SmartSwitch ATM Switch User Guide

35 Industrial Way
Rochester, NH 03866
USA
(603) 332-9400

Part Number 04-0053-01 Rev. A
Order Number 9033002

NOTICE

Cabletron Systems reserves the right to make changes in specifications and other information contained in this document without prior notice. The reader should in all cases consult Cabletron Systems to determine whether any such changes have been made. The hardware, firmware, and software described in this manual are subject to change without notice.

IN NO EVENT SHALL CABLETRON SYSTEMS BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, BUT NOT LIMITED TO, LOST PROFITS) ARISING OUT OF OR RELATED TO THIS MANUAL OR THE INFORMATION CONTAINED IN IT, EVEN IF CABLETRON SYSTEMS HAS BEEN ADVISED OF, KNOWN, OR SHOULD HAVE KNOWN, THE POSSIBILITY OF SUCH DAMAGES.

Copyright 1998 - 99 by Cabletron Systems, Inc., P.O. Box 5005, Rochester, NH 03866-5005
All Rights Reserved

Printed in the United States of America

SmartSwitch ATM Switch User Guide

Part Number 04-0053-01 Rev. A

Order Number: 9033002

SmartSwitch, SPECTRUM, LANVIEW, MicroMMAC, and BRIM are registered trademarks and Element Manager, EPIM, EPIMA, EPIM-F1, EPIM-F2, EPIM-F3, EPIM-T, EPIM-X, FOT-F, FOT-F3, HubSTACK, SEH, SEHI, and TMS-3 are trademarks of Cabletron Systems, Inc. All other product names mentioned in this manual may be trademarks or registered trademarks of their respective companies.

FCC CLASS A NOTICE

This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Note This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment uses, generates, and can radiate radio frequency energy and if not installed in accordance with the appropriate Setup and Installation Guide, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause interference in which case the user will be required to correct the interference at his own expense.

Caution Changes or modifications made to this device which are not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

DOC CLASS A NOTICE

This digital apparatus does not exceed the Class A limits for radio noise emissions from digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications.

Le present appareil numerique n'emet pas de bruits radioelectriques depassant les limites applicables aux appareils numeriques de la class A prescrites dans le Reglement sur le brouillage radioelectrique edicte par le ministere des Communications du Canada.

DECLARATION OF CONFORMITY ADDENDUM

Application of Council Directive(s):	89/336/EEC 73/23/EEC
Manufacturer's Name:	Cabletron Systems, Inc.
Manufacturer's Address:	35 Industrial Way P. O. Box 5005 Rochester, NH 03866
Product Name:	SmartSwitch ATM switches
European Representative Name:	Mr. J. Solari
European Representative Address:	Cabletron Systems, Limited Nexus House, Newbury Business Park London Road, Newbury Berkshire RG13 2PZ, England
Conformance to Directive(s)/Product Standards:	EC Directive 89/336/EEC EC Directive 73/23/EEC EN 55022 EN 50082-1 EN 60950
Equipment Type/Environment:	Networking Equipment, for use in a Commercial or Light Industrial Environment.

We the undersigned, hereby declare, under our sole responsibility, that the equipment packaged with this notice conforms to the above directives.

Manufacturer:	Full Name:	**Mr. Ronald Fotino**
	Title:	**Principal Compliance Engineer**
	Location:	**Rochester, NH. U.S.A.**
Legal Repersentative in Europe:	Full Name:	**Mr. J. Solari**
	Title:	**Managing Director - E.M.E.A.**
	Location:	**Newbury, Berkshire, England**

SAFETY INFORMATION
CLASS 1 LASER TRANSCEIVERS

The connectors on I/O modules containing the part numbers IOM-29-4-MIX, IOM-29-4-IR, IOM-29-4-LR, IOM-39-1 and IOM-39-1-LR use Class 1 Laser transceivers. Read the following safety information before installing or operating one of these modules.

The Class 1 Laser transceivers use an optical feedback loop to maintain Class 1 operation limits. This control loop eliminates the need for maintenance checks or adjustments. The output is factory set, and does not allow any user adjustment. Class 1 Laser transceivers comply with the following safety standards:

- 21 CFR 1040.10 and 1040.11 U. S. Department of Health and Human Services (FDA).
- IEC Publication 825 (International Electrotechnical Commission).
- CENELEC EN 60825 (European Committee for Electrotechnical Standardization).

When operating within their performance limitations, laser transceiver output meets the Class 1 accessible emission limit of all three standards. Class 1 levels of laser radiation are not considered hazardous.

LASER RADIATION AND CONNECTORS

When the connector is in place, all laser radiation remains within the fiber. The maximum amount of radiant power exiting the fiber (under normal conditions) is -12.6dBm or $55x10^{-6}$ watts.

Removing the optical connector from the transceiver allows laser radiation to emit directly from the optical port. The maximum radiance from the optical port (under worst case conditions) is 0.8 W cm^{-2} or $8x10^{3}$ W m^{-2} sr-1.

Do not use optical instruments to view the laser output. The use of optical instruments to view laser output increases eye hazard. When viewing the output optical port, you must remove power from the network adapter.

FIBER OPTIC PROTECTIVE CAPS

Warning READ BEFORE REMOVING FIBER OPTIC PROTECTIVE CAPS.

Cable assemblies and MMF/SMF ports are shipped with protective caps to prevent contamination. To avoid contamination, replace port caps on all fiber optic devices when not in use.

Cable assemblies and MMF/SMF ports that become contaminated may experience signal loss or difficulty inserting and removing cable assemblies from MMF/SMF ports.

Contamination can be removed from cable assemblies by:

1. Blowing surfaces with canned duster (Chemtronics p/n ES1270 or equivalent).
2. Using a fiber port cleaning swab (Alcoa Fujikura LTS p/n ACT-01 or equivalent) saturated with optical-grade isopropyl alcohol, gently wipe the end surface of ferrules first; then wipe down the sides of both ferrules.
3. Blow ferrule surfaces dry with canned duster.

Contamination can be removed from MMF/SMF ports by:

1. Using the extension tube supplied with canned duster, blow into the optical port, being careful not to allow the extension tube to touch the bottom of the optical port.
2. Reconnect cable and check for proper mating. If problems remain, gently wipe out optical port with a DRY fiber port cleaning swab and repeat step 1.

Warning To avoid contamination, replace port caps on all fiber optic devices when not in use.

REGULATORY COMPLIANCE SUMMARY

SAFETY

SmartSwitch ATM switches meet the safety requirements of UL 1950, CSA C22.2 No. 950, EN 60950, IEC 950, and 73/23/EEC.

EMC

SmartSwitch ATM switches meet the EMC requirements of FCC Part 15, EN 55022, CSA C108.8, VCCI V-3/93.01, EN 50082-1, and 89/336/EEC.

REVISION HISTORY

Document Name: *SmartSwitch ATM Switch User Guide*
Document Part Number: 04-0053-01 Rev. A
Document Order Number: 9033002

Author: Bruce Jordan

Editor: Ayesha Maqsood

Illustrator: Mike Fornalski

Date	Revision	Description
March 1999	A	Initial release

TABLE OF CONTENTS

1 Introduction 1-1

1.1 Contents of the User Guide 1-1

1.2 SmartSwitch ATM Switch Differences 1-2

2 IP Over ATM and LANE 2-1

2.1 Creating an IP over ATM VLAN 2-1
2.1.1 Default ATM Addressing for IP over ATM 2-3

2.2 Creating an Emulated LAN 2-4
2.2.1 ATM Addressing for LAN Emulation 2-6
2.2.2 ELANs Across Multiple Switches 2-8
2.2.3 Switch Clients 2-9
2.2.4 Distributed LANE Services 2-9
2.2.5 ELAN Join Policies 2-11
2.2.6 LANE Over WAN Circuits 2-14
2.2.7 Using LNNI 2-16

3 PNNI Routing 3-1

3.1 PNNI Node Addressing 3-1
3.1.1 Default PNNI Addressing 3-1

3.2 Multi-level PNNI Topology 3-3
3.2.1 Connecting Multiple Peer Groups 3-3
3.2.2 Physical Connections Between Peer Groups 3-7

3.3 Managing Parallel PNNI Links 3-9
3.3.1 Aggregation Tokens 3-10
3.3.2 PNNI Link Timing 3-11

4 Routing 4-1

4.1 Additional Routing Protocols 4-1

4.2 IISP Routes 4-1
4.2.1 IISP Routing Considerations 4-2
4.2.2 IISP Link Timing 4-4

4.3 UNI Routes 4-5
4.3.1 UNI Link Timing 4-6

4.4 Route Metrics 4-7
4.4.1 Administrative Weights 4-7
4.4.2 Creating Route Metrics 4-7

4.5 IP Routing for Management 4-9

5 Virtual Ports and Static Connections 5-1

5.1 PVC Connections 5-1
5.1.1 Point-to-Point PVCs 5-1
5.1.2 Point-to-Multipoint PVCs 5-2
5.1.3 Connecting to Local Switch Client Through a PVC 5-4

5.2 PVP Connections 5-5
5.2.1 Connecting PVPs 5-7

5.3 Virtual Ports 5-7
5.3.1 Creating Virtual Ports 5-8

5.4 Soft PVC and PVP Connections 5-11
5.4.1 Soft PVC and Soft PVP differences 5-11
5.4.2 Making Soft PVC and PVP Connections 5-12
5.4.3 Creating a soft PVC 5-12
5.4.4 Creating a Soft PVP 5-15

6 Traffic Management 6-1

6.1 Traffic Management Capabilities 6-1
6.1.1 Traffic Descriptors 6-1
6.1.2 Call Admission Control Policy 6-3
6.1.3 Queue Buffers 6-5
6.1.4 EFCI, EPD, and RM Cell Marking 6-7

7 Firmware Upgrades and Bootline Commands 7-1

7.1 Update Firmware Commands 7-1

7.2 Bootline Commands 7-2
7.2.1 Accessing the Bootline Prompt 7-3
7.2.2 Bootline Commands Explanations 7-4
7.2.3 Upgrading Boot Load firmware 7-6
7.2.4 Upgrading POST Diagnostic firmware 7-7
7.2.5 Upgrading Switch Operating firmware 7-8

8 ATM Filtering and Clocking 8-1

8.1 Port ATM Address Filters 8-1
8.1.1 Creating ATM Address Filters 8-1
8.1.2 How ATM Address Filters Work 8-1
8.1.3 ATM Address Filter Example 8-2
8.1.4 Filter Considerations Regarding LANE and IP over ATM 8-3

8.2 Port Clock Configuration 8-3
8.2.1 Network Clocking 8-4

9 Troubleshooting 9-1

9.1 Troubleshooting IP over ATM 9-1

9.2 Troubleshooting LAN Emulation 9-2

9.3 Troubleshooting PNNI Links 9-3
9.3.1 Switches in Same Peer Group 9-3
9.3.2 Switches in Different Peer Groups 9-3

9.4 Troubleshooting Congestion 9-4
9.4.1 Diagnosing Congestion 9-4
9.4.2 Global Congestion 9-4
9.4.3 Port Congestion 9-5
9.5 Events and Alarms 9-6
9.5.1 Event Categories 9-6
9.5.2 Viewing Events and Alarms 9-7
9.5.3 Deleting Events and Alarms 9-8
9.6 Saving Core Dumps 9-9

A Agent Support A-1
A.1 MIB, SMI, MIB Files and Internet MIB Hierarchy A-1
A.1.1 CSI ZeitNet Proprietary MIBs A-2
A.1.2 Relation Between Object Identifier and the Represented Value A-3
A.1.3 Supported protocols A-4
A.1.4 Supported SMI Formats A-4
A.1.5 CSI ZeitNet Proprietary MIB Groups A-4
A.1.6 ATM SmartSwitch MIB Support A-6
A.1.7 MIB Exceptions A-6
A.2 Managing an ATM SmartSwitch A-7
A.2.1 Console Commands that Affect the Agent A-7
A.2.2 Default Community Strings A-8

B Technical Support B-1
B.1 Telephone Assistance B-1
B.2 FAX Service B-1
B.3 Electronic Services B-1
B.4 Placing A Support Call B-1
B.5 Hardware Warranty B-2
B.6 Software Warranty B-2
B.7 Repair Services B-2

Index I-1

LIST OF FIGURES

Figure 2-1 Single PVP connection between clients and LANE services 2-15
Figure 2-2 Multiple PVP connection between clients and LANE services 2-15
Figure 2-3 LNNI Redundant LECSs on same network 2-17
Figure 2-4 LNNI call set up load sharing 2-18
Figure 2-5 How LNNI handles ELAN join requests 2-19
Figure 3-1 Physical connectivity for multi-peer group example 3-4
Figure 3-2 Logical representation of connectivity between groups A and B 3-7
Figure 3-3 Adding a third PNNI node for next level connectivity 3-8
Figure 3-4 Aggregation token values and parallel links 3-11
Figure 4-1 IISP route across PNNI domain 4-3
Figure 4-2 Routes needed for a second IISP switch 4-4
Figure 4-3 IP routing through SW1 for connectivity to the Ethernet network 4-10
Figure 5-1 Terminating PVPs 5-7
Figure 5-2 Soft PVC across PNNI network 5-12
Figure 5-3 Soft PVC heals (is rerouted) to bypass broken link 5-12
Figure 7-1 Memory locations affected by the bootline commands 7-5
Figure A-1 Internet MIB hierarchy A-2
Figure A-2 CSI ZeitNet Private MIBs A-3
Figure A-3 Cabletron ATM SmartSwitch object identifier example A-4

LIST OF TABLES

Table 2-1 ELAN Join Policies 2-12
Table 6-1 Traffic descriptor type number explanation 6-2
Table 7-1 Bootline commands 7-4
Table 9-1 Settings for Class of Service Queues 9-4
Table A-1 CSI Zeitnet proprietary MIB groupings A-4

1 INTRODUCTION

Welcome to the SmartSwitch ATM User Guide. This manual provides instructions and information about switch use, maintenance, and problem solving for all SmartSwitch ATM switches. These include

- SmartSwitch 2500 Workgroup and Backbone ATM switches
- SmartSwitch 6A000 ATM switch modules
- SmartSwitch 9A100 ATM switch modules
- SmartSwitch 6500 ATM switch

Note For installation instructions and initial set up procedures for your particular SmartSwitch ATM switch, see the appropriate SmartSwitch ATM Switch Installation and Setup Guide.

1.1 CONTENTS OF THE USER GUIDE

The SmartSwitch ATM User Guide provides instructions and examples on using the SmartSwitch ATM switch features. By reading this manual you will learn how to perform the following operations:

- Creating and managing IP over ATM VLANs
- Creating and managing ELANS
- Using distributed LANE servers
- Configuring LNNI for LANE redundancy and load sharing through
- Creating and managing multi-level PNNI network topologies
- Adding routes (PNNI, IISP, UNI, and routes between ATM and Ethernet networks)
- Creating PVC and PVP connections
- Creating soft PVCs and soft PVPs
- Creating and using virtual ports
- Creating traffic descriptors
- Managing bandwidth, switch traffic, and congestion
- Upgrading switch firmware
- Configuring ATM address filters
- Configuring network clocking
- Troubleshooting VLANs, ELANs, PNNI topologies, and traffic congestion problems

Note For detailed descriptions of individual SmartSwitch ATM console commands, see the SmartSwitch ATM Reference Manual.

