5	Port

TABLE 4-26 lists the fields in the `connUnitEventDescr` variable.

TABLE 4-26 `connUnitEventDescr` Variable Field Descriptions

<code>connUnitEventDescr</code> Variable	Description
Tag #	The number that identifies the event.
Event #	The event counter.
Chassis	The switch on which the event occurred.
Blade	The I/O blade on which the event occurred.
Port	The port on which the event occurred.
TimeDate	The time stamp of the event.
Module	The software module where the event was initiated.
Addr	The address in the software module where the event was initiated.
Descr	The description of the event.

TABLE 4-27 lists the trap information returned for the `connUnitEventDescr` variable.

**TABLE 4-27** Filter Trap Levels

Trap Type	Cause	Filter Level
connUnitPortStatusChange	User port config change	eventSeverity_info
	User port state change	eventSeverity_info
	E_Port alarm	eventSeverity_critical
connUnitDeletedTrap	Fabric change and unit deleted	eventSeverity_info
connUnitStatusChange	Switch state change	eventSeverity_info
	Switch reset	eventSeverity_critical
connUnitSensorStatusChange	Power supply bad alarm	eventSeverity_critical
	Power supply OK alarm	eventSeverity_critical
	Fan bad alarm	eventSeverity_critical
	Fan OK alarm	eventSeverity_critical
	Overheat alarm	eventSeverity_critical
	Overwarm alarm	eventSeverity_critical
	Temperature OK alarm	eventSeverity_critical
connUnitEventTrap	SNMP config change	eventSeverity_info
	Switch config change	eventSeverity_info
	System config change	eventSeverity_info
	Topology change	eventSeverity_info
	Zoning change	eventSeverity_info
	Zoning merge failure	eventSeverity_critical
	NameServer change	eventSeverity_info
	Generic alarm	eventSeverity_critical
	Generic event	eventSeverity_warning

## connUnitSensorStatusChange (1.3.6.1.3.94.0.5)

The overall status of the connectivity unit has changed. The recommended severity level (for filtering) is “alert”. Sent whenever any of the following notifications occur:

- `Chassis.PsBadAlarm`
- `Chassis.PsOkAlarm`
- `Chassis.FanBadAlarm`
- `Chassis.FanOkAlarm`
- `Blade.OverheatAlarm`
- `Blade.OverwarmAlarm`

- Variables: { *connUnitSensorStatus* }

## connUnitPortStatusChange (1.3.6.1.3.94.0.6)

The overall status of the connectivity unit has changed. The recommended severity level (for filtering) is “alert”. Sent whenever a `UserPort.StateChange` or `UserPort.OperChange` event occurs.

Enterprise: `fcmgmt`

Variables: { *connUnitPortStatus*, *connUnitPortState* }

## coldStart

A `coldStart` trap signifies that the SNMPv2 entity, acting in an agent role, is re-initializing itself and that its configuration may have been altered.

## authenticationFailure

An `authenticationFailure` trap signifies that the SNMPv2 entity, acting in an agent role, has received a protocol message that is not properly authenticated. While all implementations of the SNMPv2 must be capable of generating this trap, the `snmpEnableAuthenTraps` object indicates whether this trap will be generated.



# Fabric Element MIB Objects

---

This section covers the implementation details for the Fabric Element Management Information Bases (FE-MIB) on the switch.

---

## Fibre Channel FE MIB Definitions

The textual substitutions in [TABLE 5-1](#) are specific to the FE-MIB and can be used in place of primitive data types.

**TABLE 5-1** FA-MIB Textual Substitutions

Description	Syntax
MilliSeconds	Unsigned32
MicroSeconds	Unsigned32
FcNameId	OCTET STRING (SIZE (8))
FcAddressId	OCTET STRING (SIZE (3))
FcRxDataFieldSize	Integer32 (128..2112)
FcBbCredit	Integer32 (0..32767)
FcphVersion	Integer32 (0..255)
FcStackedConnMode	INTEGER { none(1), transparent(2), lockedDown(3) }

**TABLE 5-1** FA-MIB Textual Substitutions

Description	Syntax
FcCosCap	BITS { classF(0), class1(1), class2(2), class3(3), class4(4), class5(5), class6(6) }
FcCosCap	BITS { classF(0), class1(1), class2(2), class3(3), class4(4), class5(5), class6(6) }
FcFeModuleCapacity	Unsigned32
FcFeFxpPortCapacity	Unsigned32
FcFeModuleIndex	Unsigned32
FcFeFxpPortIndex	Unsigned32
FcFeNxPortIndex	Integer32 (1..126)
FcBbCreditModel	INTEGER {regular(1), alternate (2)}

# Configuration Group

This group consists of scalar objects and tables. It contains the configuration and service parameters of the Fabric Element and the FxPorts. The group represents a set of parameters associated with the Fabric Element or an FxPort to support its NxPorts. The objects described in this section are not in a table format. An example of how to access one of these objects is:

```
"snmpget localhost public fcFeFabricName.0".
```



## fcFeFabricName (1.3.6.1.2.1.75.1.1.1)

The Name\_Identifier of the Fabric to which this Fabric Element belongs.

### Syntax

FcNameId

### Access

read-write

### Status

Current

### Return Value

The World Wide Name of the principal switch. For example, 10 00 00 C0 DD 00 71 C2. Writes are not supported.

## fcFeElementName (1.3.6.1.2.1.75.1.1.2)

The Name\_Identifier of the Fabric Element.

### Syntax

FcNameId

### Access

read-write

### Status

Current

### Return Value

The World Wide Name of the switch. For example, 10 00 00 C0 DD 00 71 C9. Writes are not supported.

## fcFeModuleCapacity (1.3.6.1.2.1.75.1.1.3)

The maximum number of modules in the Fabric Element, regardless of their current state.

### Syntax

`FcFeModuleCapacity`

### Access

read-only

### Status

Current

### Return Value

The total number of switches in the fabric if ProxyEnable setting is Enabled on the out-of-band switch. If ProxyEnable setting is disabled on the out-of-band switch, return value = 1.

---

## Module Table

The objects described in this section are in table format indexed by switch. An example of how to access one of these objects is: `"snmpget localhost public fcFeModuleDescr.1"`. This table contains one entry for each module.

## fcFeModuleDescr (1.3.6.1.2.1.75.1.1.4.1.2)

A textual description of the module. This value should include the full name and version identification of the module.

### Syntax

`SnmpAdminString`

### Access

read-only

## Status

current

## Return Value

A configuration description of the module table entry.  
The default is Sun Storage FC Switch 5802.

## fcFeModuleObjectID (1.3.6.1.2.1.75.1.1.4.1.3)

The vendor's authoritative identification of the module. This value may be allocated within the SMI enterprises subtree (1.3.6.1.4.1), and provides a means for determining what kind of module is being managed.

For example, this object could take the value 1.3.6.1.4.1.99649.3.9 if vendor "Neufe Inc." was assigned the subtree 1.3.6.1.4.1.99649, and had assigned the identifier 1.3.6.1.4.1.99649.3.9 to its FeFiFo-16 PlugInCard.

## Syntax

OBJECT IDENTIFIER

## Access

read-only

## Status

current

## Return Value

The module identification number is 1.3.6.1.4.1.42.2.209.

## fcFeModuleOperStatus (1.3.6.1.2.1.75.1.1.4.1.4)

Switch definitions map 1-to-1 with the MIB definitions. This object indicates the operational status of the module.

- online (1) - the module is functioning properly

- offline (2) - the module is not available
- testing (3) - the module is under testing
- faulty (4) - the module is defective in some way

## Syntax

```
INTEGER {
  online(1), - functional
  offline(2), - not available
  testing(3), - under testing
  faulty(4) - defective
}
```

## Access

read-only

## Status

Current

## Return Value

The operational status of that module.

**TABLE 5-2** Module Operational Status Return Values

Mode	Return Value
online	online(1)
offline	offline(2)
diagnostics	testing(3)
other	faulty(4)

## fcFeModuleLastChange (1.3.6.1.2.1.75.1.1.4.1.5)

This object contains the value of `sysUpTime` when the module entered its current operational status. A value of zero indicates that the operational status of the module has not changed since the agent last restarted.

## Syntax

TimeStamp

## Access

read-only

## Status

Current

## Return Value

Unsupported. Always returns error status "NoSuchName".

# fcFeModuleFxPortCapacity (1.3.6.1.2.1.75.1.1.4.1.6)

The number of FxPort that can be contained within the module. Within each module, the ports are uniquely numbered in the range from 1 to fcFeModuleFxPortCapacity inclusive. However, the numbers are not required to be contiguous.

## Syntax

FcFeFxPortCapacity

## Access

read-only

## Status

current

## Return Value

The total number of ports capability on the switch. Sun Storage Fibre Channel Switch 5802 = 1 - 8, 12, 16, 20 (varies depending on number of licensed ports)

## fcFeModuleName (1.3.6.1.2.1.75.1.1.4.1.7)

The Name\_Identifier of the switch.

### Syntax

FcNameId

### Access

read-write

### Status

current

### Return Value

The World Wide Name of the switch. Writes are not supported. For example, 10 00 00 C0 DD 00 71 C9.