1.2 SMARTSWITCH ATM SWITCH DIFFERENCES

Not all features are supported on all SmartSwitch ATM switches. The SmartSwitch 6500 has capabilities that are not supported by the other SmartSwitch ATM switches. The following is a list of capabilities supported by the SmartSwitch 6500 only:

- PVPs
- Soft PVPs (all SmartSwitch ATM switches support soft PVCs)
- BUS logical multicasting
- Switch redundancy and automatic fail-over
- Network clocking

Note It is clearly stated within the text of this User Guide whether a particular feature is supported only by the SmartSwitch 6500.

2 IP OVER ATM AND LANE

This chapter describes working with the SmartSwitch ATM switch IP over ATM VLAN and emulated LAN capabilities. At the end of this chapter you will be able to use your SmartSwitch ATM switch to:

- Create an IP over ATM VLAN
- Create an emulated Ethernet LAN (LANE)

2.1 CREATING AN IP OVER ATM VLAN

This section describes implementing IP over ATM on your SmartSwitch ATM switch. The following assumptions are made:

- The SmartSwitch ATM switch will have a client on the IP over ATM VLAN
- The ARP server will reside on the switch and correspond to the address of the switch client
- All end nodes (computers, edge devices, and so on) support Switched Virtual Circuits (SVCs)

1. Log into the switch, either through the terminal port or through the Ethernet interface by telnet.
2. Create a client on the switch and assign it as the ARP server for the VLAN.

```
SmartSwitch # add ipatmclient
ClientNumber(0) : 1
ServerType(NONE) : local              — the ARP server is assigned to the switch client
ServerAddress()  :
IPAddress()  : 90.1.1.1               — IP address is for example only
NetMask(255.0.0.0) : 255.255.255.0    — subnet mask is for example only
MTU(9180)  :
SmartSwitch #
```

The example above creates a client on the switch, designates the client as the ARP server for the VLAN (**`ServerType`** = **`local`**), and assigns the client an IP address and subnet mask.

Note The command **`add ipatmclient`** always prompts you with a subnet mask that is appropriate for the IP address. However, if necessary, you can change the subnet mask to correspond to the strategy employed within your networks.

Caution Never create an IP over ATM VLAN (or an IP over ATM client) with the same subnet as the ATM SmartSwitch Ethernet port.

3. Enter the **show client** command to make sure the client is operational and to obtain the 20-byte ATM address of the ARP server. For instance, if you used the client number (client 1) from the example in step 2, enter the following command:

```
SmartSwitch # show client 1
IP/ATM Client 1
================================================================================
Client State      : Operational
Client Address    : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:00:5A:01:01:01:00
Server            : is local
Server Connection : Established
MTU               : 9180
IP Address        : 90.1.1.1
IP NetMask        : 255.255.255.0
SmartSwitch #
```

4. Physically connect your end nodes and edge devices to the ATM SmartSwitch ports.

Note Your end nodes do not need to be directly attached to the switch that contains the ARP server. For example, an end station is connected to an ATM SmartSwitch that is connected through a route to the switch containing the ARP server. No special configuration is needed for this end station to participate in the VLAN because the end station automatically finds its path across the route to the ARP server and the other VLAN members.

5. Configure the ATM interface or adapter for end nodes and edge devices. Typically, configuration consists of designating IP over ATM as the connection type, assigning the device an IP address, and specifying the 20-byte ATM address of the ARP server (the switch's client address). For details on the ATM SmartSwitch automatic addressing scheme for IP over ATM, see Section 2.1.1.
6. As your end devices are configured and started, they register with the ARP server. You can test whether your IP over ATM VLAN is functional by pinging from one end device to another.

To make certain that all end devices are registered with the ARP server, you can inspect the switch's ARP table using the **show ipatmarp** command. For example, if three end devices with IP addresses 90.1.1.2, 90.1.1.3, and 90.1.1.4 are added to the VLAN, the following ARP table entries should exist:

```
SmartSwitch # show ipatmarp
ClientNumber(ALL)                        :
IP/ATM Server 2 ARP Table
IP Address         ATM Address
================================================================================
90.1.1.2 39:00:00:00:00:00:00:00:00:00:14:41:80:00:00:5A:01:01:02:00
IP/ATM Server 3 ARP Table
IP Address         ATM Address
================================================================================
90.1.1.3 39:00:00:00:00:00:00:00:00:00:14:41:80:00:00:5A:01:01:03:00
IP/ATM Server 4 ARP Table
IP Address         ATM Address
================================================================================
90.1.1.4 39:00:00:00:00:00:00:00:00:00:14:41:80:00:00:5A:01:01:04:00
SmartSwitch #
```

Note If configured devices fail to join the VLAN, see Chapter 4, "Routing." Section 4.3. Also, see Chapter 9, "Troubleshooting."

2.1.1 Default ATM Addressing for IP over ATM

ATM SmartSwitches provide a default format for ATM addresses used by IP over ATM.

Note SmartSwitch 2500 family ATM switches and SmartSwitch 6500 switches use different methods for producing the default netprefix.

Default Netprefix for SmartSwitch 2500 Family Switches

The default **netprefix** is constructed from

39 + nine zero bytes + last three bytes of CPU MAC address

For example, if the chassis MAC address = 00:20:D4:14:41:80, then

Default netprefix = 39:00:00:00:00:00:00:00:00:00:14:41:80

Default Netprefix for SmartSwitch 6500

The default **netprefix** is constructed from

39 + nine zero bytes + last three bytes of chassis MAC address

For example, if the chassis MAC address = 00:00:1D:80:A3:34, then

Default netprefix = 39:00:00:00:00:00:00:00:00:00:80:A3:34

Default IP Over ATM Local Client Address

The default **Local client** address is constructed from

netprefix + two zero bytes + client IP address (in hexadecimal) + trailing zero byte

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

chassis MAC address = 00:00:1D:A3:87:0B

client IP address = 90.1.1.1 (5A.01.01.01 in hexadecimal)

then,

IP over ATM client address = 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:5A:01:01:01:00

2.2 CREATING AN EMULATED LAN

This section describes the steps for implementing an Emulated LAN (ELAN) on your SmartSwitch ATM switch.

Note If LANE services are to be reached through a virtual port on an ATM SmartSwitch, this switch must be a SmartSwitch 6500. Only the SmartSwitch 6500 supports logical multicasting. If LANE services are NOT reached through a virtual port, LANE services can reside on any ATM SmartSwitch.

The following assumptions are made:

- The ATM SmartSwitch will contain a client on the ELAN
- All end nodes (computers, edge devices, other switches, and so on) support the Well Known LECS Address or the Anycast Address or can obtain the address of the LECS using ILMI
- All end nodes support Switched Virtual Circuits (SVCs)

Note An ELAN comes pre-configured on all SmartSwitch ATM switches. The ELAN name is "ELAN000." To use this ELAN, start the LECS, configure your end nodes and edge devices to use ELAN name ELAN000, and then plug them into the ATM SmartSwitch.

1. Enter the **`start lecs`** command to activate LANE server services on this ATM SmartSwitch.

```
SmartSwitch # start lecs
NOTICE - 'LECS' ***** LECS started ***** — This assumes the LES/BUS is running (default)
SmartSwitch #
```

2. Create an ELAN on your ATM SmartSwitch by executing the **`add elan`** command. The following is an example.

```
SmartSwitch # add elan
ELANNumber(0) : 1                — 1 is used instead of the default, (0)
ELANName(ELAN001): Marketing     — ELAN is named Marketing instead of the default, (ELAN001)
ConnectMethod(SVC):
ELANType(802.3)                  —The default (Ethernet) is used
Multipoint(YES) :
MTU(1516) :
ErrorLogEnable(NO)  :            — Take the default
MinimumTDEnable(NO) :            — Take the default
Distribute(PROXY) :
SmartSwitch #
```

3. Use the **`add laneclient`** command to create a client for the switch on the ELAN:

```
SmartSwitch # add laneclient
ClientNumber(0) :1               — One is used instead of the default, (0)
LanName(ELAN001) : Marketing     — ELAN name is Marketing, not the default, (ELAN001)
ServerType(LECS) :
ServerAddress()                  — No LANE server address is specified; see note below
IPAddress() : 90.1.1.1           — IP address and subnet mask are specified only as examples
NetMask(255.0.0.0): 255.255.255.0
MTU(1516) :
SmartSwitch #
```

Note When you create a client, it automatically finds the LECS address using ILMI.

Note The command **`add laneclient`** always prompts you with a subnet mask that is appropriate for the IP address. However, if necessary, you can change the subnet mask to correspond to the strategy employed within your networks.

As the local client joins the ELAN, the following messages are sent to the Event Log (see Chapter 9, "Troubleshooting." Section 9.5):

```
NOTICE - 'ZLESSRV'  LES Join 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:
14:41:82:00
NOTICE - 'ZLESSRV'  BUS Connect 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:
14:41:82:00
```

Caution Never create an ELAN (or ELAN client) with the same subnet as the ATM SmartSwitch's Ethernet port.

4. Enter the **`show client`** command verify that the client is operational.

```
SmartSwitch # show client 1
LANE Client 1
================================================================================
Client State      : Operational
Client Address    : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:81:00
LAN Name          : Marketing
LECS Addr Source  : ILMI
LECS Address      : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:80:01
LES Address       : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:82:02
LAN Type          : 802.3
MTU               : 1516
IP Address        : 90.1.1.1
IP NetMask        : 255.255.255.0
SmartSwitch #
```

Note While creating an ELAN client for the switch is not absolutely necessary, it does provide management connectivity with the switch over its ATM ports (instead of the Ethernet port). See Chapter 4, "Routing." Section 4.5 for information about how to reach switches not directly connected to the Ethernet network.

5. Physically connect your end nodes and edge devices to the ATM SmartSwitch ports.

6. Configure the ATM interface or adapter for all end nodes and edge devices. Typically, configuration consists of specifying LAN Emulation as the connection type, assigning the device an IP address that corresponds to the subnet of the switch's client, and indicating that you want the device to either

acquire the LECS address through ILMI or use the Well Known Address as the address for the LECS. For details on the ATM SmartSwitch automatic addressing scheme for LANE, see Section 2.2.1.

7. As each end device registers with the LES and BUS, messages are sent to the event log of the ATM SmartSwitch containing the LECS. You can check connectivity by pinging between end nodes.

Note If configured devices fail to join the ELAN, see Chapter 4, "Routing." Section 4.3. Also, see Chapter 9, "Troubleshooting."

Your ELAN is now operational. Additional ELANs can be created in the same way.

Note While it is possible for a single ELAN on an ATM SmartSwitch to support multiple subnets, in general, switch performance is best (and management easiest) when the “One-subnet-per-ELAN” rule is observed.

2.2.1 ATM Addressing for LAN Emulation

All ATM SmartSwitches provide default formats for ATM addresses used by LAN emulation entities (local client, LECS, LES, and BUS). The SmartSwitch 2500 family of ATM switches and the SmartSwitch 6500 use different methods for constructing these default addresses.

SmartSwitch 2500 Family Default LANE Addressing

The **netprefix** is constructed from:

39 + nine zero bytes + last three bytes of CPU MAC address

For example, the chassis MAC address = 00:20:14:41:80,

then

default netprefix = 39:00:00:00:00:00:00:00:00:00:14:41:80

The **local client** address is constructed from:

netprefix **+** CPU MAC address with last byte summed with the client number **+** zero selector byte

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:14:41:80

CPU MAC address = 00:20:D4:14:41:80,

client number = 5

then,

client five’s default ATM address = 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:85:00

The **LECS** address is constructed from:

netprefix + CPU MAC address + selector byte of 01

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:14:41:89

chassis MAC address = 00:20:D4:14:41:80

then,

default LECS address = 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:80:01

The **LES** and **BUS** have the same ATM address. LES and BUS addresses are constructed from:

netprefix + CPU MAC address with last byte summed with the ELAN number + numerical value two (2)

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

CPU MAC address = 00:20:D4:14:41:80

ELAN number = 3

then,

default LES and BUS addresses = 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:83:02

SmartSwitch 6500 Default LANE Addressing

The **netprefix** is constructed from:

39 + nine zero bytes + last three bytes of chassis MAC address

For example, the chassis MAC address = 00:00:1D:A3:87:0B,

then

default netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

The **local client** address is constructed from:

netprefix **+** CPU MAC address, with last byte summed with the client number **+** zero selector byte

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

chassis MAC address = 00:00:1D:A3:87:0B,

CPU MAC address = 00:20:D4:14:41:80,

client number = 5

then,

client five's default ATM address = 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:20:D4:14:41:85:00

The **LECS** address is constructed from:

netprefix + chassis MAC address + selector byte of 01

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

chassis MAC address = 00:00:1D:A3:87:0B

then,

default LECS address = 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:01

The **LES** and **BUS** have the same ATM address. LES and BUS addresses are constructed from:

netprefix + chassis MAC address + ELAN number summed with the numerical value two (2)

For example

netprefix = 39:00:00:00:00:00:00:00:00:00:A3:87:0B

chassis MAC address = 00:00:1D:A3:87:0B

ELAN number = 3

then,

default LES and BUS addresses = 39:00:00:00:00:00:00:00:00:00:00:1D:A3:87:0B:05

2.2.2 ELANs Across Multiple Switches

ELANs can exist within a single switch, or they can span multiple switches. When an ELAN spans multiple switches, it's important that all switches within the group use the same LECS (see note, below). The general rule is: "Within an administrative domain (a group of switches with related ELANs), there should be one and only one LECS." For this reason, never start the LECS on more than one switch within the administrative domain.

Note The exception to the statement above is that if LNNI is enabled, multiple, redundant LECS' and LES/BUS' can exist within the same administrative domain. See Section 2.2.7 "Using LNNI."

Note If an uplink, end node, or other ATM switch does not support PNNI, or if its version of ILMI is incompatible, it may be necessary to set up a static route between the device and the rest of the ELAN. See Chapter 4, "Routing."

2.2.3 Switch Clients

It is important to understand the concept of ATM SmartSwitch client connections. A switch client connection is actually a VLAN connection to the ATM SmartSwitch's CPU (Here, we use the term VLAN to mean any type of *"virtual LAN,"* whether LANE or IP over ATM.). This CPU connection appears as if the switch is an end station on the virtual LAN. The ATM SmartSwitch uses local clients to connect itself to the VLANs that it supports.

This is analogous to a phone company that supports a communication system. Even though the phone company maintains the circuits, a call to the phone company itself cannot be made unless the phone company has its own number and connection on its own phone system. Similarly, VLAN membership (and the reachability) of an ATM SmartSwitch on any particular VLAN depends upon whether the ATM SmartSwitch has a local client connection for that VLAN.

Clients are created using the command **`add laneclient`** for LAN emulation, and **`add ipatmclient`** for IP over ATM.