---

## FxPort Configuration Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `snmpget localhost public fcFxPortName.1.1`. This table contains one entry for each FxPort and Configuration parameters of the ports. This table contains, one entry for each FxPort, configuration parameters of the ports.

## fcFxPortName (1.3.6.1.2.1.75.1.1.5.1.2)

The World Wide Name of this FxPort. Each FxPort has a unique Port World Wide Name within the Fabric.

### Syntax

FcNameId

### Access

read-only

## Status

current

## Return Value

Returns the Port World Wide Name for each port on switch. For example, the return value for port #2 would be 20 02 00 C0 DD 00 71 C9, and the return value for port #14 would be 20 0E 00 C0 DD 00 71 C9.

## fcFxpPortFcphVersionHigh (1.3.6.1.2.1.75.1.1.5.1.3)

The highest or most recent version of FC-PH that the FxPort is configured to support.

## Syntax

FcphVersion

## Access

read-only

## Status

Current

## Return Value

Always returns 32 (0x20).

## fcFxpPortFcphVersionLow (1.3.6.1.2.1.75.1.1.5.1.4)

The lowest or earliest version of FC-PH that the FxPort is configured to support.

## Syntax

FcphVersion

## Access

read-only

## Status

current

## Return Value

Always returns 9.

# fcFxPortBbCredit (1.3.6.1.2.1.75.1.1.5.1.5)

The total number of receive buffers available for holding Class 1 connect-request, Class 2, or Class3 frames from the attached NxPort. It is for buffer-to-buffer flow control in the direction from the attached NxPort (if applicable) to FxPort.

## Syntax

FcBbCredit

## Access

read-only

## Status

current

## Return Value

The default number of receive buffers for each port, unless extended credits are used. The default is 16.

# fcFxPortRxBufSize (1.3.6.1.2.1.75.1.1.5.1.6)

The largest Data\_Field Size (in octets) for an FT\_1 frame that can be received by the FxPort.

## Syntax



FcRxDataFieldSize

## Access

read-only

## Status

current

## Return Value

Always returns 2112 (0x840).

## fcFxpPortRatov (1.3.6.1.2.1.75.1.1.5.1.7)

The `Resource_Allocation_Timeout` Value configured for the FxPort. This is used as the timeout value for determining when to reuse an NxPort resource such as a `Recovery_Qualifier`. It represents `E_D_TOV` plus twice the maximum time that a frame may be delayed within the fabric and still be delivered. Refer to [“fcFxpPortEdtov \(1.3.6.1.2.1.75.1.1.5.1.8\)” on page 231](#) for more information.

## Syntax

MilliSeconds

## Access

read-only

## Status

Current

## Return Value

The default is: 10000 (0x2710).

## fcFxpPortEdtov (1.3.6.1.2.1.75.1.1.5.1.8)

The `E_D_TOV` value configured for the FxPort. The `Error_Detect_Timeout` Value is used as the timeout value for detecting an error condition.

## Syntax

Milliseconds

## Access

read-only

## Status

current

## Return Value

The default is: 2000 (0x7D0).

# fcFxpPortCosSupported (1.3.6.1.2.1.75.1.1.5.1.9)

A value indicating the set of classes of service supported by the FxPort.

## Syntax

FcCosCap

## Access

read-only

## Status

Current

## Return Value

Always returns Class 3, 2, and F (0x0D).

# fcFxpPortIntermixSupported (1.3.6.1.2.1.75.1.1.5.1.10)

A flag indicating whether or not the FxPort supports an Intermixed Dedicated Connection.

## Syntax

TruthValue

## Access

read-only

## Status

current

## Return Value

Always returns `False` (2).

# fcFxPortStackedConnMode (1.3.6.1.2.1.75.1.1.5.1.11)

A value indicating the mode of Stacked Connect supported by the FxPort.

## Syntax

FcStackedConnMode

## Access

read-only

## Status

current

## Return Value

Always returns `None` (1).

# fcFxPortClass2SeqDeliv (1.3.6.1.2.1.75.1.1.5.1.12)

A flag indicating whether or not Class 2 Sequential Delivery is supported by the FxPort.

## Syntax

TruthValue

## Access

read-only

## Status

current

## Return Value

Always returns True (1).

# fcFxpPortClass3SeqDeliv (1.3.6.1.2.1.75.1.1.5.1.13)

A flag indicating whether or not Class 3 Sequential Delivery is supported by the FxPort.

## Syntax

TruthValue

## Access

read-only

## Status

current

## Return Value

Always returns True (1).

# fcFxpPortHoldTime (1.3.6.1.2.1.75.1.1.5.1.14)

The maximum time, in microseconds, that the FxPort shall hold a frame before discarding the frame if it is unable to deliver the frame. The value 0 means that the FxPort does not support this parameter.

## Syntax

MicroSeconds

## Access

read-only

## Status

current

## Return Value

The default ED\_TOV parameter is: 2000 (0x7D0).

---

# The Status Group

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: “snmpget localhost public fcFxPortId.1.1”. This group consists of tables that contain operational status and established service parameters for the Fabric Element and the attached NxPorts.

This group consists of tables that contains operational status and established service parameters for the Fabric Element and the attached NxPorts. This table contains one entry for each FxPort, and the operational status and parameters of the FxPorts.

## fcFxPortID (1.3.6.1.2.1.75.1.2.1.1.1)

The address identifier by which this FxPort is identified within the fabric. The FxPort may assign its address identifier to its attached NxPort(s) during Fabric Login.

## Syntax

FcAddressId

## Access

read-only

## Status

current

## Return Value

The address of each port based on Domain, Area, and ALPA. Example, 64 03 00.

## fcFxpPortBbCreditAvailable (1.3.6.1.2.1.75.1.2.1.1.2)

The number of buffers currently available for receiving frames from the attached port in the buffer-to-buffer flow control. The value should be less than or equal to `fcFxpPortBbCredit`.

## Syntax

Gauge32

## Access

read-only

## Status

Current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxpPortOperMode (1.3.6.1.2.1.75.1.2.1.1.3)

The current operational mode of the FxPort.

## Syntax

INTEGER {*unknown*(1), *fPort*(2), *flPort*(3)}

## Access

read-only

## Status

current

Return Value

Refer to [TABLE 5-3](#) for `fcFxpPortOperMode` return values.

**TABLE 5-3** Port Operational Modes

Mode	Return Value
Unknown	1
F_Port	2
FL_Port	3

`fcFxpPortAdminMode` (1.3.6.1.2.1.75.1.2.1.1.4)

The desired operational mode of the FxPort.

Syntax

INTEGER {*fPort*(2), *flPort*(3)}

Access

read-write

Status

Current

Return Value

Unsupported. Always returns error status 'NoSuchName'.

---

# FxPort Physical Level Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `“snmpget localhost public fcFxPortPhysAdminStatus.1.1”`. This table contains one entry for each FxPort in the Fabric Element, the physical level status, and parameters of the FxPorts.

This table contains one entry for each FxPort in the Fabric Element, and the physical level status and parameters of the FxPorts.

## fcFxPortPhysAdminStatus (1.3.6.1.2.1.75.1.2.2.1.1)

The desired state of the FxPort. A management station may place the FxPort in a desired state by setting this object accordingly. The testing(3) state indicates that no operational frames can be passed. When a Fabric Element initializes, all FxPorts start with `fcFxPortPhysAdminStatus` in the offline(2) state. As the result of either explicit management action or per configuration information accessible by the Fabric Element, `fcFxPortPhysAdminStatus` is then changed to either the online(1) or testing(3) states, or remains in the offline state.

### Syntax

```
INTEGER {  
    online(1), - place port online  
    offline(2), - take port offline  
    testing(3) - initiate test procedures  
}
```

### Access

read-write

### Status

current

### Return Value



Refer to [TABLE 5-4](#) for `fcFxpPortPhysAdminStatus` read values.

**TABLE 5-4** `fcFxpPortPhysAdminStatus` Read Return Values

Port	Value
Online	online (1)
Offline	offline (2)
Diagnostics	testing (3)
Down	offline (2)

Refer to [TABLE 5-5](#) for `fcFxpPortPhysAdminStatus` write values.

**TABLE 5-5** `fcFxpPortPhysAdminStatus` Write Values

Port	Value
Online	online (1)
Offline	offline (2)
Diagnostics	testing (3)

## `fcFxpPortPhysOperStatus` (1.3.6.1.2.1.75.1.2.2.1.2)

The current operational status of the FxPort. The testing(3) indicates that no operational frames can be passed. If `fcFxpPortPhysAdminStatus` is offline(2), then `fcFxpPortPhysOperStatus` should be offline(2). If `fcFxpPortPhysAdminStatus` is changed to online(1), then `fcFxpPortPhysOperStatus` should change to online(1). If the FxPort is ready to accept Fabric Login request from the attached NxPort, it should proceed and remain in the link- failure(4) state if, and only if, there is a fault that prevents it from going to the online(1) state.

### Syntax

```
INTEGER {  
    online(1), - Login may proceed  
    offline(2), - Login cannot proceed  
    testing(3), - port is under test  
    linkFailure(4) - failure after online/testing  
}
```

Access

read-only

Status

current

Return Value

Refer to [TABLE 5-6](#) for fcFxPortPhysOperStatus return values.

**TABLE 5-6** fcFxPortPHysOperStatus Return Values

Status	Return Value
Online	online (1)
Offline	offline (2)
Diagnostic	testing (3)
Down	linkfailure (4)

fcFxPortPhysLastChange (1.3.6.1.2.1.75.1.2.2.1.3)

The value of sysUpTime at the time the FxPort entered its current operational status. A value of zero indicates that the FxPort's operational status has not changed since the agent last restarted.

Syntax

TimeStamp

Access

read-only

Status

current

Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortPhysRttov (1.3.6.1.2.1.75.1.2.2.1.4)

The `Receiver_Transmitter_Timeout` value of the `FxPort`. This is used by the receiver logic to detect a loss of synchronization.

### Syntax

Milliseconds

### Access

read-write

### Status

current

### Return Value

The default `RT_TOV` parameter is: 100 (0x64). This is a global setting for the switch. If writing value to a port, all ports will reflect this new value.

---

## Fx Port Fabric Login Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `“snmpget localhost public fcFxPortFcphVersionAgreed.1.1”`. This table contains one entry for each `FxPort` in the fabric element and the service parameters that have been established from the most recent Fabric Login (implicit or explicit).

This table contains one entry for each `FxPort` in the fabric element, and the service parameters that have been established from the most recent Fabric Login, implicit or explicit.

## fcFxPortFcphVersionAgreed (1.3.6.1.2.1.75.1.2.3.1.2)

The version of FC-PH that the `FxPort` has agreed to support from the Fabric Login.

## Syntax

FcphVersion

## Access

read-only

## Status

current

## Return Value

Unsupported.

## fcFxpPortNxPortBbCredit (1.3.6.1.2.1.75.1.2.3.1.3)

The total number of buffers available for holding class 1 connect-request, class 2, or class 3 frames to be transmitted to the attached NxPort. It is for buffer-to-buffer flow control in the direction from FxPort to NxPort. The buffer-to-buffer flow control mechanism is indicated in the respective fcFxpPortBbCreditModel.

## Syntax

FcBbCredit

## Access

read-only

## Status

current

## Return Value

Unsupported.

## fcFxPortNxPortRxDataFieldSize (1.3.6.1.2.1.75.1.2.3.1.4)

The Receive Data Field Size of the attached NxPort. This object specifies the largest Data Field Size for an FT\_1 frame that can be received by the NxPort.

### Syntax

FcRxDataFieldSize

### Access

read-only

### Status

current

### Return Value

Unsupported.

## fcFxPortCosSuppAgreed (1.3.6.1.2.1.75.1.2.3.1.5)

A variable indicating that the attached NxPort has requested the FxPort for the support of classes of services and the FxPort has granted the request.

### Syntax

FcCosCap

### Access

read-only

### Status

current

### Return Value

The bits have the following bit-mapped definition:

Bit 7 Class-six

Bit 6 Class-five

Bit 5 Class-four

Bit 4 Class-three

Bit 3 Class-two

Bit 2 Class-one

Bit 1 Class F

For example: If Class 3, return value 0x10.

## fcFxPortIntermixSuppAgreed (1.3.6.1.2.1.75.1.2.3.1.6)

A variable indicating that the attached NxPort has requested the FxPort for the support of Intermix and the FxPort has granted the request. This flag is only valid if Class 1 service is supported.

### Syntax

TruthValue

### Access

read-only

### Status

current

### Return Value

Always returns `false` (2).

## fcFxPortStackedConnModeAgreed (1.3.6.1.2.1.75.1.2.3.1.7)

A variable indicating whether the FxPort has agreed to support stacked connect from the Fabric Login. This is only meaningful if the ports are using Class 1 service.

### Syntax

FcStackedConnMode

### Access

read-only

### Status

current

### Return Value

Always returns none (1).

## fcFxPortClass2SeqDelivAgreed (1.3.6.1.2.1.75.1.2.3.1.8)

A variable indicating whether the FxPort has agreed to support Class 2 sequential delivery from the Fabric Login. This is only meaningful if the ports are using Class 2 service.

### Syntax

TruthValue

### Access

read-only

### Status

Current

### Return Value

Always returns `true` (1).

## fcFxpPortClass3SeqDelivAgreed (1.3.6.1.2.1.75.1.2.3.1.9)

A flag indicating whether the FxpPort has agreed to support Class 3 sequential delivery from the Fabric Login. This is only meaningful if the ports are using Class 3 service.

### Syntax

TruthValue

### Access

read-only

### Status

current

### Return Value

Always returns `true` (1).

## fcFxpPortNxPortName (1.3.6.1.2.1.75.1.2.3.1.10)

The port name of the attached NxPort.

### Syntax

FcNameId

### Access

read-only

### Status

Current



## Return Value

Returns the Switch Port's Port World Wide Name for the attached device.

## fcFxpPortConnectedNxPort (1.3.6.1.2.1.75.1.2.3.1.11)

The address identifier of the destination NxPort with which this FxPort is currently engaged in a either a Class 1 or loop connection. If this FxPort is not engaged in a connection, then the value of this object is "000000"H.

## Syntax

FcAddressId

## Access

read-only

## Status

Current

## Return Value

Unsupported.

## fcFxpPortBbCreditModel (1.3.6.1.2.1.75.1.2.3.1.12)

This object identifies the BB\_Credit model used by the FxPort.

## Syntax

FcBbCreditModel

## Access

read-write

## Status

current

## Return Value

Returns `alternate (2)`. Writes not supported.

---

# The Error Group

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `"snmpget localhost public fcFxpPortLinkFailures.1.1"`. This group consists of tables that contain information about the various types of errors detected. The management station may use the information in this group to determine the quality of the link between the FxPort and its attached NxPort.

The FxPort Error table contains, one entry for each FxPort in the Fabric Element, counters recording numbers of errors detected since the management agent re-initialized. The first 6 columnar objects after the port index corresponds to the counters in the Link Error Status Block.

## fcFxpPortLinkFailures (1.3.6.1.2.1.75.1.3.1.1.1)

The number of link failures detected by this FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

## Return Value

The total number of `LinkFailures` encountered for a port.

## fcFxpPortSyncLosses (1.3.6.1.2.1.75.1.3.1.1.2)

The number of loss of synchronizations detected by the FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of LossOfSyncs encountered for a port.

## fcFxpPortSigLosses (1.3.6.1.2.1.75.1.3.1.1.3)

The number of loss of signals detected by the FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortPrimSeqProtoErrors (1.3.6.1.2.1.75.1.3.1.1.4)

The number of primitive sequence protocol errors detected by the FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `PrimSeqErrors` encountered for a port.

## fcFxPortInvalidTxWords (1.3.6.1.2.1.75.1.3.1.1.5)

The number of invalid transmission words detected by the FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `DecodeErrors` encountered for a port.

## fcFxpPortInvalidCrcs (1.3.6.1.2.1.75.1.3.1.1.6)

The number of invalid CRCs detected by this FxpPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of InvalidCRCs encountered for a port.

## fcFxpPortDelimiterErrors (1.3.6.1.2.1.75.1.3.1.1.7)

The number of Delimiter Errors detected by this FxpPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxpPortAddressIdErrors (1.3.6.1.2.1.75.1.3.1.1.8)

The number of address identifier errors detected by this FxpPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `InvDestAddrs` encountered for a port.

## fcFxpPortLinkResetIns (1.3.6.1.2.1.75.1.3.1.1.9)

The number of Link Reset Protocols received by this FxpPort from the attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `RxLinkResets` received by a port.

## fcFxpPortLinkResetOuts (1.3.6.1.2.1.75.1.3.1.1.10)

The number of Link Reset Protocols issued by this FxpPort to the attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of TxLinkResets sent by a port.

## fcFxpPortOlsIns (1.3.6.1.2.1.75.1.3.1.1.11)

The number of Offline Sequences received by this FxpPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of RxOfflineSeqs received by a port.

## fcFxPortOlsOuts (1.3.6.1.2.1.75.1.3.1.1.12)

The number of Offline Sequences issued by this FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of TxOfflineSeqs sent by a port.

---

## Accounting Groups

Each group consists of a table that contains accounting information for the FxPorts in the Fabric Element: Class 1 Accounting Group, Class 2 Accounting Group, and Class 3 Accounting Group.

### Class 1 Accounting Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `"snmpget localhost public fcFxPortC1InFrames.1.1"`. This table contains one entry for each FxPort in the Fabric Element and Counter32s for certain types of events that have occurred in the FxPorts since the management agent was re-initialized.

## fcFxPortC1InFrames (1.3.6.1.2.1.75.1.4.1.1.1)

The number of Class 1 frames (other than Class 1 connect-request) received by this FxPort from its attached NxPort.

### Syntax



Counter32

#### Access

read-only

#### Status

current

#### Return Value

Unsupported. Always returns error status “NoSuchName”.

### fcFxpPortC1OutFrames (1.3.6.1.2.1.75.1.4.1.1.2)

The number of Class 1 frames (other than Class 1 connect- request) delivered through this FxPort to its attached NxPort.