For example, the following command adds a switch client to the ELAN **`elan1`**:

```
SmartSwitch# add laneclient
ClientNumber(0)                          : 1
LanName(ELAN001)                         : elan1
ServerType(LECS)                         :
ServerAddress()                          :
IPAddress()                              : 90.1.1.45      — Just for this example
NetMask(255.255.0.0)                     :255.255.255.0   — Just for this example
MTU(1516)                                :
SmartSwitch#
```

Prior to creating this local client connection, end devices could communicate with each other through **`elan1`**, but they could not communicate with the SmartSwitch ATM switch, itself.

2.2.4 Distributed LANE Services

LANE services (LECS, LES, and BUS) can reside on different ATM SmartSwitches. For example, the LECS can reside on one ATM SmartSwitch, while the LES and BUS reside on another. Use the **`add lecselan`**, **`add leselan`**, and **`add buselan`** to distribute LANE services among ATM SmartSwitches.

The following steps create an ELAN with the LECS on switch SW1 and the LES and BUS on switch SW2.

1. Use the **`add buselan`** command to create the BUS on switch SW2:

```
SW2 # add buselan
ELANNumber(0)                           : 1          — We'll use ELAN number = 1 throughout the example
ELANName(ELAN001)                       : mis1       — We'll call the ELAN "mis1" throughout the example
ConnectMethod(SVC)                      :
ELANType(802.3)                         :
Multipoint(YES)                         :
MTU(1516)                               :
ErrorLogEnable(NO)                      :
MinimumTDEnable(NO)                     :

SW2 #
```

2. Use the **add leselan** command to create an LES on switch SW2:

```
SW2 # add leselan
ELANNumber(0)                           : 1
ELANName(ELAN001)                       : mis1
ConnectMethod(SVC)                      :
ELANType(802.3)                         :
Multipoint(YES)                         :
MTU(1516)                               :
ErrorLogEnable(NO)                      :
MinimumTDEnable(NO)                     :
ForwardPeakCellRate(0)                  :
BackwardPeakCellRate(0)                 :
Distribute(PROXY)                       :
BUSATMAddress(39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:81:02): — Created by add buselan

SW2 #
```

3. Use the **show leselan** command on SW2 to obtain the ATM address of the LES:

```
SW2 # show leselan 1

   ELAN : mis1

ELAN Number           : 1
ELAN Name             : mis1
ATM Address         : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:81   — ATM address of LES
:02
Max Frame Size        : 1516
Connection Method     : SVC
Distribute VPI/VCI    : 0/0
Distribute Method     : PROXY
ELAN Type             : 802.3
Multipoint            : YES
Error Logging         : NO
Min TD Negotiation    : NO
BUS Address           : 39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:81
:02

SW2 #
```

4. On switch SW1, use the command **add lecselan** to create the LECS:

```
SW1 # add lecselan
ELANNumber(0)                           : 1
ELANName(ELAN001)                       : mis1
LESAddress(39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:03):39:00:00:00:00:00:00:00:0
0:00:14:41:80:00:20:d4:14:41:81:02                                    — Specify the LES address on SW2
ELANType(802.3)                         :
MTU(1516)                               :
TLVSet()                                :

SW1 #
```

5. Use the **add laneclient** command on SW1 to add a client to the ELAN:

```
SW1 # add laneclient
ClientNumber(0)                         : 1
LanName(ELAN001)                        : mis1
ServerType(LECS)                        :
ServerAddress()                         :
IPAddress()                             : 90.1.1.22          — This IP address is for example only
NetMask(255.0.0.0)                      : 255.255.255.0      — This subnet mask is for example only
MTU(1516)                               :

SW1 #
```

6. Use the **show client** command on SW1 to see that the client has reached all the distributed LANE services and has successfully joined ELAN **mis1**.

```
SW1 # show client
ClientNumber(ALL)                        :

Client Type    IP Address      Server Type  Server Conn  Status
=============================================================================
    1  LANE    90.1.1.22       LECS         Established  Operational

SW1 #
```

Notice in the example above that creating an ELAN with distributed services is a process of building from the bottom up: First, the BUS is created so that its address can be specified to the LES. Next, the LES is created so that its address can be specified to the LECS. Finally, the LECS is created.

If needed, all three ELAN services can exist on separate switches. For example, the BUS can exist on one switch (use the **add buselan** command), the LES can exist on another switch (use the **add leselan** command), and the LECS can exist on another switch (use the **add lecselan** command).

Note If LNNI is enabled, each associated LES and BUS must reside on the same switch. See Section 2.2.7, “Using LNNI” for details.

2.2.5 ELAN Join Policies

ATM SmartSwitches provide control over the assigning of clients to ELANs. Control is accomplished by ELAN join policies. By default, ATM SmartSwitches have a single ELAN join policy defined — *Best Effort*. When a client attempts to join LANE services, the ATM SmartSwitch uses information provided by the client to performs the *Best Effort* ELAN join test.

Note Additional security can be achieved through the use of ATM address filtering. See Section 8.1 for information regarding ATM address filtering.

Best Effort Elan Join Test

The following describe the *Best Effort* test.

1. Does the client specify the name of the ELAN it wants to join?

- If yes, check whether an ELAN exists by that name. If an ELAN exists by that name, assign the client to the ELAN. If no ELAN exists by that name, assign the client to the default ELAN (ELAN 0).
- If no, check the client against the configuration information stored by the **add lecselanlec** command (see The LECSELANLEC Table, on page -13). If an entry exists that corresponds to the client, assign the client to the ELAN indicated. If the client does not correspond to an entry, assign it to the default ELAN (ELAN 0).

Note If the default ELAN (ELAN 0) has been deleted, the client is dropped.

By using ELAN join policies, clients attempting to join LANE services can be assigned to specific ELANs. Table 2-1 lists the ELAN join policies that can be configured on an ATM SmartSwitch.

Table 2-1 ELAN Join Policies

Policy No.	ELAN Join Policy	Information Source Checked
1	Best Effort	Default ELAN policy. Checks configuration information stored by the **`add lecselanlec`** command and during ELAN creation (**`add elan`** command).
2	By ATM Address	Checks configuration information stored by the **`add lecselanlec`** command.
3	By MAC Address	Checks configuration information stored by the **`add lecselanlec`** command.
4	By Route Descriptor	Checks configuration information stored by the **`add lecselanlec`** command.
5	By LAN Type	Checks configuration information stored during ELAN creation (**`add elan`** command).
6	By Packet Size	Checks configuration information from the **`add lecspacketsize`** command.
7	By ELAN Name	Checks configuration information stored by the **`add lecselannametable`** command.

Note For detailed information on each of the commands that ELAN join policies interacts with, see the command descriptions in the SmartSwitch ATM Reference Manual.

You can give each ELAN join policy a priority value to determine its hierarchy among other ELAN join policies. If you define several ELAN join policies, the policy with the greatest priority value is tried first. If that policy fails, the policy with the next greatest priority value is attempted, and so on. ELAN join policies with the same priority value are ANDed together. For example, if three join policies are create, each with the same priority value, a client requesting LANE services must meet the criteria of all three policies to be assigned an ELAN. If the client fails to meet the requirements of all three policies, the policy with the next lowest priority value will attempt to assign the client to an ELAN.

Use the **add lecselanpolicy** command to create ELAN join policies. The following is an example of creating an ELAN join policy based on the *By Packet Size* policy.

```
SmartSwitch # add lecselanpolicy
PolicyIndex()                              : 2       — Can be any value other than one (1)
Type()                                     : ?       — Use ? to see possible types
ELAN Policy Type (Values from 1 to 7 representing, in order, the policies BestEffort, byATMAddress,
byMacAddress, byRouteDescriptor, byLANType, byPacketSize and byELANName).
Type()                                  : 6      — Specify type 6, assign ELAN by packet size requested by client
Priority()                                 : 1000    — Weight the policy at 1000

SmartSwitch #
```

Note The lower the numerical value of a priority, the higher the priority. In the example above, a priority value of 1000 was specified. Subsequently, This policy will be tried before *Best Effort* (policy value = 65001).

Use the **show lecselanpolicy** command to show the newly created ELAN join policy.

```
SmartSwitch # show lecselanpolicy

 Index   Assignment Policy           Priority Value
================================================================================
    1    Best Effort (Proprietary)       65001
    2    By Packet Size                   1000   — The created policy, its index number, and its priority

SmartSwitch #
```

Note In the example above, index 2 (or greater) was used because the *Best Effort* policy reserves index one.

The LECSELANLEC Table

Many of the ELAN join policies use the information supplied by the **add lecselanlec** command. Use the **add lecselanlec** command to create a list of clients and to assign the ELAN each client should join.

Note You can also assign a TLV set to be used by the client on the specified ELAN.

Clients are identified within the **lecselanlecs** list by one (or a combination of) the following attributes:

- ATM address
- MAC address
- Token Ring route descriptor (segment ID and bridge number)
- IP address

In the following example, a client is identified by its ATM address and IP address, and associates it with ELAN number 1.

```
SmartSwitch # add lecselanlec
AtmAddress()                          : 39:00:00:00:00:00:00:00:00:00:44:55:66:11:22:33:44:55:66:00
MACAddress/RouteDesc()                :      — No MAC address is specified
Layer3Address[IP]()                   : 204.123.91.7
ELANNumber(0)                         : 1    — ELAN is specified by ELAN number
TLVSet()                              :      — No TLV set is specified

SmartSwitch #
```

If the currently defined ELAN policies use either *Best Effort* or *By ATM Address* and/or *By IP Address*, the client with the ATM address and IP address specified above will be assigned to ELAN 1.

Note To specify a TLV set with the **`add lecselanlec`** command, the TLV set must currently exist. Use the **`add lecstlvset`** command to create a TLV set. For detailed information on the **`add lecstlvset`** command, see the SmartSwitch ATM Reference Manual.

2.2.6 LANE Over WAN Circuits

SmartSwitch ATM switches allows LANE server support across WAN ATM connections. In this type of configuration, a SmartSwitch running LANE services (LECS, LES and BUS) resides on one side of an ATM WAN, while SmartSwitch ATM switches on the other side of the WAN provide connectivity for LANE clients across the WAN to the LANE server. In effect, the connections created between the LANE server and its clients “tunnel” across the ATM WAN’s PVP connections.

Note See Chapter 5, "Virtual Ports and Static Connections." for information about PVP connections and virtual ports.

Physical Versus Logical BUS Multicasting

When connecting to LANE services across an ATM WAN, it’s important to consider the WAN-to-LAN connectivity. Typically, PVPs (assigned by services provides) are terminated on the end switches using virtual ports. In a simple configuration, with a single PVP terminated by a single virtual port at each end, clients submitting ELAN join requests can traverse the WAN and reach LANE services. Likewise, the LANE servers (especially the BUS) can reply back across this single connection. In effect, all traffic between the end switches is “tunneled” across the PVP WAN connection. In this case, the BUS creates its point-to-multipoint client connections using physical multicasting across the WAN (see Figure 2-1).

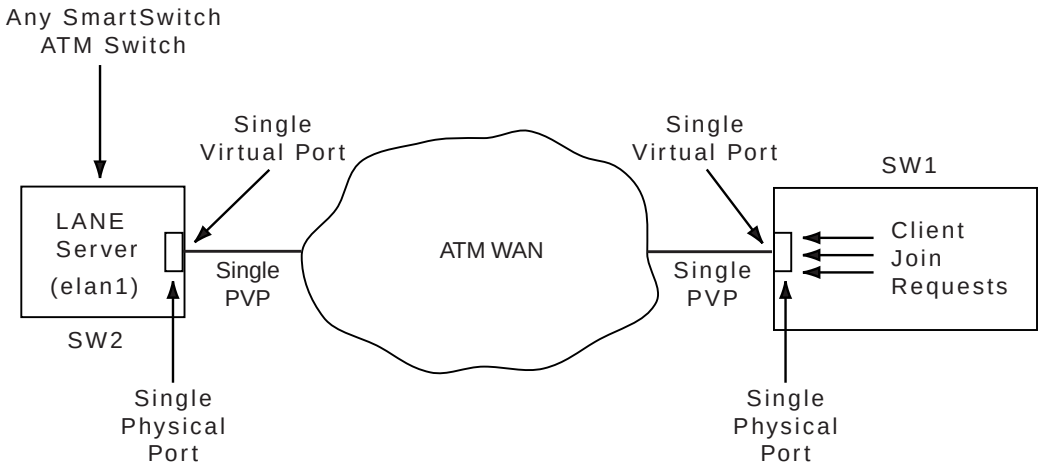

Figure 2-1 Single PVP connection between clients and LANE services

Physical BUS multicasting implies that the BUS performs multicasting according to physical ports. With a single PVP, the BUS understands that all requests are coming from a particular port. Accordingly, the BUS replies over that port, and it is up to the switch at the other end of the PVP connection to sort out which reply belongs to which client (see Figure 2-2).

Another possible ATM WAN configuration involves multiple PVPs across the WAN, with each PVP terminated on its own virtual port, and all virtual ports residing on the same physical port. In this configuration, LANE join requests for the same ELAN may appear on different virtual ports of the same physical port of the switch running LANE services. Because these requests are appearing on multiple logical entities (multiple virtual ports), this requires the BUS to be capable of logical multicasting.

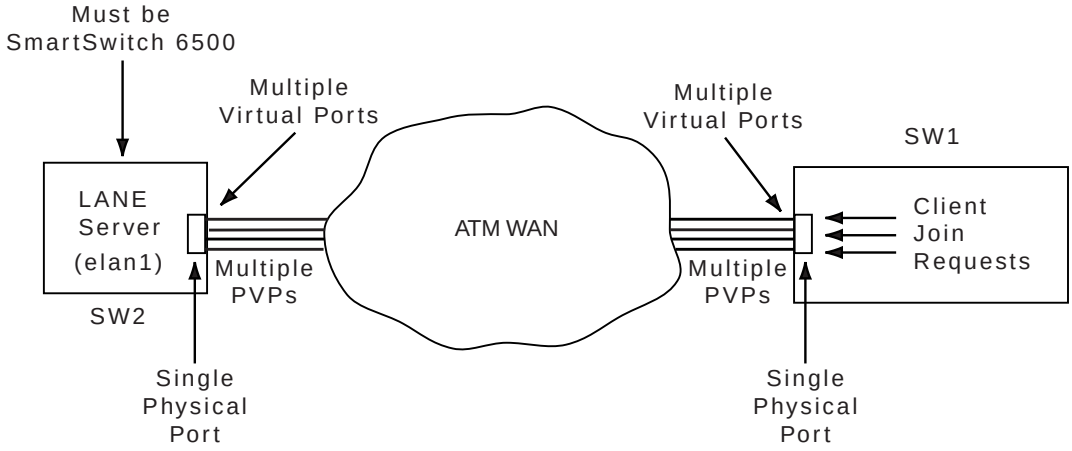

Figure 2-2 Multiple PVP connection between clients and LANE services

Logical BUS multicasting implies that the BUS of a particular ELAN can distinguish the difference between virtual ports on the same physical port. In essence, the BUS treats each virtual port as a physical entity, and keeps track of its point-to-multipoint connections to the clients through various PVPs.