#### Syntax

Counter32

#### Access

read-only

#### Status

current

#### Return Value

Unsupported. Always returns error status “NoSuchName”.

### fcFxpPortC1InOctets (1.3.6.1.2.1.75.1.4.1.1.3)

The number of Class 1 frame octets, including the frame delimiters, received by this FxPort from its attached NxPort.

#### Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status “NoSuchName”.

## fcFxpPortC1OutOctets (1.3.6.1.2.1.75.1.4.1.1.4)

The number of Class 1 frame octets, including the frame delimiters, delivered through this FxPort its attached NxPort.

## Syntax

ounter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status “NoSuchName”.

## fcFxpPortC1Discards (1.3.6.1.2.1.75.1.4.1.1.5)

The number of Class 1 frames discarded by this FxPort.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortC1FbsyFrames (1.3.6.1.2.1.75.1.4.1.1.6)

The number of F\_BSY frames generated by this FxPort against Class 1 connect-request.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortC1FrjtFrames (1.3.6.1.2.1.75.1.4.1.1.7)

The number of F\_RJT frames generated by this FxPort against Class 1 connect-request.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortC1InConnections (1.3.6.1.2.1.75.1.4.1.1.8)

The number of Class 1 connections successfully established in which the attached NxPort is the source of the connect-request.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortC1OutConnections (1.3.6.1.2.1.75.1.4.1.1.9)

The number of Class 1 connections successfully established in which the attached NxPort is the destination of the connect-request.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

## fcFxPortC1ConnTime (1.3.6.1.2.1.75.1.4.1.1.10)

The cumulative time that this FxPort has been engaged in Class 1 connection. The amount of time is counted from after a connect-request has been accepted until the connection is disengaged, either by an EOFdt or Link Reset.

## Syntax

Milliseconds

## Access

read-only

## Status

current

## Return Value

Unsupported. Always returns error status "NoSuchName".

---

## Class 2 Accounting Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: "snmpget localhost public fcFxPortC2InFrames.1.1". This table contains one entry for each FxPort in the Fabric Element and Counter32s for certain types of events that have occurred in the FxPorts since the management agent was re-initialized.

## fcFxPortC2InFrames (1.3.6.1.2.1.75.1.4.2.1.1)

The number of Class 2 frames received by this FxPort from its attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `Class2FramesIn` received by a port.

## fcFxPortC2OutFrames (1.3.6.1.2.1.75.1.4.2.1.2)

The number of Class 2 frames delivered through this FxPort to its attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `Class2FramesOut` sent by a port.

## fcFxpPortC2InOctets (1.3.6.1.2.1.75.1.4.2.1.3)

The number of Class 2 frame octets, including the frame delimiters, received by this FxPort from its attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `Class2WordsIn` received by a port.

## fcFxpPortC2OutOctets (1.3.6.1.2.1.75.1.4.2.1.4)

The number of Class 2 frame octets, including the frame delimiters, delivered through this FxPort to its attached NxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `Class2WordsOut` sent by a port.

## fcFxPortC2Discards (1.3.6.1.2.1.75.1.4.2.1.5)

The number of Class 2 frames discarded by this FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `Class2Toss` discarded by a port.

## fcFxPortC2FbsyFrames (1.3.6.1.2.1.75.1.4.2.1.6)

The number of `F_BSY` frames generated by this FxPort.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of `FBusy` frames generated by this port for Class 2 and 3 frames.



## fcFxpPortC2FrjtFrames (1.3.6.1.2.1.75.1.4.2.1.7)

The number of F\_RJT frames generated by this FxpPort against Class 2 frames.

### Syntax

Counter32

### Access

read-only

### Status

current

### Return Value

The total number of FReject frames generated by this port for Class 2 and 3 frames.

---

## Class 3 Accounting Table

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: "snmpget localhost public fcFxpPortC3InFrames.1.1". This table contains one entry for each FxpPort in the Fabric Element and Counter32s for certain types of events that have occurred in the FxpPorts since the management agent has re-initialized.

## fcFxpPortC3InFrames (1.3.6.1.2.1.75.1.4.3.1.1)

The number of Class 3 frames received by this FxpPort from its attached NxPort.

### Syntax

Counter32

### Access

read-only

## Status

current

## Return Value

The total number of `Class3FramesIn` received by a port.

## fcFxpPortC3OutFrames (1.3.6.1.2.1.75.1.4.3.1.2)

The number of Class 3 frames delivered through this FxPort to its attached NxPort.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

The total number of `Class3FramesOut` sent by a port.

## fcFxpPortC3InOctets (1.3.6.1.2.1.75.1.4.3.1.3)

The number of Class 3 frame octets, including the frame delimiters, received by this FxPort from its attached NxPort.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

The total number of `Class3WordsOut` received by a port.

## fcFxpPortC3OutOctets (1.3.6.1.2.1.75.1.4.3.1.4)

The number of Class 3 frame octets, including the frame delimiters, delivered through this FxPort to its attached NxPort.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

The total number of `Class3WordsOut` sent by a port.

## fcFxpPortC3Discards (1.3.6.1.2.1.75.1.4.3.1.5)

The number of Class 3 frames discarded by this FxPort.

## Syntax

Counter32

## Access

read-only

## Status

current

## Return Value

The total number of `Class3Toss` discarded by a port.

---

# Capability Group

The objects described in this section are in table format indexed by switch index and port index. An example of how to access one of these objects is: `snmpget localhost public fcFxPortName.1.1`. The Capability Group consists of a table describing information about what each FxPort is inherently capable of operating or supporting. A capability may be used as expressed in its respective object value in the Configuration group.

## fcFxPortCapFcphVersionHigh (1.3.6.1.2.1.75.1.5.1.1.1)

The highest or most recent version of FC-PH that the FxPort is capable of supporting.

### Syntax

FcphVersion

### Access

read-only

### Status

current

## Return Value

Always returns 32 (0x20).

## fcFxPortCapFcphVersionLow (1.3.6.1.2.1.75.1.5.1.1.2)

The lowest or earliest version of FC-PH that the FxPort is capable of supporting.

### Syntax

FcphVersion

### Access

read-only

### Status

current

### Return Value

Always returns 9.

## fcFxPortCapBbCreditMax (1.3.6.1.2.1.75.1.5.1.1.3)

The maximum number of receive buffers available for holding Class 1 connect-request, Class 2, or Class 3 frames from the attached NxPort.

### Syntax

FcBbCredit

### Access

read-only

### Status

current

### Return Value

The default is: 255 (0xFF).

## fcFxpPortCapBbCreditMin (1.3.6.1.2.1.75.1.5.1.1.4)

The minimum number of receive buffers available for holding Class 1 connect-request, Class 2, or Class 3 frames from the attached NxPort.

### Syntax

`FcBbCredit`

### Access

read-only

### Status

current

### Return Value

The default is: 0 (0x00).

## fcFxpPortCapRxDataFieldSizeMax (1.3.6.1.2.1.75.1.5.1.1.5)

The maximum size in bytes of the Data Field in a frame that the FxpPort is capable of receiving from its attached NxPort.

### Syntax

`FcRxDataFieldSize`

### Access

read-only

### Status

current

### Return Value

2112 (0x840).

## fcFxpPortCapRxDataFieldSizeMin (1.3.6.1.2.1.75.1.5.1.1.6)

The minimum size in bytes of the Data Field in a frame that the FxPort is capable of receiving from its attached NxPort.

### Syntax

FcRxDataFieldSize

### Access

read-only

### Status

current

### Return Value

128 (0x80).

## fcFxpPortCapCos (1.3.6.1.2.1.75.1.5.1.1.7)

A value indicating the set of classes of service that the FxPort is capable of supporting.

### Syntax

FcCosCap

### Access

read-only

### Status

current

### Return Value

Always returns Class F, 2, and 3 (0x0d).

## fcFxpPortCapIntermix (1.3.6.1.2.1.75.1.5.1.1.8)

A flag indicating whether or not the FxpPort is capable of supporting the intermixing of Class 2 and Class 3 frames during a Class 1 connection. This flag is only valid if the port is capable of supporting Class 1 service.

### Syntax

TruthValue

### Access

read-only

### Status

current

### Return Value

Always returns `False` (2).

## fcFxpPortCapStackedConnMode (1.3.6.1.2.1.75.1.5.1.1.9)

A value indicating the mode of Stacked Connect request that the FxpPort is capable of supporting.

### Syntax

FcStackedConnMode

### Access

read-only

### Status

current

### Return Value

Always returns `None` (1).



## fcFxpPortCapClass2SeqDeliv (1.3.6.1.2.1.75.1.5.1.1.10)

A flag indicating whether or not the FxpPort is capable of supporting Class 2 Sequential Delivery.

### Syntax

TruthValue

### Access

read-only

### Status

current

### Return Value

Always returns true (1).

## fcFxpPortCapClass3SeqDeliv (1.3.6.1.2.1.75.1.5.1.1.11)

A flag indicating whether or not the FxpPort is capable of supporting Class 3 Sequential Delivery.

### Syntax

TruthValue

### Access

read-only

### Status

current

### Return Value

Always returns `true` (1).

## fcFxpPortCapHoldTimeMaxv (1.3.6.1.2.1.75.1.5.1.1.12)

The maximum holding time that the FxPort is capable of supporting, in microseconds.

### Syntax

MicroSeconds

### Access

read-only

### Status

current

### Return Value

20000 (0x4E20)

## fcFxpPortCapHoldTimeMin (1.3.6.1.2.1.75.1.5.1.1.13)

The minimum holding time that the FxPort is capable of supporting, in microseconds.

### Syntax

MicroSeconds

### Access

read-only

### Status

current

## Return Value

10 (0x0A)



## Custom MIB Objects

---

This section covers the implementation details for the custom Management Information Bases on the Sun Storage Fibre Channel Switch 5802.

---

### Custom MIB Definitions

This MIB replaces the `fcFxPortPhysTable` module defined in `FIBRE-CHANNEL-FE-MIB`, and defines volatile control objects for ports in a switch. If the switch gets reset, these values revert back to the default values in the configuration file.

#### `fcQxPortPhysAdminStatus` (1.3.6.1.4.1.1663.1.3.10.1.1.3)

The desired state of the `FxPort`. A management station may place the `FxPort` in a desired state by setting this object accordingly. The `testing(3)` state indicates that no operational frames can be passed. When a Fabric Element initializes, all `FxPorts` start with `fcQxPortPhysAdminStatus` in the `offline(2)` state. As the result of either explicit management action or per configuration information accessible by the Fabric Element, `fcQxPortPhysAdminStatus` is then changed to either the `online(1)` or `testing(3)` states, or remains in the `offline` state.

#### Syntax

INTEGER {

`online(1)`, - place port online

`offline(2)`, - take port offline

```
testing(3) - initiate test procedures
}
```

Access

read-write

Status

current

Return Value

Refer to [TABLE 6-1](#) for fcQxPortPhysAdminStatus read values.

**TABLE 6-1** fcQxPortPhysAdminStatus Read Return Values

Port	Value
Online	online (1)
Offline	offline (2)
Diagnostics	testing (3)
Down	offline (2)

Refer to [TABLE 6-2](#) for fcQxPortPhysAdminStatus write values.

**TABLE 6-2** fcQxPortPhysAdminStatus Write Values

Port	Value
Online	online (1)
Offline	offline (2)
Diagnostics	testing (3)

fcQxPortPhysOperStatus  
(1.3.6.1.4.1.1663.1.3.10.1.1.4)

The current operational status of the FxPort. The testing(3) indicates that no operational frames can be passed. If fcQxPortPhysAdminStatus is offline(2), then fcQxPortPhysOperStatus should be offline(2). If fcQxPortPhysAdminStatus is changed to online(1), then fcQxPortPhysOperStatus should change to

online(1). If the FxPort is ready to accept Fabric Login request from the attached NxPort, it should proceed and remain in the link- failure(4) state if, and only if, there is a fault that prevents it from going to the online(1) state.

Syntax

```
INTEGER {  
  
    online(1), - Login may proceed  
  
    offline(2), - Login cannot proceed  
  
    testing(3), - port is under test  
  
    linkFailure(4) - failure after online/testing  
  
}
```

Access

read-only

Status

current

Return Value

Refer to [TABLE 6-3](#) for fcQxPortPhysOperStatus return values.

**TABLE 6-3** fcFxPortPHysOperStatus Return Values

Status	Return Value
Online	online (1)
Offline	offline (2)
Diagnostic	testing (3)
Down	linkfailure (4)



# Related Traps

The following traps contain the trap information being sent from the agent to the manager.

## qlSB2PortLinkDown (qLogicExperimental 0 10)

A linkDown trap signifies that the SNMP entity, acting in an agent role, has detected that the `fcQxPortPhysOperStatus` object for one of its communication links has left the online state and transitioned to some other state. The current state is indicated by the included value of `fcQxPortPhysOperStatus`.

Variables:

*{fcQxPortPhysAdminStatus, fcQxPortPhysOperStatus}*

## qlSB2PortLinkUp (qLogicExperimental 0 11)

A linkUp trap signifies that the SNMP entity, acting in an agent role, has detected that the `fcQxPortPhysOperStatus` object for one of its communication links has entered the online state from some other state. The current state is indicated by the included value of `fcQxPortPhysOperStatus`.

Variables:

*{fcQxPortPhysAdminStatus, fcQxPortPhysOperStatus}*

## qlconnUnitAddedTrap (qLogicExperimental 0 12)

A connUnit has been added to this agent.

Variables:

*{connUnitId}*



## Firmware Download MIB Objects

---

This section covers the implementation details for the Firmware Download Management Information Bases (FD-MIB) on the switch.

---

### Firmware Download MIB Definitions

The FD-MIB enables you to download, install, and activate new firmware using the Trivial File Transfer Protocol (TFTP). The downloaded firmware can be activated using a hot reset (non-disruptive) or a hard reset (disruptive).

A hot reset is a Non-Disruptive Code Load Activation (NDCLA) operation. The firmware will be activated, the switch will be reset without a Power On Self Test, and switch traffic will not be disrupted. The switch does not need to be rebooted after the firmware is activated. During a hotreset operation, fabric services will be unavailable for a short period (30-75 seconds depending on switch model). To ensure that the NDCLA operation is successful, verify that all administrative changes to the fabric (if any) are complete. When you need to do NDCLA/hotreset to multiple switches, only perform the NDCLA/hotreset on one switch at a time, and wait 75 seconds before performing the NDCLA/hotreset operation on the next switch.

A hard reset is a Normal (or regular) reset operation. The firmware will be activated, the switch will be reset with a Power On Self Test, and switch traffic will be disrupted. The switch must be rebooted after the firmware is activated.

#### qlgcChFwOpResult (1.3.6.1.4.1.3873.3.1.1.2.1)

The status of the last firmware download and/or installation attempt.

Syntax

## OBJECT IDENTIFIER

### Access

read-only

### Status

Current

### Return Value

Returns the following:

DownloadNoError: 1.3.6.1.4.1.3873.3.1.1.1.1.1,  
DownloadHostError: 1.3.6.1.4.1.3873.3.1.1.1.1.2,  
DownloadFileError: 1.3.6.1.4.1.3873.3.1.1.1.1.3,  
InstallNoError: 1.3.6.1.4.1.3873.3.1.1.1.2.1,  
InstallFileError: 1.3.6.1.4.1.3873.3.1.1.1.2.2,  
InstallFileErrorNoAdmin: 1.3.6.1.4.1.3873.3.1.1.1.2.3,  
ResetNoError: 1.3.6.1.4.1.3873.3.1.1.1.3.1,  
ResetNoErr: 1.3.6.1.4.1.3873.3.1.1.1.3.2,  
ResetNoAdmin: 1.3.6.1.4.1.3873.3.1.1.1.3.3

## qlgcChFwOpRequest (1.3.6.1.4.1.3873.3.1.1.2.2)

Starts the operation to download/install firmware, and/or reset the switch.

### Syntax

```
INTEGER {  
  (1) - auto (downloads, installs, and resets the switch)  
  (2) - downloadOnly  
  (3) - installOnly  
  (4) - resetOnly  
}
```

## Access

read-write

## Status

current

# qlgcChFwDwldHostAddrType (1.3.6.1.4.1.3873.3.1.1.2.3)

The type of the IP address from which the firmware file is accessed.

## Syntax

INTEGER

## Access

read-write

## Status

current

# qlgcChFwDwldHostAddr (1.3.6.1.4.1.3873.3.1.1.2.4)

The IP address from which the firmware file is accessed.

## Syntax

IP ADDRESS

## Access

read-write

## Status

current

## qlgcChFwDwldHostPort (1.3.6.1.4.1.3873.3.1.1.2.5)

The port number (defaults is 69) used to transfer the firmware file.

### Syntax

INTEGER

### Access

read-write

### Status

current

## qlgcChFwDwldPathName (1.3.6.1.4.1.3873.3.1.1.2.6)

The full directory name on the server where the firmware file is located. If the firmware file to be downloaded is in a subdirectory, setting this path name to the name of that subdirectory is required.

### Syntax

DisplayString

### Access

read-write

### Status

current

## qlgcChFwDwldFileName (1.3.6.1.4.1.3873.3.1.1.2.7)

The filename of the firmware being transferred.

## Syntax

DisplayString

## Access

read-write

## Status

current

# qlgcChFwResetMethod (1.3.6.1.4.1.3873.3.1.1.2.8)

The value for the type of reset (hot reset or hard reset).