Currently, the SmartSwitch 6500 is the only SmartSwitch ATM switch that supports logical multicasting. For this reason, if you are connecting to LANE services across an ATM WAN using multiple PVPs and if client join requests for the same ELAN are received over different PVPs, you must use a SmartSwitch 6500 as the LANE services switch. If on the other hand, your WAN connection consists of a single PVP, any of the SmartSwitch ATM switches can be used as the LANE services switch.

The rules for selecting the appropriate SmartSwitch ATM switch for providing LANE services across an ATM WAN are summarized below:

- A single PVP connection terminated on the LANE server switch with a single virtual port — Any SmartSwitch ATM switch as the LANE server (physical BUS multicasting)
- Multiple PVP connections terminated on the LANE server switch through virtual ports on the same physical port, where each PVP supports client connection requests for separate ELANs — Any SmartSwitch ATM switch (physical BUS multicasting)
- Multiple PVP connections terminated on the LANE server switch through virtual ports on different physical ports — Any SmartSwitch ATM switch (physical BUS multicasting)
- Multiple PVP connections terminated on the LANE server switch through virtual ports on the same physical port, where each PVP supports client connection requests for the same ELAN — SmartSwitch 6500 only (logical BUS multicasting required).

2.2.7 Using LNNI

SmartSwitch ATM switches provide support for LNNI. LNNI gives LANE redundancy and load-sharing capabilities by allowing multiple LECSs to exist on the same network, and by allowing multiple LES/ BUSs and SMSs to service the same ELANs.

Note For an explanation of all LNNI related commands and parameters, see the SmartSwitch ATM Switch Reference Manual.

LANE Service Redundancy

As many as eight (8) LECSs (one per SmartSwitch ATM switch) can be deployed on the same network; each LECS can support multiple ELANs. This is especially useful on large, mission-critical networks and eliminates the possibility of the LECS being a potential single point-of-failure. If, for some reason, LANE services go down on a particular switch, the clients that this switch supports can reestablish their connection to their ELAN through one of the other LECSs (see Figure 2-3).

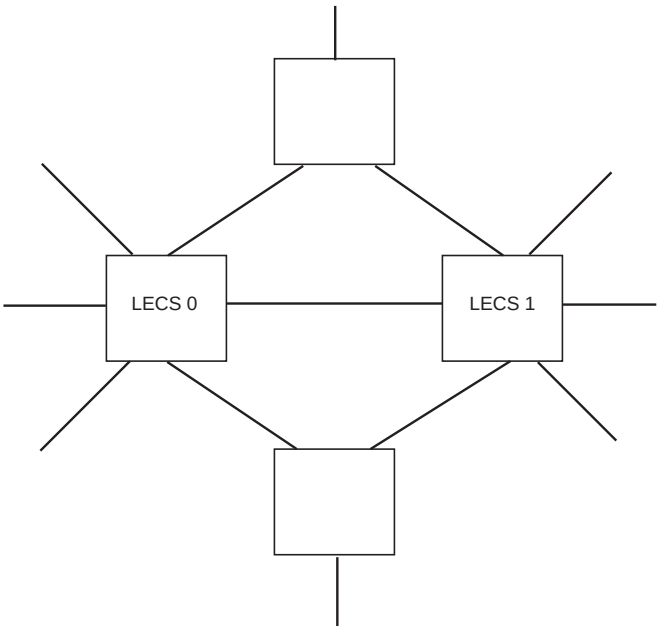

Figure 2-3 LNNI Redundant LECSs on same network

LANE Load Sharing

Running multiple LECSs, alleviates the bottleneck of a single LECS supporting all clients on all ELANs. Under LNNI, a client requesting a call setup is serviced by the LECS, LES and BUS on the switch that it's directly connected to, leaving other SmartSwitch ATM switches free to service the call setups from their directly attached clients (see Figure 2-4).

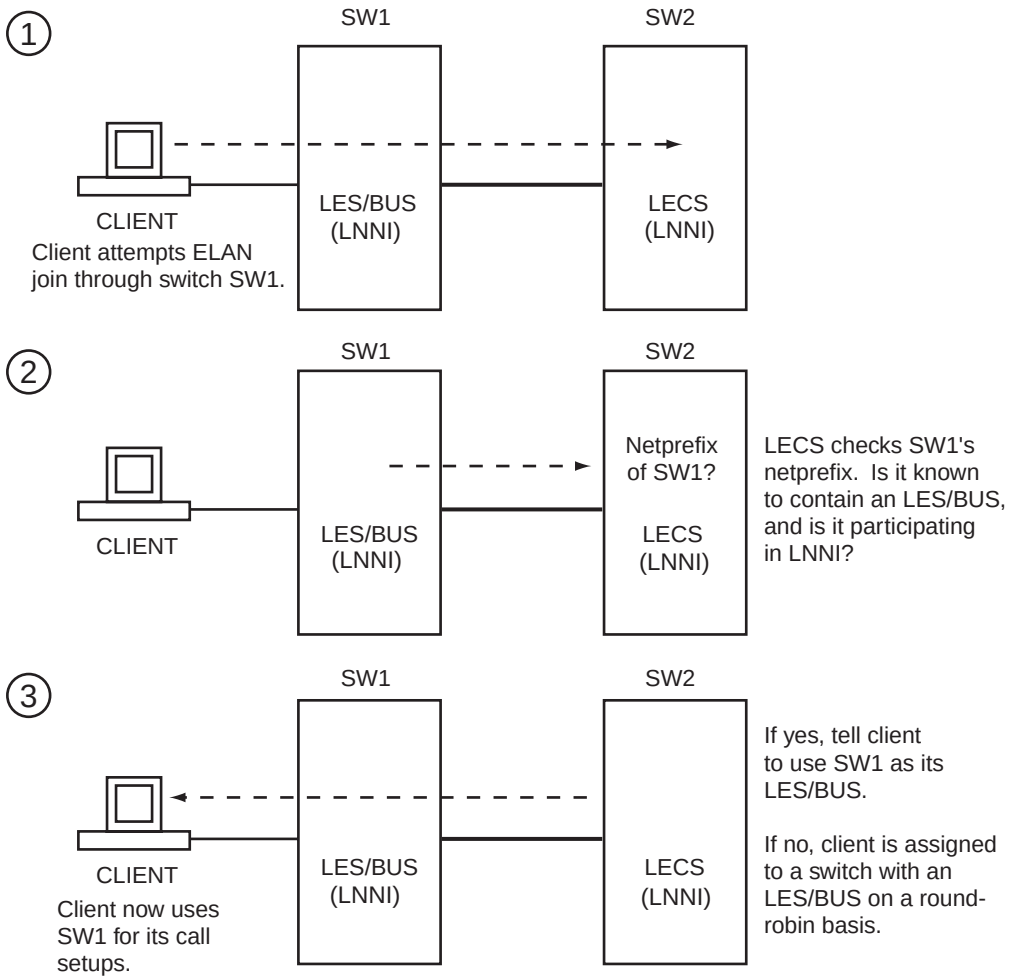

Figure 2-4 LNNI call set up load sharing

Additional load sharing can be achieved using LNNI and distributed LANE services. Using distributed LANE, LNNI allows each switch containing an LECS to support up to eight (8) LES/BUSs on eight other (separate) switches on the same ELAN. This allows for a possible 64 LES/BUSs supporting each ELAN.

When a client attempts an ELAN join, the LECS checks the netprefix of the switch through which the client is attempting to join. If the netprefix of the switch corresponds to a switch known to be participating in LNNI and containing an LES/BUS, the LECS assigns the client to the LES/BUS on its directly connected switch. This keeps the client's call setups local to his directly attached switch, and allows other LES/BUSs (on other switches) free to service the call setups of their locally attached clients.

For example, In Figure 2-5, Clients A, B, and C are assigned to the LES/BUS of the switch to which each is physically attached. Client D's switch is not running an LES/BUS under LNNI, and is assigned to an LES/BUS on some other switch.

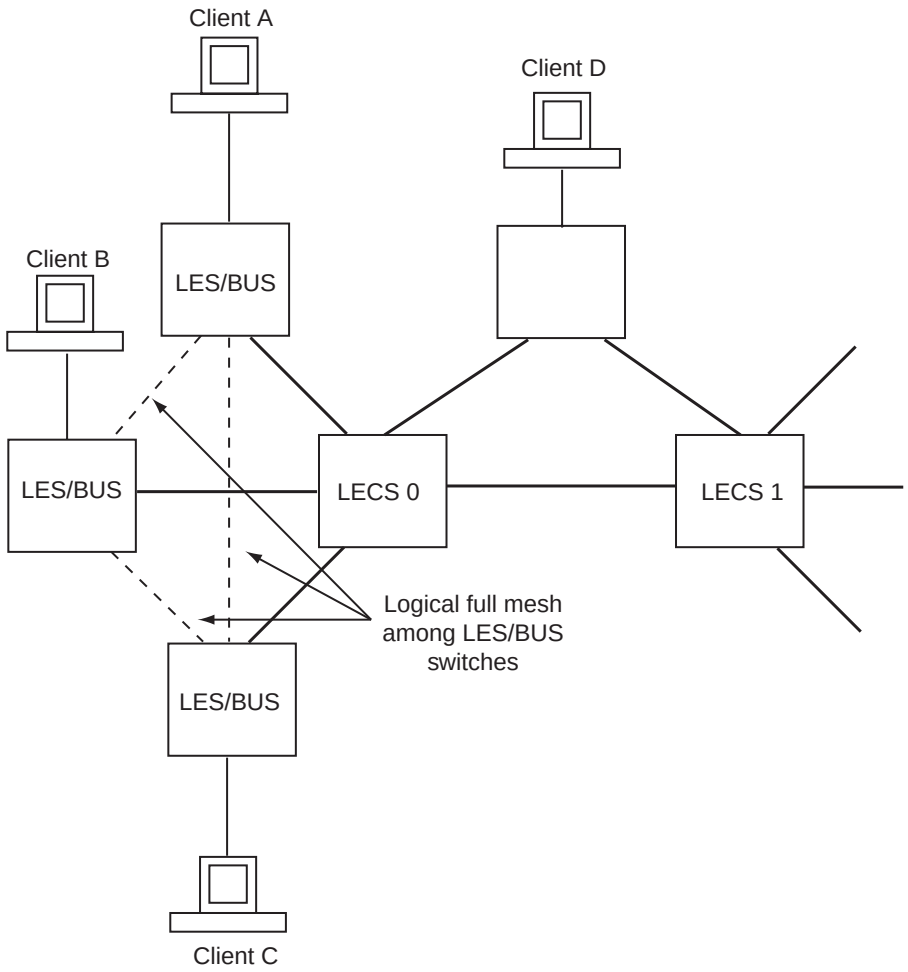

Figure 2-5 How LNNI handles ELAN join requests

Setting up LNNI LECs

The procedure for setting up LNNI on a SmartSwitch ATM switch is performed by executing the following basic steps:

- Shut down all LANE services — LECS, LES and BUS
- Configure LNNI
- Enable LNNI
- Start LANE services

The following is an example of enabling LNNI on a network and configuring neighbor LECSs on two separate switches (SW1 and SW2).

1. On both SW1 and SW2, enter the **stop lecs** command to make sure each LECS is down

```
SW1 # stop lecs

Confirm(y/n)?:y
NOTICE - 'LECS' ***** LECS shutdown *****

SW1 #
```

2. On both SW1 and SW2, enter the **stop les** command to stop each switch's LES and BUS

```
SW1 # stop les

          STOPPING  LES/BUS

Confirm(y/n)?:y
NOTICE - 'ZLESSRV' ***** LES shutdown *****

SW1 #
```

3. On both SW1 and SW2, enter the **set lnniinfo** command to assign a number to each switch's LECS. Make sure that each **LECSID** is unique.

```
SW1 # set lnniinfo
LECSID(-1)                              : 0   — On SW1, LECSID will be zero

SW1 #
```

Similarly, on SW2, enter the **set lnniinfo** command, specifying a different LECSID for SW2

```
SW2# set lnniinfo
LECSID(-1)                              : 1   — On SW2, LECSID will be one

SW2 #
```

Note The default **LECID** -1, indicates that the LECS is not used on this switch. The default value (-1) is used as the **LECID** on switches participating in LNNI that are running only the LES/BUS (see next section, "Configuring LNNI Distributed LES/BUS servers").

4. On both SW1 and SW2 enter the **set lnnistatus** command to enable LNNI and SCSP (Server Cache Synchronization Protocol).

```
SW1 # set lnnistatus
LNNIStatus(Disabled)                     : enable
SCSPStatus(Disabled)                     : enable
SW1 #
```

Enter the **show lnnistatus** command to make certain that LNNI has started on each switch

```
SW1 # show lnnistatus
LNNI Status                  : Enabled
SCSP Status                  : Enabled
SW1 #
```

5. On both SW1 and SW2, use the **start les** and **start lecs** commands to start LANE services

```
SW1 # start les
NOTICE - 'ZLESSRV' ***** LES started *****

SW1 # start lecs
NOTICE - 'LECS' ***** LECS started *****

SW1 #
```

6. On SW1, create an ELAN; in this example, we create **elan1**:

```
SW1 # add elan
ELANNumber(0)                          : 1
ELANName(ELAN001)                      : elan1
ConnectMethod(SVC)                     :
ELANType(802.3)                        :
Multipoint(YES)                        :
MTU(1516)                              :
ErrorLogEnable(NO)                     :
MinimumTDEnable(NO)                    :
Distribute(PROXY)                      :

SW1 #
```

Similarly, create the same ELAN (**elan1**) on SW2:

```
SW2 # add elan
ELANNumber(0)                          : 1
ELANName(ELAN001)                      : elan1
ConnectMethod(SVC)                     :
ELANType(802.3)                        :
Multipoint(YES)                        :
MTU(1516)                              :
ErrorLogEnable(NO)                     :
MinimumTDEnable(NO)                    :
Distribute(PROXY)                      :

SW2 #
```

7. On SW1, enter the **show elan 1** command to obtain the ATM address of the LECS on that switch

```
SW1 # show elan 1
ELAN  1

================================================================================
ELAN Number       : 1
LECS Address     : 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:01 — LECS address on SW1
LES Address       : 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:03
ELAN Name         : elan1
ELAN Type         : 802.3
MTU               : 1516
Connection Method : SVC
Distribute VPI/VCI: 0/0
Distribute Method : PROXY
Multipoint        : YES
Error Logging         : NO
Min TD Negotiation    : NO



SW1 #
```

Similarly, enter the **show elan 1** command on SW2 to obtain SW2's LECS address

```
SW2 # show elan 1
ELAN  1

================================================================================
ELAN Number       : 1
LECS Address     : 39:00:00:00:00:00:00:00:00:00:BF:BA:26:00:00:1D:BF:BA:26:01 — LECS address on SW2
LES Address       : 39:00:00:00:00:00:00:00:00:00:BF:BA:26:00:00:1D:BF:BA:26:03
ELAN Name         : elan1
ELAN Type         : 802.3
MTU               : 1516
Connection Method : SVC
Distribute VPI/VCI: 0/0
Distribute Method : PROXY
Multipoint        : YES
Error Logging         : NO
Min TD Negotiation    : NO



SW2 #
```

8. On SW1 use the **add lecsneighbor** command to specify the ATM address of the LECS on SW2

```
SW1 # add lecsneighbor
NeighborATMAddress()                 : 39:00:00:00:00:00:00:00:00:00:bf:ba:26:00:00:1d:bf:ba:26:01

SW1 #
```

Similarly, on SW2 use the **add lecsneighbor** command to specify the ATM address of the LECS on SW1

```
SW2 # add lecsneighbor
NeighborATMAddress()                 : 39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b:01

SW2 #
```

The LECSs on switch SW1 and SW2 are now configured for LNNI and are running redundantly. If, for example, LANE services goes down on SW1, its clients can rejoin the ELAN by registering with LANE services on SW2.

Use the **show lecsneighborinfo** command on any LNNI active switch running an LECS to see a list of known neighbor LECSs. For example, on SW1, entering **show lecsneighborinfo** shows information about SW2:

```
SW1 # show lecsneighborinfo
LECS Sync PMP VCC VPI/VCI : 0/48

                                                        Outgoing   Incoming
                    Neighbor ATM Address                State      VPI/VCI
================================================================================
39:00:00:00:00:00:00:00:00:00:BF:BA:26:00:00:1D:BF:BA:26:01 Active       0/49


SW1 #
```

Configuring LNNI Distributed LES/BUS Servers

Under LNNI each switch running an LECS is capable of supporting eight (8) switches running an LES/BUS on the same ELAN. LES/BUS neighbor information is distributed to the LES/BUS switches by the LECSs. However, server cache information is distributed among the LES/BUS servers themselves using SCSP (Server Cache Synchronization Protocol). To assure that SCSP information can be exchanged between all LES/BUS switches, the switches should be

connected by a logical full-mesh topology. In this case, the term "*logical*" means only that all LNNI switches participating within a particular domain should be able to reach each other. Typically, a full-mesh topology is satisfied by PNNI, and does not require all LES/BUS switches to be directly connected.

The following is an example of configuring a distributed LNNI LES/BUS on SW3. This example continues from the example above — Two LECS' are running redundantly for ELAN 1 (**elan1**).

1. On switch SW3, enter the **stop lecs** command on the switch to contain the LES/BUS. This is done to make sure the LECS is not running on this switch.

```
SW3 # stop lecs

Confirm(y/n)?:y
NOTICE - 'LECS' ***** LECS shutdown *****

SW3 #
```

2. On switch SW3, use the **add buselan** command to associate this switches BUS with the ELAN on switches SW1 and SW2 (**elan1**).

```
SW3 # add buselan
ELANNumber(0)                          : 1
ELANName(ELAN001)                      : elan1
ConnectMethod(SVC)                     :
ELANType(802.3)                        :
Multipoint(YES)                        :
MTU(1516)                              :
ErrorLogEnable(NO)                     :
MinimumTDEnable(NO)                    :

SW3 #
```

3. On switch SW3, use the **add leselan** command to associate this switches LES with the ELAN on switches SW1 and SW2 (**elan1**).

```
SW3 # add leselan
ELANNumber(0)                          : 1
ELANName(ELAN001)                      : elan1
ConnectMethod(SVC)                     :
ELANType(802.3)                        :
Multipoint(YES)                        :
MTU(1516)                              :
ErrorLogEnable(NO)                     :
MinimumTDEnable(NO)                    :
Distribute(PROXY)                      :
BUSATMAddress(39:00:00:00:00:00:00:00:00:00:BD:AE:20:00:00:1D:BD:AE:20:03):

SW3 #
```

4. On switch SW3, use the **stop les** command to stop the LES/BUS service

```
SW3 # stop les

          STOPPING  LES/BUS

Confirm(y/n)?:y
NOTICE - 'ZLESSRV' ***** LES shutdown *****

SW3 #
```

5. On switch SW3, use the **set lnniinfo** to configure LNNI

```
SW3 # set lnniinfo
LECSID(-1)                             :     — Accept -1, there will be no LECS on this switch

SW3 #
```

6. On switch SW3, use the **`set lnnistatus`** command to enable LNNI and SCSP (Server Cache Synchronization Protocol).

```
SW3 # set lnnistatus
LNNIStatus(Disabled)                    : enable
SCSPStatus(Disabled)                    : enable
SW3 #
```

Note SCSP does not have to be enabled for an LES to take part in LNNI. However, without SCSP enabled, ARP server information is not shared. As a result, client connects may be slowed by the client's need to broadcast to find the LES with the appropriate ARP information.

7. On SW3, use the **`start les`** command to activate the switch's LES and BUS.

```
SW3 # start les
NOTICE - 'ZLESSRV' ***** LES started *****

SW3 #
```

Once the LES/BUS is started, it registers with each LECS running LNNI on the network. In turn, the LECS' communicate the LES/BUS' existence to all other distributed LES/BUS' participating in LNNI. Finally, the LES/BUS on SW3 begins exchanging server cache information (through SCSP) with other LNNI LES/BUS'.

To see a list of servers (LES/BUS or SMS servers) known to a particular LNNI LECS, enter the **show lecsserverlist** command on a switch running an LNNI LECS:

```
SW1 # show lecsserverlist
ELANNumber(ALL)                          : 1

LES/SMS servers known for ELAN 1
================================================================================

ATM Address        : 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:03
Learned From (LECS): 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:01
Type               : LES
Alive Time (secs)  : 28
Locally Attached   : Yes
Config Direct VCC  : 0/47
Server ID          : 0x0000
LECID Range        : 0x0001 - 0x03FF

ATM Address        : 39:00:00:00:00:00:00:00:00:00:BD:AE:20:00:00:1D:BD:AE:20:03
Learned From (LECS): 39:00:00:00:00:00:00:00:00:00:A3:87:0B:00:00:1D:A3:87:0B:01
Type               : LES
Alive Time (secs)  : 27
Locally Attached   : Yes
Config Direct VCC  : 0/59
Server ID          : 0x0001
LECID Range        : 0x0400 - 0x07FF

ATM Address        : 39:00:00:00:00:00:00:00:00:00:BF:BA:26:00:00:1D:BF:BA:26:03
Learned From (LECS): 39:00:00:00:00:00:00:00:00:00:BF:BA:26:00:00:1D:BF:BA:26:01
Type               : LES
Alive Time (secs)  : 21
Locally Attached   : No      — LES/BUS of this switch (SW3) is not associated with switch SW1
Config Direct VCC  : --
Server ID          : --
LECID Range        : --

SW1 #
```

In this example, **show lecsserverlist** is entered on SW1. Notice that the parameter **Locally Attached** indicates whether the server is associated with the LECS on the switch on which the **show lecsserverlist** command was executed. If the server is associated with this switch's LECS (SW1), **Locally Attached** returns **yes**. If the server is associated with an LECS on a different switch, **Locally Attached** returns **no**.

3 PNNI ROUTING

All ATM SmartSwitches use PNNI version 1.0 as their default routing protocol. PNNI provides automatic and dynamic connectivity among all PNNI nodes within the same peer group. By configuring multi-level PNNI topologies and peer group leaders, full hierarchical PNNI routing can be established with connectivity between different peer groups.

Note For a complete explanation of all PNNI related commands, see the SmartSwitch ATM Reference Manual.

3.1 PNNI NODE ADDRESSING

By default, all ATM SmartSwitches come configured with a single PNNI node. All PNNI nodes are in the same peer group and at the same group level.

3.1.1 Default PNNI Addressing

All PNNI entities on SmartSwitch ATM switches are assigned default values (which can be changed). The following describes the formulae used in creating these values.

Default Peer Group ID = 50:39:00:00:00:00:00:00:00:00:00:00:00

Default Group Level = 80 (50 hexadecimal)

SmartSwitch 2500 Family Default Node ID

Default Node ID = level + child node's peer group level (see note) + 39 + nine zero (00) bytes + last three bytes of CPU MAC address + CPU MAC address with 127 summed with the last byte + zero (00) byte

Note If the node does not have a child node, and the node is also at the lowest level, the second byte is assigned the constant value A0 (160 decimal).

For example, for a node at the lowest level (80), the *level* and *address length* bytes are 50 (80 in hexadecimal) and a0 (160 in hexadecimal), respectively.

SmartSwitch 6500 Family Default Node ID

Default Node ID = level + child node's peer group level (see note) + 39 + nine zero (00) bytes + last three bytes of chassis MAC address + switch MAC address with 127 summed with the last byte + zero (00) byte

Note If the node does not have a child node, and the node is also at the lowest level, the second byte is assigned the constant value A0 (160 decimal).

For example, for a node at the lowest level (80), the *level* and *address length* bytes are 50 (80 in hexadecimal) and a0 (160 in hexadecimal), respectively.

SmartSwitches assign default Node ATM Addresses based on the following format:

SmartSwitch 2500 Family Default Node ATM Address

Default Node ATM Address = 39 + nine zero (00) bytes + last three bytes of CPU MAC address + CPU MAC address with 127 summed with the last byte + byte containing node index starting at zero (0) for the first node

Use the **`show pnninode`** command to view SmartSwitch ATM switch PNNI node parameters. For example:

```
SmartSwitch # show pnninode
NodeIndex(1)                              :
================================================================================
Node Index   : 1
Node Level   : 80
Node Id      : 50:a0:39:00:00:00:00:00:00:00:00:14:59:00:00:20:d4:14:59:7f:00
Lowest       : True
Admin Status : Up
Oper Status  : Up
Node ATM Addr: 39:00:00:00:00:00:00:00:00:00:14:59:00:00:20:d4:14:59:7f:00
Peer Group Id: 50:39:00:00:00:00:00:00:00:00:00:00:00
Rst Transit  : False
Complex Rep  : False
Rst Branching: False
DB Overload  : False
Ptse         : 2

SmartSwitch #
```

Note Keep in mind that the **`Node ATM Address`** is not the same as the ATM address of the switch client (if any). The **`Node ATM Address`** is used by PNNI to identify PNNI nodes and does not correspond to LANE entities.

SmartSwitch 6500 Default Node ATM Address

Default Node ATM Address = 39 + nine zero (00) bytes + last three bytes of chassis MAC address + CPU MAC address with 127 summed with the last byte + byte containing node index starting at zero (0) for the first node

Use the **show pnninode** command to view ATM SmartSwitch PNNI node parameters. For example:

```
SmartSwitch # show pnninode
NodeIndex(1)                                :
================================================================================
Node Index   : 1
Node Level   : 80
Node Id      : 50:a0:39:00:00:00:00:00:00:00:00:83:91:e5:00:20:d4:29:0e:ff:00
Lowest       : True
Admin Status : Up
Oper Status  : Up
Node ATM Addr: 39:00:00:00:00:00:00:00:00:00:83:91:e5:00:20:d4:29:0e:ff:00
Peer Group Id: 50:39:00:00:00:00:00:00:00:00:00:00:00
Rst Transit  : False
Complex Rep  : False
Rst Branching: False
DB Overload  : False
Ptse         : 2

SmartSwitch #
```

Note Keep in mind that the **Node ATM Address** is not the same as the ATM address of the switch client (if any). The **Node ATM Address** is used by PNNI to identify PNNI nodes and does not correspond to LANE entities.

3.2 MULTI-LEVEL PNNI TOPOLOGY

Having all ATM switches on your network in the same peer group is a simple way of assuring connectivity between all nodes. However, depending on the size and complexity of your network, there are advantages to dividing your PNNI network into different peer groups and levels. The basic steps for creating multiple peer groups and multiple levels are as follows:

- Set the peer group IDs of ATM SmartSwitches to differentiate their peer group membership.
- Select one (or more) ATM SmartSwitch within each peer group as the Peer Group Leader (PGL).
- Add a higher-level PNNI node to each PGL switch. This higher-level node represents its peer group as a Logical Group Node (LGN) within the next highest (parent) peer group. Connectivity between the peer groups is established within the parent peer group.
- Communicate the PGL's existence to the rest of the peer group by setting its leadership priority.
- Physically connect the two peer groups.

3.2.1 Connecting Multiple Peer Groups

This section presents a practical, step-by-step example of creating a multi-level, multiple peer group topology. The example is based on the following components and organization (see Figure 3-1).

- Six ATM SmartSwitches divided into two peer groups:
  - Three ATM SmartSwitches in peer group A (switches SWA1, SWA2, and SWA3)
  - Three ATM SmartSwitches in peer group B (switches SWB1, SWB2, and SWB3)

1. Physically connect switches SWA1, SWA2, and SWA3. Similarly, physically connect switches SWB1, SWB2, and SWB3 (see Figure 3-1).

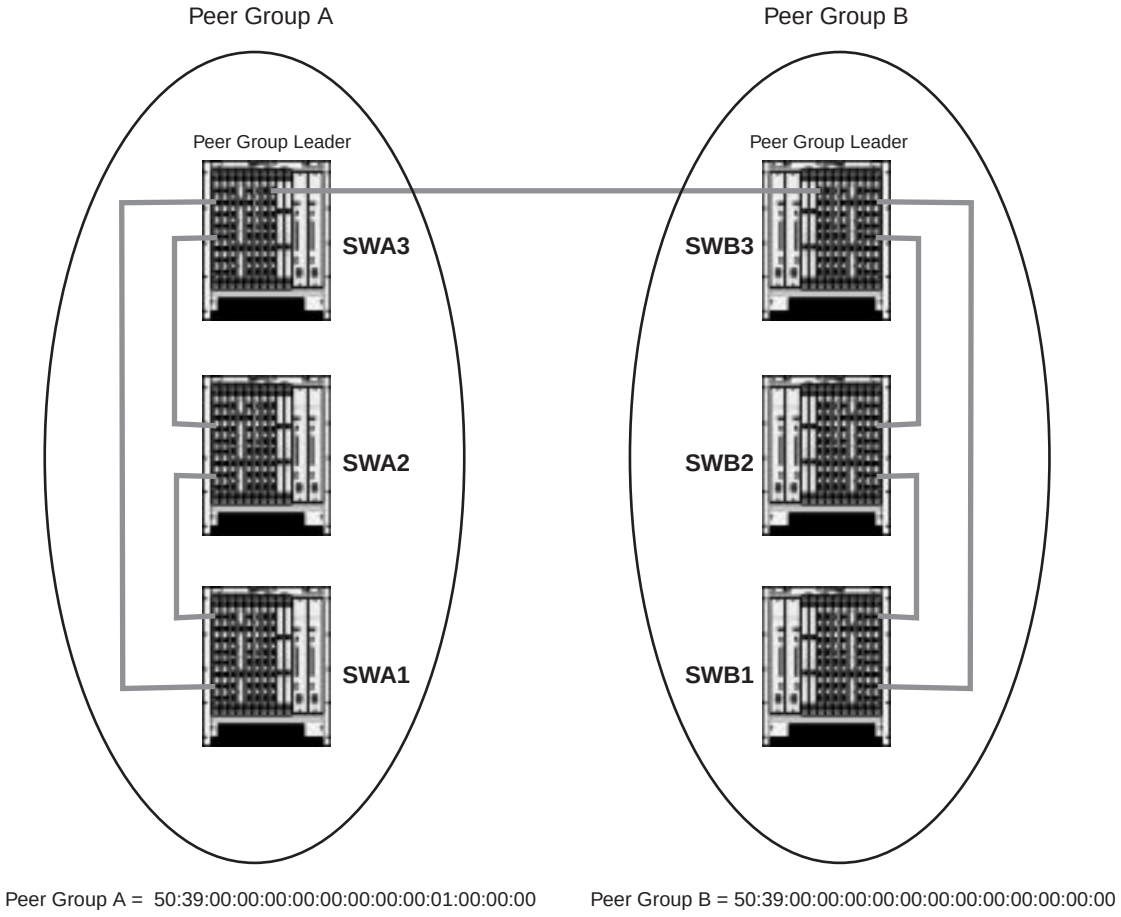

Figure 3-1 Physical connectivity for multi-peer group example

2. Use the **`set pnnipeergroupid`** command to change the peer group ID of the switches in group A to 50:39:00:00:00:00:00:00:00:00:00:01:00:00:00:00. The three remaining switches with the default peer group ID will comprise group B:

```
SWA1 # set pnnipeergroupid
NodeIndex(1)                                :
PeerGroupId(50:39:00:00:00:00:00:00:00:00:00:00:00:00): 50:39:00:00:00:00:00:00:
00:00:01:00:00:00 — Change the tenth byte to 01

Console: You have changed the node configuration.  If this node has a parent node,
make sure its parent node configuration is compatible with the new configuration.

Console: You will have to reboot for the new node configuration to take effect.

SWA1 #
```

Reboot the switch, and repeat the process for switches SWA2 and SWA3.