## Syntax

INTEGER {

(1) - Normal (disruptive)

(2) - NDCLA (Non-Disruptive Code Load Activation)

## Access

read-write

## Status

current



# Glossary

---

<b>AL_PA</b>	Arbitrated Loop Physical Address
<b>Arbitrated Loop</b>	A Fibre Channel topology where ports use arbitration to establish a point-to-point circuit.
<b>Arbitrated Loop Physical Address (AL_PA)</b>	A unique one-byte value assigned during loop initialization to each NL_Port on a Loop.
<b>Abstract Syntax Notation (ASN.1)</b>	Abstract Syntax Notation number One (ASN.1) is an international standard that specifies data used in communication protocols.
<b>Authentication Trap</b>	Enables or disables the reporting of SNMP authentication failures. If enabled, a notification trap is sent to the configured trap addresses in the event of an authentication failure. The default value is False.
<b>BER</b>	Bit Error Rate
<b>Bit Error Rate</b>	The probability that a transmitted bit will be erroneously received. The BER is measured by counting the number of bits in error at the output of a receiver and dividing by the total number of bits in the transmission. BER is typically expressed as a negative power of 10.
<b>Buffer Credit</b>	A measure of port buffer capacity equal to one frame.
<b>Class 2 Service</b>	A service that multiplexes frames at frame boundaries to or from one or more N_Ports with acknowledgment provided.
<b>Class 3 Service</b>	A service that multiplexes frames at frame boundaries to or from one or more N_Ports without acknowledgment.
<b>Contact</b>	Specifies the name of the contact person who is to be contacted to respond to trap events. The default is undefined.
<b>Datagram</b>	A message sent between two communicating entities for which no explicit link level acknowledgement is expected.

<b>Domain ID</b>	User defined name that identifies the switch in the fabric.
<b>Fabric Management Switch</b>	The switch through which the fabric is managed.
<b>Flash Memory</b>	Memory on the switch that contains the chassis control firmware.
<b>Frame</b>	Data unit consisting of a start-of-frame (SOF) delimiter, header, data payload, CRC, and an end-of-frame (EOF) delimiter.
<b>ICMP</b>	Internet Control Message Protocol
<b>IETF</b>	Internet Engineering Task Force
<b>Initiator</b>	The device that initiates a data exchange with a target device.
<b>Internet Engineering Task Force</b>	A large open international community of network designers, operators, vendors, and researchers concerned with evolution and smooth operation of the Internet, and responsible for producing RFCs. The standards body responsible for Internet standards, including SNMP, TCP/IP and policy for QoS.
<b>Internet Control Message Protocol</b>	A control protocol strongly related to IP and TCP, and used to convey a variety of control and error indications.
<b>InteropCredit</b>	Port configuration parameter that adjusts the number of port buffer credits to allow interoperability with some switches that are not FC-SW-2 compliant.
<b>IP</b>	Internet Protocol
<b>ISLSecurity</b>	ISLSecurity determines which switches a port will establish a link with. ANY - we will link with any switch. Ours - we will only link to another switches that is FC-SW-2 compliant. None - the port will not establish an ISL link.
<b>LCFEnable</b>	LCFEnable gives preference to Link control frames (such as Class 2 ACK frames) over other frames, when queued for transmission in the switch. This may provide better performance when running Class 2 traffic. LCFEnable is incompatible with MFSEnable, and both cannot be selected. (True / False)
<b>LIP</b>	Loop Initialization Primitive sequence
<b>Location</b>	Specifies the switch location. The default is undefined.
<b>Logged-In LED</b>	A port LED that indicates device login or loop initialization status.
<b>Management Information Base</b>	A set of guidelines and definitions for the Fibre Channel functions. The specification and formal description of a set of objects and variables that can be read and possibly written using the SNMP protocol. Various standard MIBs are defined by the Internet Engineering Task Force.



<b>Management Workstation</b>	Workstation that manages the fabric through the fabric management switch.
<b>MIB</b>	Management Information Base
<b>MSEnable</b>	Determines whether GS-3 management server commands will be accepted on the port. It can be used to prevent in-band management of the switch on any or all ports. (True / False)
<b>NL_Port</b>	Node Loop Port. A Fibre Channel device port that supports arbitrated loop protocol.
<b>N_Port</b>	Node Port. A Fibre Channel device port in a point-to-point or fabric connection.
<b>NMS</b>	Network Management Station
<b>Network Management Station</b>	The console through which an administrator performs management functions.
<b>NoClose</b>	Causes the switch to keep the loop open, if no other device is arbitrating. It is intended to improve performance when there is a single L_Port device connected to the switch. (True / False).
<b>Node</b>	An addressable entity connected to an I/O bus or network. Used primarily to refer to computers, storage devices, and storage subsystems. The component of a node that connects to the bus or network is a port.
<b>Object</b>	In the context of access control, an entity to which access is controlled and/or usage of which is restricted to authorized subjects.
<b>QoS</b>	Quality of Service
<b>POST</b>	Power On Self Test
<b>Power On Self Test (POST)</b>	Diagnostics that the switch chassis performs at start up.
<b>Private Device</b>	A device that can communicate only with other devices on the same loop.
<b>Private Loop</b>	A loop of private devices connected to a single switch port.
<b>Read Community</b>	Read Community Authentication. A write-only field; the value on the switch and the SNMP management server must be the same. The default value is Public.
<b>Request For Comment (RFC)</b>	Internet-related specifications, including standards, experimental definitions, informational documents and best practice definitions, produced by the IETF.
<b>Enterprise Fabric Suite 2007</b>	Switch management application.
<b>SFF</b>	Small Form-Factor transceiver.

<b>SFP</b>	Small Form-Factor Pluggable. A transceiver device, smaller than a GigaBit Interface Converter, that plugs into the Fibre Channel port.
<b>Simple Network Management Protocol</b>	The protocol governing network management and that allows monitoring of network devices.
<b>SMI</b>	Structure of Management Information
<b>Small Form Factor</b>	A transceiver device, smaller than a GigaBit Interface Converter, that is permanently attached to the circuit board.
<b>Small Form-Factor Pluggable</b>	A transceiver device, smaller than a GigaBit Interface Converter, that plugs into the Fibre Channel port.
<b>SNMP</b>	Simple Network Management Protocol
<b>Structure of Management Information</b>	A notation for setting or retrieving management variables over SNMP.
<b>Target</b>	A storage device that responds to an initiator device.
<b>TCP</b>	Transmission Control Protocol
<b>Trap Address</b>	Specifies the IP address to which SNMP traps are sent. The default is 127.0.0.1. A maximum of 5 trap addresses are supported.
<b>Trap Community</b>	Trap Community Authentication. A write-only field; the value on the switch and the SNMP management server must be the same. The default value is Public.
<b>Trap Port</b>	The port number on which the trap is set.
<b>Trap Severity</b>	Specifies a severity level to assign to the trap. Trap severity levels include Unknown, Emergency, Alert, Critical, Error, Warning, Notify, Info, Debug, and Mark
<b>UDP</b>	User Datagram Protocol
<b>User Datagram Protocol</b>	An Internet protocol that provides connection-less datagram delivery service to applications. Abbreviated UDP. UDP over Internet Protocol adds the ability to address multiple endpoints within a single network node to IP.
<b>VIEnable</b>	FC-VI. When enabled, VI preference frames will be transmitted ahead of other frames. (True / False)
<b>Worldwide Name (WWN)</b>	A unique 64-bit address assigned to a device by the device manufacturer.

<b>Write Community</b>	Write Community Authentication. A write-only field; the value on the switch and the SNMP management server must be the same. The default value is Private.
<b>WWN</b>	Worldwide Name
<b>Zone</b>	A set of ports or devices grouped together to control the exchange of information.
<b>Zone Set</b>	A set of zones grouped together. The active zone set defines the zoning for a fabric.



# Index

---

## A

- Accounting Groups, 254
- Additional IP Objects, 49
- Additional TCP Objects, 68
- Address Translation Group, 26
- Agent, 2
- Alert, 3, 212
- atIfIndex, 27
- atNetAddress, 28
- atPhysAddress, 27
- atTable, 27

## B

- book
  - organization, vii

## C

- Capability Group, 266
- Class 1 Accounting Table, 254
- Class 2 Accounting Table, 259
- Class 3 Accounting Table, 263
- Configuration Group, 222
- configurationChangeTime, 97
- Configuring switch, 9
- Connectivity Table, 98
- Connectivity Unit Group, 95
- connUnitConfigurationChangeTime, 107
- connUnitContact, 112
- connUnitControl, 110
- connUnitDomainId, 104