Note The first byte of the peer group ID indicates the peer group's level. It also indicates the number of significant bits used in the peer group ID. For example, if the level indicator is 50 (80 decimal), then 80 bits / 8 = 10 bytes; and only 10 of the 13 bytes are significant (39:00:00:00:00:00:00:00:00:00). If you create a new peer group ID, make sure that the bytes you change are within the range of significant bytes for the peer group's level.

3. Use the **show pnnilink** command to check the PNNI connectivity within each peer group. For example, switch SWA3 sees links to the other two members of its peer group:

```
SWA3 # show pnnilink
Num(ALL)                                :

Num  Port     Node   Remote Node     Hello State    Link Type
     Number   Index  IP Addr
=============================================================================
1    7A2          1  206.61.237.20   2WayInside     Lowest Level Horizontal Link
2    7A3          1  206.61.237.19   2WayInside     Lowest Level Horizontal Link

SWA3 #
```

4. Select switch SWA3 to be the PGL of group A and switch SWB3 to be the PGL of group B.

5. Use the **add pnninode** command to add a second, higher-level, node to switch SWA3:

```
SWA3 # add pnninode
NodeIndex(2)                            :      — Specifies node number 2
NodeLevel(72)                           :      — 72 is above the group A's level of 80
ComplexRepresentation(N)                :

SWA3 #
```

Do the same for switch SWB3:

```
SWB3 # add pnninode
NodeIndex(2)                            :      — Specifies node number 2
NodeLevel(72)                           :      — 72 is above the group B's level of 80
ComplexRepresentation(N)                :

SWB3 #
```

6. Use the **set pnnipglelection** command to set SWA3 and SWB3's leadership priority so that they are elected as PGLs within their respective peer groups:

```
SWA3 # set pnnipglelection
NodeIndex(1)                            :
LeadershipPriority(0)                   : 205     — Highest priority in election process
ParentNodeIndex(0)                      : 2       — Node 2 will represent the peer group A in the parent group
InitTime(15)                            :
OverrideDelay(30)                       :
ReElectTime(15)                         :

SWA3 #
```

Do the same on switch SWB3:

```
SWB3 # set pnnipglelection
NodeIndex(1)                              :
LeadershipPriority(0)                     : 205     — Highest priority in election process
ParentNodeIndex(0)                        : 2       — Node 2 will represent the peer group B in the parent group
InitTime(15)                              :
OverrideDelay(30)                         :
ReElectTime(15)                           :

SWB3 #
```

7. Use the **show pnnipglelection** command to verify that switches SWA3 and SWB3 have become the PGLs of their respective peer groups. For example, on switch SWA3, enter the following:

```
SWA3 # show pnnipglelection
NodeIndex(1)                              :

PGL Election Information
================================================================================
Node Index            : 1
Leadership Priority   : 205
Parent Node Index     : 2
Init Time             : 15 secs
Override Delay        : 30 secs
Reelect Time          : 15 secs
Time Stamp            : 228588
Election State        : Operating as PGL      — Switch SWA3 has become PGL of group A
Preferred PGL         : 50:a0:39:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:28
:c1:ff:00
Peer Group Leader     : 50:a0:39:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:28
:c1:ff:00
Active Parent Node Id : 48:50:39:00:00:00:00:00:00:00:00:00:00:01:00:20:d4:28
:c1:ff:00

SWA3 #
```

8. Physically connect switch SWA3 to SWB3 to establish connectivity between peer groups A and B.

9. Use the **show pnnilink** command to check the connectivity between the peer groups. In the following example, **show pnnilink** is entered on switch SWA3 and shows a link to switch SWB3 (SWB3's IP address is 206.61.237.23):

```
SWA3 # show pnnilink
Num(ALL)                                  :

Num  Port     Node   Remote Node     Hello State    Link Type
     Number   Index  IP Addr
================================================================================
1   7A1         1  206.61.237.20   2WayInside     Lowest Level Horizontal Link
2   7A3         1  206.61.237.19   2WayInside     Lowest Level Horizontal Link
3   7B1        1  206.61.237.23   CommonOut     Outside and Uplink          — Physical link to switch SWB3
4   --          2  N/A          2WayInside    Horizontal Link to/from LGN  — Logical link between switches

SWA3 #
```

Note Notice that the IP address entry for the logical link between the LGNs is N/A (Not Applicable). Logical entities do not have IP addresses.

Connectivity is now established between the two peer groups. For example, if LANE services are running on a switch within peer group A, LANE clients can exist in group B. The clients in group B will traverse the link between the two groups, find the LANE server in group A, and join the ELAN. Figure 3-2 shows a logical representation of the topology created in the example.

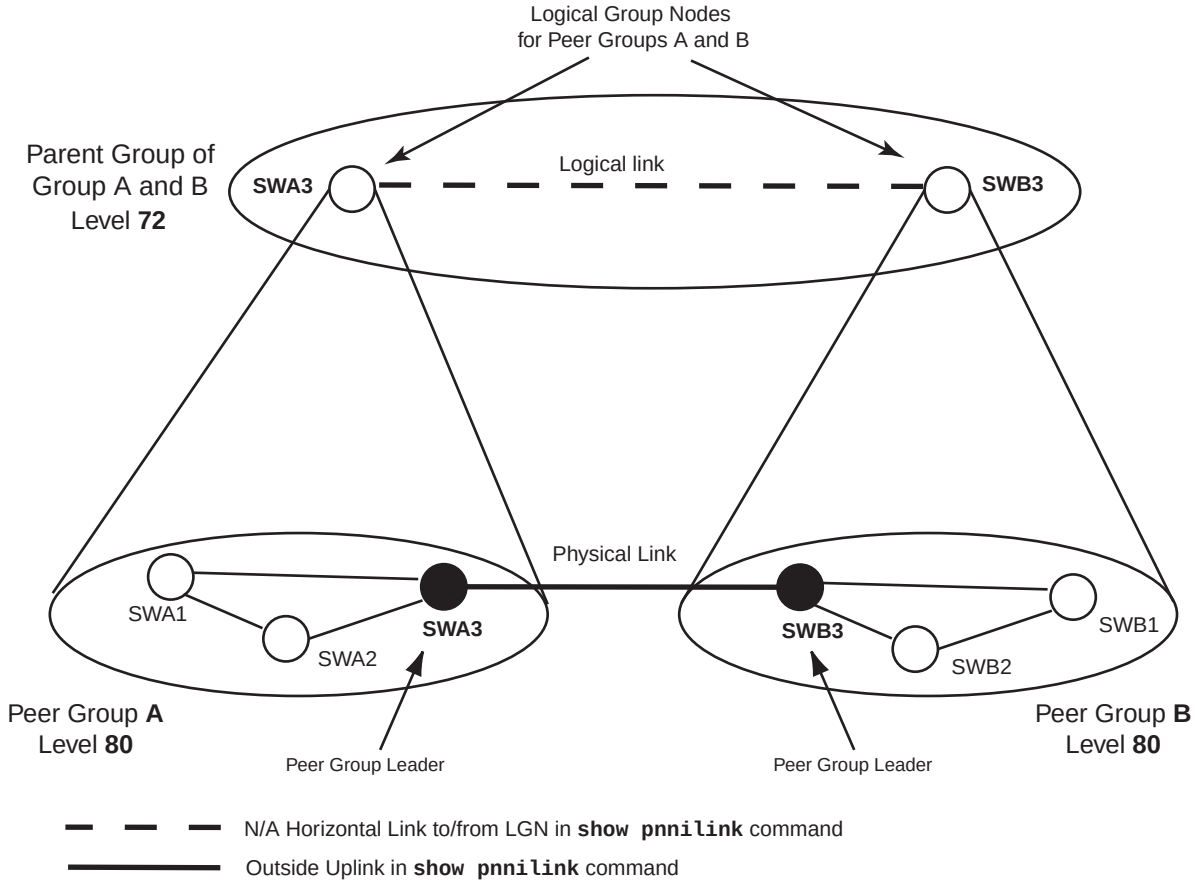

Figure 3-2 Logical representation of connectivity between groups A and B

3.2.2 Physical Connections Between Peer Groups

Keep in mind that the two PGL switches (switches SWA3 and SWB3) do not have to be directly connected to each other for the two peer groups to maintain connectivity. PGLs can find each other through any physical link that connects the two groups. For example, if a second physical link is made between two other switches in groups A and B (for instance, between SWA1 and SWB2), and if the physical link between the PGLs is removed, the PGLs will reestablish their connectivity across the second physical link.

Adding Higher-level Peer Groups

Adapting the process in the example above, more sophisticated PNNI topologies can be created. For example, to establish connectivity with other parent groups at level 72, do the following:

1. Make a physical connection between any two switches represented in the separate parent groups.

2. Add a third node (at level 64) to either switch SWA3 or SWB3.
3. Use the **`set pnnipglelection`** command to designate the switch's second node (*not third*) as the PGL for the parent peer group, and specify the third node as the parent node of the second.
4. Perform steps 2 and 3 for switches with the same role in the other level 72 parent groups.

These steps create a grandparent group at level 64, and establishes a virtual link between the LGNs that represent the LGNs at level 72 (see Figure 3-3).

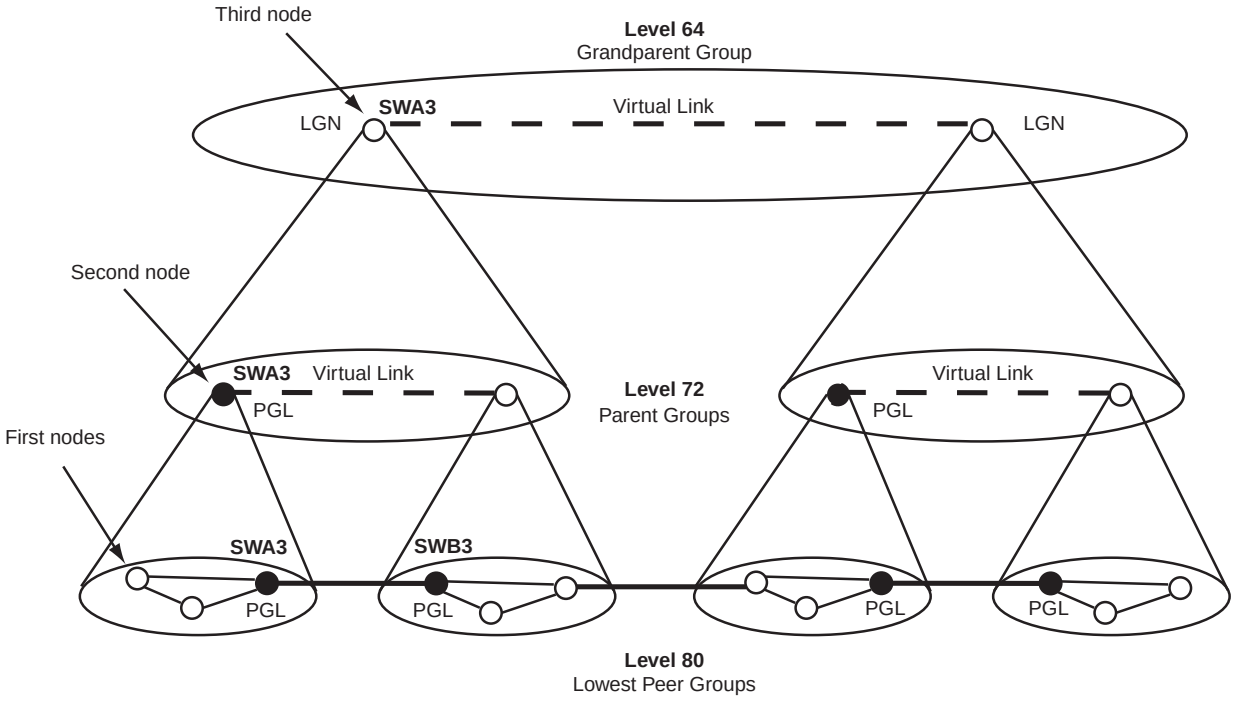

Figure 3-3 Adding a third PNNI node for next level connectivity

3.3 MANAGING PARALLEL PNNI LINKS

ATM SmartSwitches can be connected by more than one physical link. PNNI treats these connections as parallel physical links. By default, parallel links are considered to have equal capabilities with regard to call set ups.

For example, if a second link is added between switch SWA3 and switch SWB3 (from the example above), this parallel link can be seen using the **`show pnnilink`** command.

```
SWA3 # show pnnilink
Num(ALL)                                :

Num  Port     Node   Remote Node     Hello State    Link Type
     Number   Index  IP Addr
==========================================================================
1    7A1          1  206.61.237.20   2WayInside     Lowest Level Horizontal Link
2    7A3          1  206.61.237.19   2WayInside     Lowest Level Horizontal Link
3    7B1          1  206.61.237.23   CommonOut      Outside and Uplink
4   7B2         1  206.61.237.23   CommonOut     Outside and Uplink          — Second physical link to B3
5    --           2  N/A             2WayInside     Horizontal Link to/from LGN
6    --          2  N/A           2WayInside    Horizontal Link to/from LGN  — Second logical link to B3

SWA3 #
```

You can adjust the advertised capabilities of each link (on a per-port, per-service class basis) by changing the link's administrative weights. Use the **`show pnniinterface`** command to view the current administrative weights. For example:

```
SmartSwitch # show pnniinterface
PortNumber(ALL)                         :

Port       Admin Wt    Admin Wt    Admin Wt    Admin Wt    Admin Wt  Aggregation
Number       CBR         RTVBR       NRTVBR       ABR          UBR       Token
================================================================================
CPU            5040        5040        5040        5040        5040           0
CPU.1          5040        5040        5040        5040        5040           0
7A1            5040        5040        5040        5040        5040           0
7A2            5040        5040        5040        5040        5040           0
7A3            5040        5040        5040        5040        5040           0
7A4            5040        5040        5040        5040        5040           0
7B1            5040        5040        5040        5040        5040           1
7B2            5040        5040        5040        5040        5040           0
7B3            5040        5040        5040        5040        5040           0

SmartSwitch #
```

A link's administrative weight defines its desirability to the PNNI routing service when setting up a call of a particular class of service. The lower the numerical value of the administrative weight, the more desirable the route. For example, a route with administrative weight 200 for a particular class of service is considered a better route than one with the default weight of 5040 for that service. As a result, the administrative weight provides a quantitative way to control which routes are favored for call set up with regard to service class.

The ability to control the PNNI routing service in this fashion allows for parallel routes to be weighted such that one link is designated as the favored for a particular service class, while a parallel link can be designated as the favored route for a different service class.

Use the **set pnniinterface** command to set the administrative weight of a physical link originating from a particular port. The following, is an example of increasing the administrative weight for CBR call setups through the physical link on port 7a1:

```
SmartSwitch # set pnniinterface
PortNumber()                          : 7a1   — Link on port 7a1
AdminWtCBR(5040)                      : 100   — Set the weight for CBR connections higher on this link
AdminWtRTVBR(5040)                    :
AdminWtNRTVBR(5040)                   :
AdminWtABR(5040)                      :
AdminWtUBR(5040)                      :
AggregationToken(0)                   :
RccServCategory(NRTVBR)               :
RccServCategory(NRTVBR)               :

SmartSwitch #
```

3.3.1 Aggregation Tokens

An aggregation token is associated with each physical PNNI link. The value of the token determines how a physical link is advertised to the rest of the network. By default, all physical links (even parallel links) use an aggregation token of zero (0). When physical PNNI links have the same token value, the links are represented as a single logical link within the parent peer group. For example, no matter how many physical links connect peer groups A and B, they are represented within the parent group as a single logical link. Using different token values for physical links causes the links to be represented (and advertised) as separate logical links within the parent group.

Continuing with the earlier example of multi-level topologies, add a second physical PNNI link between peer groups A and B by physically connecting switch SWA2 to switch SWB2. By setting the aggregation token of this physical link to a value different from the physical link connecting switches SWA3 and SWB3, a second logical link appears within the parent group.

For example, the physical link between SWA3 and SWB3 has an aggregation token value of zero (0). Use the **set pnniinterface** command to change the value of the aggregation token for the physical link between SWA2 and SWB2 to one (1):

```
SWA2 # set pnniinterface
PortNumber()                          : 7b2  — Link on switch SWA2 comes from this port
AdminWtCBR(5040)                      :
AdminWtRTVBR(5040)                    :
AdminWtNRTVBR(5040)                   :
AdminWtABR(5040)                      :
AdminWtUBR(5040)                      :
AggregationToken(0)                   : 1   — Change the value of the aggregation token from the default
RccServCategory(NRTVBR)               :

SWA2 #
```

Perform the same operation on switch SWB2 in group B:

```
SWB2 # set pnniinterface
PortNumber()                          : 4a3  — Link on switch SWB2 comes from this port
AdminWtCBR(5040)                      :
AdminWtRTVBR(5040)                    :
AdminWtNRTVBR(5040)                   :
AdminWtABR(5040)                      :
AdminWtUBR(5040)                      :
AggregationToken(0)                   : 1   — Change the value of the aggregation token from the default
RccServCategory(NRTVBR)               :

SWB2 #
```

The physical connection from switch SWA2 to switch SWB2 is now advertised as a second logical link within the parent peer group (see Figure 3-4).

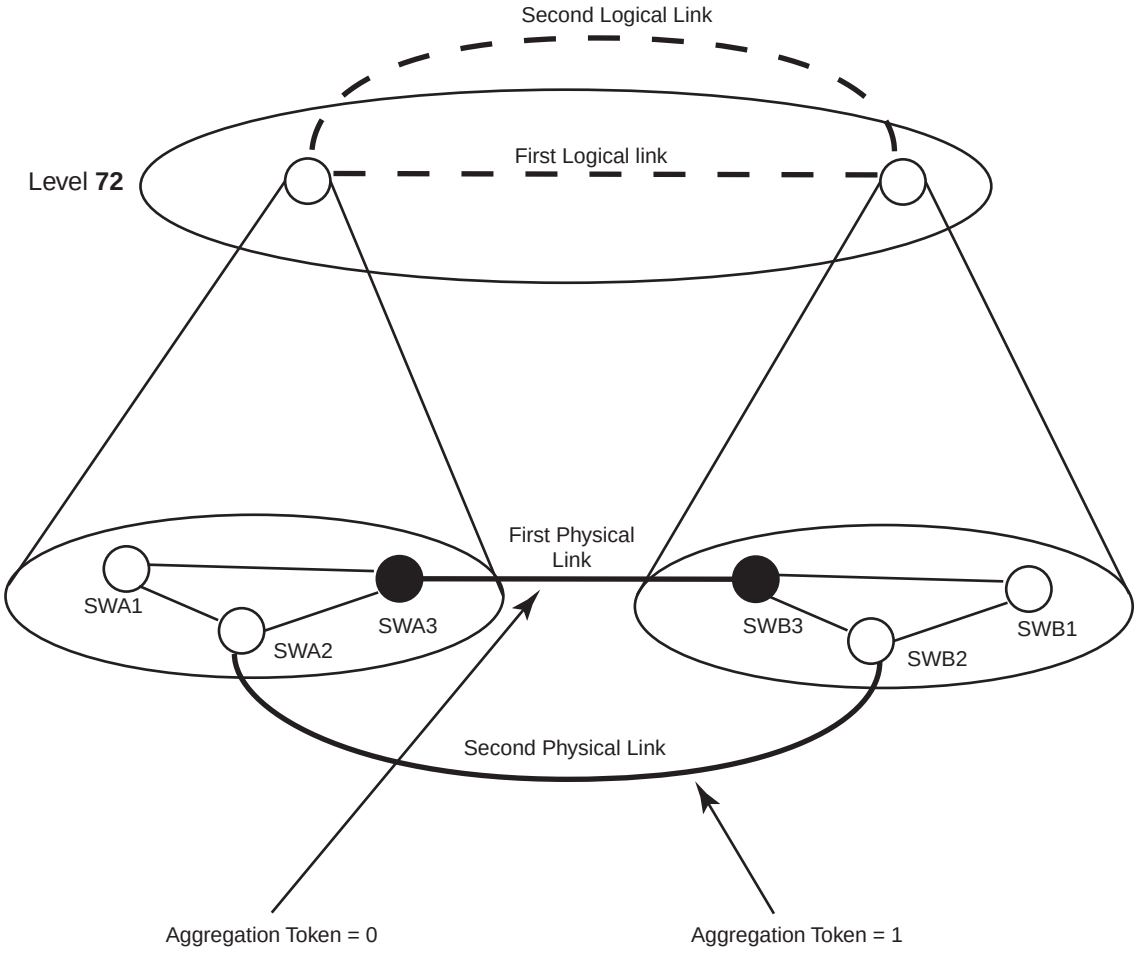

Figure 3-4 Aggregation token values and parallel links

3.3.2 PNNI Link Timing

By default, if a PNNI link loses connectivity, the link fails after three (3) seconds. This short amount of time is designed as a buffer in case of minor latency. By waiting three seconds before releasing resources and tearing down the connection, a minor latency occurrence (less than three seconds) will not bring the link down, and will keep the PNNI network from going through the process of reconfiguration.

Note Link failure is determined either by hardware, when a *"loss of frame"* is detected; or by the signaling software, when the QSAAL link goes down.

However, certain time-sensitive implementations of PNNI may require that link fail occur either immediately or after a period of time longer than three seconds. Use the **`set linkmonitortimeout`** command to control the time required for the SmartSwitch ATM switch to assume a link has failed.

For example, two SmartSwitch ATM switches are connected with parallel PNNI links. To configure the switches to immediately recognize any lapse in traffic as a downed link, enter the following on both switches:

```
SmartSwitch # set linkmonitortimeout
TimeoutValue(3)                          : 0   — Make the timeout instantaneous

SmartSwitch #
```

If a traffic lapse occurs on one of the links, that link's port immediately frees up all resource, and all traffic is routed between the switches through the remaining link.

Notice that the **`set linkmonitortimeout`** command controls the **`TimeoutValue`** on a switch-wide basis (not a per-port basis).

Caution Remember that while some special uses of PNNI may require the **`TimeoutValue`** to be zero (0), setting the **`TimeoutValue`** to less than three seconds may cause the PNNI network to "bounce," entering a state of constant (and unnecessary) reconfiguration. For this reason, care should be taken when setting the **`TimeoutValue`** to less than three seconds.

4 ROUTING

4.1 ADDITIONAL ROUTING PROTOCOLS

Along with PNNI, all ATM SmartSwitches support additional ATM routing protocols:

- IISP — Use to connect with devices that do not support PNNI
- UNI — Use to connect end stations (also to connect devices whose implementation of ILMI is incompatible with the ATM SmartSwitch)

Note Both IISP and UNI routes are created and modified using the **ATMRoute** command. The proper route type is determined by the ATM SmartSwitch through interface signaling information.

4.2 IISP ROUTES

Use the **add atmroute** command to create an IISP route that links the ATM SmartSwitch to a device that supports only IISP routing. For example,

1. Physically connect port **5b2** of the SmartSwitch 6500 to the IISP device.
2. Enter **show netprefix** to determine the netprefix of port **5b2** on the SmartSwitch 6500:

```
SmartSwitch # show netprefix 5b2
Port      NetPrefix
================================================================================
5B2        39:00:00:00:00:00:00:00:00:00:14:41:80
SmartSwitch #
```

3. Determine the address of the IISP device. (For this example, this could be a port address, we use **52:00:00:00:00:00:00:00:00:00:14:51:80**)
4. Enter the **add atmroute** command to create a static route to the IISP device:

```
SmartSwitch # add atmroute
PortNumber()                          : 5b2
AtmAddress()                          : 52:00:00:00:00:00:00:00:00:00:14:51:80
PrefixLength(104)                     :
Index(0)                              :
Type(Internal)                        :exterior     — This is an exterior route
Scope(0)                              :
MetricsTag(0)                         : — See Section 4.4 for information on metrics
Advertising(NO)                       : — Do not advertise this address into the PNNI domain
SmartSwitch #
```

Note For IISP routes, always set the **Type** parameter of the **add atmroute** command to **external**. This indicates that the route is external to the PNNI domain.

Note The **add atmroute** command allows you to specify a set of metrics to be used with the route. Metrics are created using the **add pnnimetric** command, and are assigned to routes by metric tag numbers. By setting the appropriate administrative weights within metrics, it's possible to create parallel load-sharing or fail-over routes. For more information about metrics, administrative weights, and metric tags, see Section 4.4, Route Metrics.

5. Enter the **show atmroute** command to determine whether the route was created:

```
SmartSwitch # show atmroute
AddressNumber(ALL)                        :
No. Port Route Address                                                  Type Protocol
================================================================================
1   7B4  39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:d4:14:41:80   I   MGMT
2   7B4  39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:d4:14:41:81   I   MGMT
3    --  39:00:00:00:00:00:00:00:00:00:14:59:00                     I   PNNI
4    --  39:00:00:00:00:00:00:00:00:00:28:e9:80                     I   PNNI
5    --  39:00:00:00:00:00:00:00:00:00:28:f5:00                     I   PNNI
6   7B4  47:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01   I   MGMT
7   5B2  52:00:00:00:00:00:00:00:00:00:14:51:80                     I   MGMT   — This is our route
SmartSwitch #
```

The route to the IISP device appears as **Route 7**, and with Protocol Type of **MGMT** (management).

6. Create a route on the IISP device that refers to the netprefix (`39:00:00:00:00:00:00:00:00:00:14:41:80`) of port **5b2** on the SmartSwitch 6500.

Note For IISP routes to work with certain devices, ILMI may need to be disabled on the ATM SmartSwitch. Use the **set portconfig** command to disable ILMI on the ATM SmartSwitch on a per-port basis.

4.2.1 IISP Routing Considerations

When creating routes between an ATM SmartSwitch (running PNNI) and IISP devices, the criteria that characterize IISP connectivity still apply. To reach an ATM SmartSwitch within the PNNI domain, the IISP device must have a configured route that points directly to a port on the target ATM SmartSwitch. Conversely, there must be an ATM SmartSwitch that has a direct physical link (and a route over that link) to the IISP device. The following two examples illustrate this point.

IISP Routing Example One

In Figure 4-1 Switch A is an IISP device connected to the PNNI domain through Switch B. Switch A contains an LEC, which is a member of an ELAN whose LECS is on Switch C (within the PNNI domain). If the LEC on Switch A is to make contact with the LECS on Switch C, Switch A must contain an IISP route directly to switch C. Furthermore, Switch B must contain a route to switch A over the physical link that connects the two switches.

Note Dotted lines in the diagrams below represent one-way IISP routes to the devices pointed to by the arrowheads. Each route is defined on the device from which the dotted line originates.

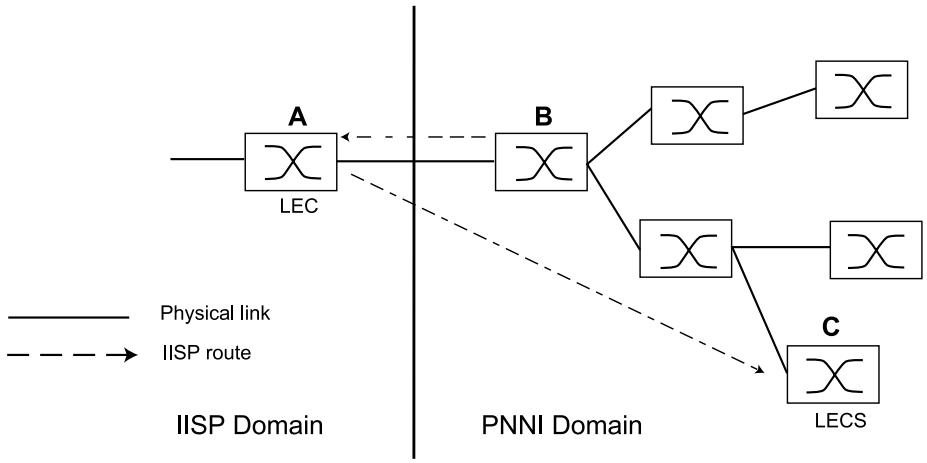

Figure 4-1 IISP route across PNNI domain

IISP Routing Example Two

A second IISP device (Switch D) is added behind Switch A. If Switch D also needs to reach Switch C for LANE support, additional IISP routes must be defined between Switches D and C, B and D, and A and D. Figure 4-2 shows the typical “route to every point reached” IISP topology.