- connUnitEventCurrID, 115
- connUnitEventDescr, 149
- connUnitEventFilter, 113
- connUnitEventId, 146
- connUnitEventIndex, 145
- connUnitEventObject, 149
- connUnitEventSeverity, 147
- connUnitEventType, 148
- connUnitEventUnitId, 144
- connUnitGlobalId, 99
- connUnitId, 98
- connUnitInfo, 110
- connUnitLinkAgentAddressTypeY, 155
- connUnitLinkAgentAddressY, 154
- connUnitLinkAgentPortY, 156
- connUnitLinkConnIdY, 157
- connUnitLinkCurrIndex, 157
- connUnitLinkIndex, 151
- connUnitLinkNodeIdX, 151
- connUnitLinkNodeIdY, 153
- connUnitLinkPortNumber, 153
- connUnitLinkPortNumberX, 152
- connUnitLinkPortWwnX, 152
- connUnitLinkPortWwnY, 154
- connUnitLinkUnitId, 150
- connUnitLinkUnitTypeY, 156
- connUnitLocation, 112
- connUnitMaxEvents, 114
- connUnitModuleId, 109
- connUnitName, 109

connUnitNumEvents, 114  
 connUnitNumports, 100  
 connUnitNumRevs, 108  
 connUnitNumSensors, 106  
 connUnitNumZones, 108  
 connUnitPortControl, 137  
 connUnitPortFCClassCap, 129  
 connUnitPortFCClassOp, 130  
 connUnitPortFCId, 135  
 connUnitPortHWState, 143  
 connUnitPortIndex, 127  
 connUnitPortModuleType, 133  
 connUnitPortName, 139  
 connUnitPortNodeWwn, 142  
 connUnitPortPhysicalNumber, 140  
 connUnitPortProtocolCap, 141  
 connUnitPortProtocolOp, 142  
 connUnitPortRevision, 136  
 connUnitPortSn, 135  
 connUnitPortSpeed, 137  
 connUnitPortStatCountAddressErrors, 195  
 connUnitPortStatCountBBCreditZero, 173  
 connUnitPortStatCountClass1FBSYFrames, 177  
 connUnitPortStatCountClass1FRJTFrames, 178  
 connUnitPortStatCountClass1PBSYFrames, 178  
 connUnitPortStatCountClass1PRJTFrames, 179  
 connUnitPortStatCountClass1RxFrames, 176  
 connUnitPortStatCountClass1TxFrames, 177  
 connUnitPortStatCountClass2FBSYFrames, 180  
 connUnitPortStatCountClass2FRJTFrames, 182  
 connUnitPortStatCountClass2PBSYFrames, 181  
 connUnitPortStatCountClass2PRJTFrames, 182  
 connUnitPortStatCountClass2RxFrames, 179  
 connUnitPortStatCountClass2TxFrames, 180  
 connUnitPortStatCountClass3Discards, 184  
 connUnitPortStatCountClass3RxFrames, 183  
 connUnitPortStatCountClass3TxFrames, 183  
 connUnitPortStatCountDelimiterErrors, 195  
 connUnitPortStatCountEncodingDisparityErrors, 196  
 connUnitPortStatCountError, 170  
 connUnitPortStatCountFBSYFrames, 174  
 connUnitPortStatCountFramesTooLong, 193  
 connUnitPortStatCountFramesTruncated, 194  
 connUnitPortStatCountFRJTFrames, 175  
 connUnitPortStatCountInputBuffersFull, 173  
 connUnitPortStatCountInvalidCRC, 190  
 connUnitPortStatCountInvalidOrderedSets, 193  
 connUnitPortStatCountInvalidTxWords, 191  
 connUnitPortStatCountLinkFailures, 190  
 connUnitPortStatCountLossofSignal, 192  
 connUnitPortStatCountLossofSynchronization, 192  
 connUnitPortStatCountNumberLinkResets, 187  
 connUnitPortStatCountNumberOfflineSequences, 189  
 connUnitPortStatCountPBSYFrames, 174  
 connUnitPortStatCountPrimitiveSequenceProtocolErrors, 191  
 connUnitPortStatCountPRJTFrames, 175  
 connUnitPortStatCountRxBroadcastObjects, 185  
 connUnitPortStatCountRxElements, 172  
 connUnitPortStatCountRxLinkResets, 186  
 connUnitPortStatCountRxMulticastObjects, 184  
 connUnitPortStatCountRxObjects, 171  
 connUnitPortStatCountRxOfflineSequences, 188  
 connUnitPortStatCountTxBroadcastObjects, 186  
 connUnitPortStatCountTxElements, 171  
 connUnitPortStatCountTxLinkResets, 187  
 connUnitPortStatCountTxMulticastObjects, 185  
 connUnitPortStatCountTxObjects, 170  
 connUnitPortStatCountTxOfflineSequences, 189  
 connUnitPortState1, 130  
 connUnitPortStatIndex, 169  
 connUnitPortStatObject, 140  
 connUnitPortStatUnitId, 169  
 connUnitPortStatus, 131  
 connUnitPortTransmitterType, 132  
 connUnitPortType, 127  
 connUnitPortUnitId, 126  
 connUnitPortWwn, 134  
 connUnitPrincipal, 105  
 connUnitProduct, 102  
 connUnitProxyMaster, 105  
 connUnitREventTime, 146  
 connUnitRevsDescription, 119  
 connUnitRevsIndex, 118

- connUnitRevsRevId, 118
- connUnitRevsUnitId, 117
- connUnitSensorCharacteristic, 125
- connUnitSensorIndex, 120
- connUnitSensorInfo, 123
- connUnitSensorMessage, 124
- connUnitSensorName, 121
- connUnitSensorStatus, 122
- connUnitSensorType, 124
- connUnitSensorUnitId, 120
- connUnitSEventTime, 147
- connUnitSn, 103
- connUnitSnsClassOfSvc, 200
- connUnitSnsFabricPortName, 203
- connUnitSnsFC4Type, 201
- connUnitSnsHardAddress, 203
- connUnitSnsId, 197
- connUnitSnsNodeIPAddress, 200
- connUnitSnsNodeName, 199
- connUnitSnsPortIdentifier, 198
- connUnitSnsPortIndex, 198
- connUnitSnsPortIPAddress, 202
- connUnitSnsPortName, 199
- connUnitSnsPortType, 202
- connUnitSnsProcAssoc, 201
- connUnitSnsSymbolicNodeName, 204
- connUnitSnsSymbolicPortName, 204
- connUnitState, 100
- connUnitStatus, 101
- connUnitStatusChangeTime, 107
- connUnitTableChangeTime, 98
- connUnitType, 99
- connUnitUpTime, 103
- connUnitUrl, 104
- Critical, 3, 212

## D

- Debug, 3, 212

## E

- EGP Group, 71
- EGP Neighbor Table, 73
- egpAs, 80

- egpInErrors, 72
- egpInMsgs, 72
- egpNeighAddr, 74
- egpNeighAs, 74
- egpNeighEventTrigger, 79
- egpNeighInErrMsgs, 76
- egpNeighInErrs, 75
- egpNeighInMsgs, 75
- egpNeighIntervalHello, 78
- egpNeighIntervalPoll, 78
- egpNeighMode, 79
- egpNeighOutErrMsgs, 77
- egpNeighOutErrs, 76
- egpNeighOutMsgs, 75
- egpNeighState, 73
- egpNeighStateDowns, 77
- egpNeighStateUps, 77
- egpNeighTable, 73
- egpOutErrors, 73
- egpOutMsgs, 72
- Emergency, 3, 212
- Enterprise, 9
- Enterprise Fabric Manager 2007, 9
- Error, 3, 212
- Error Group, 248
- Event Table, 144

## F

- fcFeElementName, 223
- fcFeFabricName, 223
- fcFeModuleCapacity, 224
- fcFeModuleDescr, 224
- fcFeModuleFxpPortCapacity, 227
- fcFeModuleLastChange, 226
- fcFeModuleName, 228
- fcFeModuleObjectID, 225
- fcFeModuleOperStatus, 225
- fcFeModuleTable, 224
- fcFxpPortAddressIdErrors, 252
- fcFxpPortAdminMode, 237
- fcFxpPortBbCredit, 230
- fcFxpPortBbCreditAvailable, 236
- fcFxpPortBbCreditModel, 247

fcFxpPortC1AccountingTable, 254  
 fcFxpPortC1ConnTime, 259  
 fcFxpPortC1Discards, 256  
 fcFxpPortC1FbsyFrames, 257  
 fcFxpPortC1FrjtFrames, 257  
 fcFxpPortC1InConnections, 258  
 fcFxpPortC1InFrames, 254  
 fcFxpPortC1InOctets, 255  
 fcFxpPortC1OutConnections, 258  
 fcFxpPortC1OutFrames, 255  
 fcFxpPortC1OutOctets, 256  
 fcFxpPortC2AccountingTable, 260  
 fcFxpPortC2Discards, 262  
 fcFxpPortC2FbsyFrames, 262  
 fcFxpPortC2FrjtFrames, 263  
 fcFxpPortC2InFrames, 260  
 fcFxpPortC2InOctets, 261  
 fcFxpPortC2OutFrames, 260  
 fcFxpPortC2OutOctets, 261  
 fcFxpPortC3Discards, 265  
 fcFxpPortC3InFrames, 263  
 fcFxpPortC3InOctets, 264  
 fcFxpPortC3OutFrames, 264  
 fcFxpPortC3OutOctets, 265  
 fcFxpPortCapBbCreditMax, 267  
 fcFxpPortCapBbCreditMin, 268  
 fcFxpPortCapClass2SeqDeliv, 271  
 fcFxpPortCapClass3SeqDeliv, 271  
 fcFxpPortCapCos, 269  
 fcFxpPortCapFcphVersionHigh, 266  
 fcFxpPortCapFcphVersionLow, 267  
 fcFxpPortCapHoldTimeMax, 272  
 fcFxpPortCapHoldTimeMin, 272  
 fcFxpPortCapIntermix, 270  
 fcFxpPortCapRxDatFieldSizeMax, 268  
 fcFxpPortCapRxDatFieldSizeMin, 269  
 fcFxpPortCapStackedConnMode, 270  
 fcFxpPortCapTable, 266  
 fcFxpPortClass2SeqDeliv, 233  
 fcFxpPortClass2SeqDelivAgreed, 245  
 fcFxpPortClass3SeqDeliv, 234  
 fcFxpPortClass3SeqDelivAgreed, 246  
 fcFxpPortConnectedNxPort, 247  
 fcFxpPortCosSuppAgreed, 243  
 fcFxpPortCosSupported, 232  
 fcFxpPortDelimiterErrors, 251  
 fcFxpPortEdtov, 231  
 fcFxpPortFcphVersionAgreed, 241  
 fcFxpPortFcphVersionHigh, 229  
 fcFxpPortFcphVersionLow, 229  
 fcFxpPortHoldTime, 234  
 fcFxpPortID, 235  
 fcFxpPortIntermixSuppAgreed, 244  
 fcFxpPortIntermixSupported, 232  
 fcFxpPortInvalidCrcs, 251  
 fcFxpPortInvalidTxWords, 250  
 fcFxpPortLinkFailures, 248  
 fcFxpPortLinkReseatIns, 252  
 fcFxpPortLinkResetOuts, 253  
 fcFxpPortName, 228  
 fcFxpPortNxPortBbCredit, 242  
 fcFxpPortNxPortName, 246  
 fcFxpPortNxPortRxDatFieldSize, 243  
 fcFxpPortOlsIns, 253  
 fcFxpPortOlsOuts, 254  
 fcFxpPortOperMode, 236  
 fcFxpPortPhysAdminStatus, 238  
 fcFxpPortPhysEntry, 238  
 fcFxpPortPhysLastChange, 240  
 fcFxpPortPhysOperStatus, 239, 276  
 fcFxpPortPhysRttov, 241  
 fcFxpPortPhysTable, 238  
 fcFxpPortPrimSeqProtoErrors, 250  
 fcFxpPortRatov, 231  
 fcFxpPortRxBufSize, 230  
 fcFxpPortSigLosses, 249  
 fcFxpPortStackedConnMode, 233  
 fcFxpPortStackedConnModeAgreed, 245  
 fcFxpPortSyncLosses, 249  
 Fx Port Fabric Login Table, 241  
 FxPort Configuration Table, 228  
 FxPort Physical Level Table, 238

## G

Groups in MIB-II, 11



## I

- ICMP Group, 49
- icmpInAddrMaskReps, 54
- icmpInAddrMasks, 54
- icmpInDestUnreachs, 50
- icmpInEchoReps, 53
- icmpInEchos, 52
- icmpInErrors, 50
- icmpInMsgs, 49
- icmpInParmProbs, 51
- icmpInRedirects, 52
- icmpInSrcQuenchs, 51
- icmpInTimeExcds, 51
- icmpInTimestampReps, 53
- icmpInTimestamps, 53
- icmpOutAddrMaskReps, 59
- icmpOutAddrMasks, 59
- icmpOutDestUnreachs, 55
- icmpOutEchoReps, 58
- icmpOutEchos, 57
- icmpOutErrors, 55
- icmpOutMsgs, 55
- icmpOutParmProbs, 56
- icmpOutRedirects, 57
- icmpOutSrcQuenchs, 57
- icmpOutTimeExcds, 56
- icmpOutTimestampReps, 59
- icmpOutTimestamps, 58
- ifAdminStatus, 19
- ifDescr, 17
- ifIndex, 17
- ifInDiscards, 22
- ifInErrors, 22
- ifInNUcastPkts, 22
- ifInOctets, 21
- ifInUcastPkts, 21
- ifInUnknownProtos, 23
- ifLastChange, 20
- ifMtu, 18
- ifNumber, 16
- ifOperStatus, 20
- ifOutDiscards, 25
- ifOutErrors, 25
- ifOutNUcastPkts, 24
- ifOutOctets, 23
- ifOutQLen, 25
- ifOutUcastPkts, 24
- ifPhysAddress, 19
- ifSpecific, 26
- ifSpeed, 19
- ifType, 18
- Info, 3, 212
- Interfaces Group, 16
- Interfaces Table, 17
- IP Address Table, 37
- IP Address Translation Table, 46
- IP Group, 28
- IP Routing Table, 39
- ipAddrTable, 37
- ipAdEntAddr, 37
- ipAdEntBcastAddr, 38
- ipAdEntIfIndex, 37
- ipAdEntNetMask, 38
- ipAdEntReasmMaxSize, 39
- ipDefaultTTL, 29
- ipForwarding, 28
- ipForwDatagrams, 31
- ipFragCreates, 36
- ipFragFails, 36
- ipFragOKs, 36
- ipInAddrErrors, 30
- ipInDelivers, 32
- ipInDiscards, 32
- ipInHdrErrors, 30
- ipInReceives, 29
- ipInUnknownProtos, 31
- ipNetToMediaIfIndex, 46
- ipNetToMediaNetAddress, 47
- ipNetToMediaPhysAddress, 47
- ipNetToMediaType, 48
- ipOutDiscards, 33
- ipOutNoRoutes, 34
- ipOutRequests, 33
- ipReasmFails, 35
- ipReasmOKs, 35
- ipReasmReqds, 34

- ipReasmTimeout, 34
- ipRouteAge, 44
- ipRouteDest, 39
- ipRouteIfIndex, 40
- ipRouteInfo, 46
- ipRouteMask, 45
- ipRouteMetric1, 40
- ipRouteMetric2, 41
- ipRouteMetric3, 41
- ipRouteMetric4, 42
- ipRouteMetric5, 45
- ipRouteNextHop, 43
- ipRouteProto, 44
- ipRouteTable, 39
- ipRouteType, 43
- ipRoutingDiscards, 49

## L

- Link Table, 150

## M

- Management information bases, 11
- Manager, 2
- Mark, 212
- MIB Definitions, 221
- MIB-II, 11
- Module Table, 224

## N

- Notify, 3, 212

## O

- organization of book, vii

## P

- Port Statistics Table, 168
- Port Table, 126

## Q

- qlgcChFwDwldFileName, 282
- qlgcChFwDwldHostAddr, 281
- qlgcChFwDwldHostAddrType, 281
- qlgcChFwDwldHostPort, 282
- qlgcChFwDwldPathName, 282

- qlgcChFwOpRequest, 280
- qlgcChFwOpResult, 279
- qlgcChFwResetMethod, 283

## R

- Revision Table, 117
- revisionNumber, 94

## S

- Sensor Table, 120
- Simple Name Server Table, 196
- Simple Network Management Protocol, 1
- SNMP Group, 80
- snmpEnableAuthenTraps, 92
- snmpInASNParseErrs, 83
- snmpInBadCommunityNames, 82
- snmpInBadCommunityUses, 82
- snmpInBadValues, 84
- snmpInBadVersions, 82
- snmpInGenErrs, 85
- snmpInGetNexts, 87
- snmpInGetRequests, 86
- snmpInGetResponses, 88
- snmpInNoSuchNames, 84
- snmpInPkts, 81
- snmpInReadOnlys, 85
- snmpInSetRequests, 87
- snmpInTooBigs, 83
- snmpInTotalReqVars, 85
- snmpInTotalSetVars, 86
- snmpInTraps, 88
- snmpOutBadValues, 89
- snmpOutGenErrs, 90
- snmpOutGetNexts, 90
- snmpOutGetRequests, 90
- snmpOutGetResponses, 91
- snmpOutNoSuchNames, 89
- snmpOutPkts, 81
- snmpOutSetRequests, 91
- snmpOutTooBigs, 88
- snmpOutTraps, 92
- SNMPv1, 1
- SNMPv2c, 1

- Status Group, 235
- statusChangeTime, 97
- sysContact, 14
- sysDescr, 12
- sysLocation, 15
- sysName, 14
- sysObjectID, 13
- sysServices, 15
- System Group, 12
- systemURL, 96
- sysUpTime, 13

## **T**

- TCP Connection Table, 65
- TCP Group, 60
- tcpActiveOpens, 62
- tcpAttemptFails, 63
- tcpConnLocalAddress, 66
- tcpConnLocalPort, 66
- tcpConnRemAddress, 67
- tcpConnRemPort, 67
- tcpConnState, 65
- tcpCurrEstab, 63
- tcpEstabResets, 63
- tcpInErrs, 68
- tcpInSegs, 64
- tcpMaxConn, 61
- tcpOutRsts, 68
- tcpOutSegs, 64
- tcpPassiveOpens, 62
- tcpRetransSegs, 65
- tcpRtoAlgorithm, 60
- tcpRtoMax, 61
- tcpRtoMin, 60
- Transmission Group, 80
- trap severity levels, 3, 212
- Trap Table, 211
- trapRegFilter, 214
- trapRegIpAddress, 213
- trapRegPort, 214
- trapRegRowState, 215

## **U**

- UDP Group, 69
- UDP Listener Table, 70
- udpInDatagrams, 69
- udpInErrors, 70
- udpLocalAddress, 71
- udpLocalPort, 71
- udpNoPorts, 69
- udpOutDatagrams, 70
- Unknown, 3, 212
- uNumber, 95

## **W**

- Warning, 3, 212