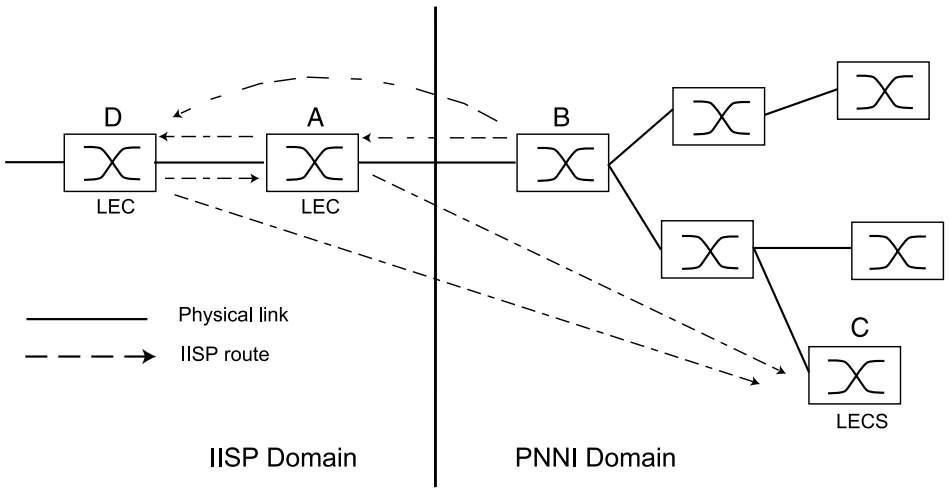

Figure 4-2 Routes needed for a second IISP switch

4.2.2 IISP Link Timing

By default, if an IISP link loses connectivity, the link fails after three (3) seconds. This short amount of time is designed as a buffer in case of minor latency. By waiting three seconds before releasing resources and tearing down the connection, a minor latency occurrence (less than three seconds) will not bring down the route.

However, certain time-sensitive implementations may require that link fail occurs either immediately or after a longer period of time than three seconds. Use the **`set linkmonitortimeout`** command to control the time required for the SmartSwitch ATM switch to assume an IISP route has failed.

For example, two SmartSwitch ATM switches are connected with parallel IISP links. To configure the switches to immediately recognize any lapse in traffic as a downed link, enter the following on both switches:

```
SmartSwitch # set linkmonitortimeout
TimeoutValue(3)                          : 0   — Make the timeout instantaneous

SmartSwitch #
```

If a traffic lapse occurs on one of the IISP links, that link's port immediately frees up all resources, and all traffic between the switches is routed through the remaining IISP link.

Notice that the **`set linkmonitortimeout`** command controls the **`TimeoutValue`** on a switch-wide basis (not a per-port basis).

Caution Remember that while some special network configurations may require the **`TimeoutValue`** to be zero (0), setting **`TimeoutValue`** to less than three seconds may cause an IISP route to fail unnecessarily. For this reason, care should be taken when setting the **`TimeoutValue`** to less than three seconds.

4.3 UNI ROUTES

Use the **add atmroute** command to create UNI routes. For example, connect an end station adapter (with MAC address 00:11:22:33:44:55) to port **7A2** of a SmartSwitch 6500. If the adapter does not support ILMI or its ILMI is incompatible with the SmartSwitch 6500, you must create a static UNI route between the adapter and port **7A2** of the SmartSwitch 6500.

The following example works with any ATM SmartSwitch, however, the port numbering may be different (for instance **A2** instead of **7A2**):

1. Enter the **show netprefix** command to obtain the netprefix of port 7A2:

```
SmartSwitch # show netprefix
PortNumber(ALL)                          : 7a2
 Port#   NetPrefix
================================================================================
7A2       39:00:00:00:00:00:00:00:00:00:14:59:00
SmartSwitch #
```

2. Reconfigure the adapter with an ATM address made from the netprefix of port 7A2 and the adapter's MAC address: 39:00:00:00:00:00:00:00:00:00:14:59:00:00:11:22:33:44:55:00.

3. Use the **add atmroute** command to create a static UNI route that specifies port 7A2 and the adapter's new ATM address.

```
SmartSwitch # add atmroute
PortNumber()                           : 7a2
AtmAddress()                         : 39:00:00:00:00:00:00:00:00:00:14:59:00:00:11:22:33:44:55:00
PrefixLength(152)                        :
Index(0)                                 :
Type(Internal)                           :      — Take the default to make this an "internal" route
Scope(0)                                 :
MetricsTag(0)                            :      — See Section 4.4 for information on metrics
Advertising(NO)                          :yes   — Advertise this address into the PNNI domain
SmartSwitch #
```

Note Always set the **Type** parameter of the **add atmroute** command to **internal** (the default) for UNI routes. This indicates that the route is internal to the PNNI domain.

Note The **add atmroute** command allows you to specify a set of metrics to be used with the route. Metrics are created using the **add pnnimetric** command, and are assigned to routes by metric tag numbers. By setting the appropriate administrative weights within metrics, it's possible to create parallel load-sharing or fail-over routes. For more information about metrics, administrative weights, and metric tags, see Section 4.4, Route Metrics.

4. Enter the **`show atmroute`** command to check that the UNI route was added.

```
SmartSwitch # show atmroute
AddressNumber(ALL)                        :
No. Port Route Address                                                  Type Protocol
=============================================================================
1   7B4  39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:d4:14:41:80   I   MGMT
2   7B4  39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:d4:14:41:81   I   MGMT
3    --  39:00:00:00:00:00:00:00:00:00:14:59:00                     I   PNNI
4   7A2  39:00:00:00:00:00:00:00:00:00:14:59:00:00:11:22:33:44:55   I   MGMT  — Our added UNI route
5    --  39:00:00:00:00:00:00:00:00:00:28:e9:80                     I   PNNI
6    --  39:00:00:00:00:00:00:00:00:00:28:f5:00                     I   PNNI
7   7B4  47:00:79:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01   I   MGMT
8   5B2  52:00:00:00:00:00:00:00:00:00:14:51:80                     I   MGMT
SmartSwitch #
```

The UNI route appears in the table as **`Route 4`**, with Protocol Type of **`MGMT`** (management).

Note For UNI routes to work with certain devices, ILMI may also need to be disabled on the ATM SmartSwitch. Use the **`set portconfig`** command to disable ILMI on the ATM SmartSwitch on a per-port basis.

4.3.1 UNI Link Timing

By default, if a UNI link loses connectivity, the link fails after three (3) seconds. This short amount of time is designed as a buffer in case of minor latency. By waiting three seconds before releasing resources and tearing down the connection, a minor latency occurrence (less than three seconds) will not bring down the route.

However, certain time-sensitive implementations may require that link fail occurs either immediately or after a longer period of time than three seconds. Use the **`set linkmonitortimeout`** command to control the time required for the SmartSwitch ATM switch to assume a UNI route has failed.

For example, a SmartSwitch ATM switch is connected to two UNI uplinks (one active, one standby) through two separate ports. One switch port is connected to the active UNI uplink and the other switch port is connected to the standby UNI uplink. To configure the switch to immediately recognize any lapse in traffic on the active UNI uplink port as a downed link, enter the following on the SmartSwitch ATM switch:

```
SmartSwitch # set linkmonitortimeout
TimeoutValue(3)                           : 0   — Make the timeout instantaneous

SmartSwitch #
```

If the active UNI uplink fails-over to the standby UNI uplink, the SmartSwitch ATM switch port connected to the failed active uplink immediately frees up all resources, and begins accepting traffic on the port connected to the standby UNI uplink.

Notice that the **`set linkmonitortimeout`** command controls the **`TimeoutValue`** on a switch-wide basis (not a per-port basis).

Caution Remember that while some special network configurations may require the **`TimeoutValue`** to be zero (0), setting **`TimeoutValue`** to less than three seconds may cause a UNI route to fail unnecessarily. For this reason, care should be taken when setting the **`TimeoutValue`** to less than three seconds.

4.4 ROUTE METRICS

Route metrics are assigned to routes using a metric tag (one of the input parameters for **`add atmroute`**). The metric tag specifies a particular pair of incoming and outgoing metrics contained within a list of metrics. Metrics are created using the **`add pnnimetric`** command (whether PNNI, IISP, or UNI routes). Each metric pair specifies a set of values that describe a route's Service Category, cell rates, bandwidth, and administrative weight. Locally, metric values determine the behavior of the link. Within PNNI networks, PNNI's Generic Call Admission Control (GCAC) assesses metrics when establishing calls.

4.4.1 Administrative Weights

The administrative weight (**`AdminWt`** parameter) of a metric allows you to control the use of a route for call set ups. By default, a metric assigns the lowest value (5040) to the **`AdminWt`** parameter. Values less than 5040 (for example 500) are considered to have greater administrative weight. Among parallel routes, the route with the greatest administrative weight is seen as the preferred route; subsequently, most calls are set up through that route. Other parallel routes with lower administrative weights are used as *"backup"* routes These backup routes will be used only if the route with the greatest administrative weight is either out of bandwidth or down.

4.4.2 Creating Route Metrics

The following section describes how to create a route metric and assign it to a route.

Note For a complete description of all **`pnnimetric`** parameters, see the SmartSwitch ATM Switch Reference Manual.

In the following example, a metric pair is created (with metric tag of 9), which specifies CBR as the Service Category, administrative weight of 200, Max Cell Rate of 1000 cells per second, and an Available Cell Rate of 750 cells per second.

Note The default value **`NotUsed`** that appears in the **`add pnnimetric`** command means "If no value is specified for the parameter, the parameter is not used within the metric." It does NOT mean that the parameter does not accept values.

1. Create the outgoing member of the metric pair:

```
SmartSwitch # add pnnimetric
Executing this command : add PnniMetrics
MetricsTag(1)                              : 9
TrafficDirection(Outgoing)                 :    — 1st pair member, we accept the default (Outgoing)
ServiceCategory(UBR)                       : cbr
GCAC_CLP(2)                                :
AdminWt(5040)                              : 200
MaxCellRate(NotUsed)                       : 1000
AvailableCellRate(NotUsed)                 : 750
MaximumCellTransferDelay(NotUsed)          :
CellDelayVariation(NotUsed)                :
CellLossRatioForCLP=0(NotUsed)             :
CellLossRatioForCLP=0+1(NotUsed)           :
CellRateMargin(NotUsed)                    :
VarianceFactor(NotUsed)                    :

SmartSwitch #
```

2. Create the incoming member of the metric pair:

```
SmartSwitch # add pnnimetric
Executing this command : add PnniMetrics
MetricsTag(1)                              : 9
TrafficDirection(Outgoing)                 : incoming   — 2nd pair member, we set as incoming
ServiceCategory(UBR)                       : cbr
GCAC_CLP(2)                                :
AdminWt(5040)                              : 200
MaxCellRate(NotUsed)                       : 1000
AvailableCellRate(NotUsed)                 : 750
MaximumCellTransferDelay(NotUsed)          :
CellDelayVariation(NotUsed)                :
CellLossRatioForCLP=0(NotUsed)             :
CellLossRatioForCLP=0+1(NotUsed)           :
CellRateMargin(NotUsed)                    :
VarianceFactor(NotUsed)                    :

SmartSwitch #
```

3. Enter **`show pnnimetric`** to view the newly created metric pair:

```
SmartSwitch # show pnnimetrics
Metrics(ALL)                               :

Metrics Metrics Tag  Direction Index      GCAC CLP  Admin Wt Service Categories
================================================================================
1       0x9          Incoming  0x10       CLP0+1        200   CBR  — Incoming pair member
2       0x9          Outgoing  0x10       CLP0+1        200   CBR  — Outgoing pair member
3       0x111113     Outgoing  0x1        CLP0+1       5040   UBR
4       0x111113     Outgoing  0x2        CLP0+1       5040   ABR
5       0x111113     Outgoing  0x4        CLP0         5040   NRTVBR
6       0x111113     Outgoing  0x18       CLP0         5040   CBR RTVBR
7       0x111114     Outgoing  0x1        CLP0+1       5040   UBR
8       0x111114     Outgoing  0x2        CLP0+1       5040   ABR
9       0x111114     Outgoing  0x4        CLP0         5040   NRTVBR
10      0x111114     Outgoing  0x18       CLP0         5040   CBR RTVBR

SmartSwitch #
```

The newly created metric pair appears at the top of the list as metrics 1 and 2.

Once the metric is created, we can specify its metric tag number within the definition of a route. In this example, an IISP route is being created:

```
SmartSwitch # add atmroute
PortNumber()                          : 6b2
AtmAddress()                          : 39:00:00:00:00:00:00:00:00:00:55:77:88
PrefixLength(104)                     :
Index(0)                              :
Type(Internal)                        :exterior
Scope(0)                              :
MetricsTag(0)                         : 9    — The index tag of our metric pair
Advertising(NO)                       :
SmartSwitch #
```

4.5 IP ROUTING FOR MANAGEMENT

ATM SmartSwitches provide limited IP routing. IP routing allows switches that are not connected directly to Ethernet to communicate with an Ethernet-based network management system (NMS). The connection is made by adding IP routes on the non-connected switches that specify a client on a connected switch as their gateway to the Ethernet.

Note ATM SmartSwitch IP routing performance is inadequate for routing between VLANs. If you need to create routes between VLANs on your ATM SmartSwitch, use a router equipped with an ATM interface. Consult Cabletron Customer Support for recommended routers.

For example,

- Switch SW1 and the NMS are on an Ethernet network with address 128.205.99.0.
- The IP address of SW1's Ethernet port is 128.205.99.254.
- The IP address of SW1's LANE client is 90.1.1.254.
- The IP address of SW2's LANE client is 90.1.1.33.
- SW2 is not physically connected to the Ethernet network.
- SW2 is connected to SW1 through PNNI, and both switches are part of the same emulated LAN.

To reach SW2 with the Ethernet-based NMS, create an IP route that assigns SW1's switch client as SW2's default gateway to the network 128.205.99.0. Enter the following on SW2 (see Figure 4-3):

```
SmartSwitch # add route
DestNetIP() : 128.205.99.0    — address of the Ethernet network to reach
GatewayIP() : 90.1.1.254      — IP address of SW1's LANE client
SmartSwitch #
```

Switch SW2 can now communicate with the NMS on the Ethernet network.

To see the route, enter the **show route** command on SW2

```
SmartSwitch # show route
ROUTE NET TABLE
destination      gateway          flags  Refcnt  Use           Interface
-------------------------------------------------------------------------
0.0.0.0          0.0.0.0          1      0       0             zn0
90.1.1.0         90.1.1.33        1      0       1688          zn1
128.205.99.0     90.1.1.254       1      3       5660          ei0
-------------------------------------------------------------------------
ROUTE HOST TABLE
destination      gateway          flags  Refcnt  Use           Interface
-------------------------------------------------------------------------
127.0.0.1        127.0.0.1        5      0       0             lo0
-------------------------------------------------------------------------
SmartSwitch #
```

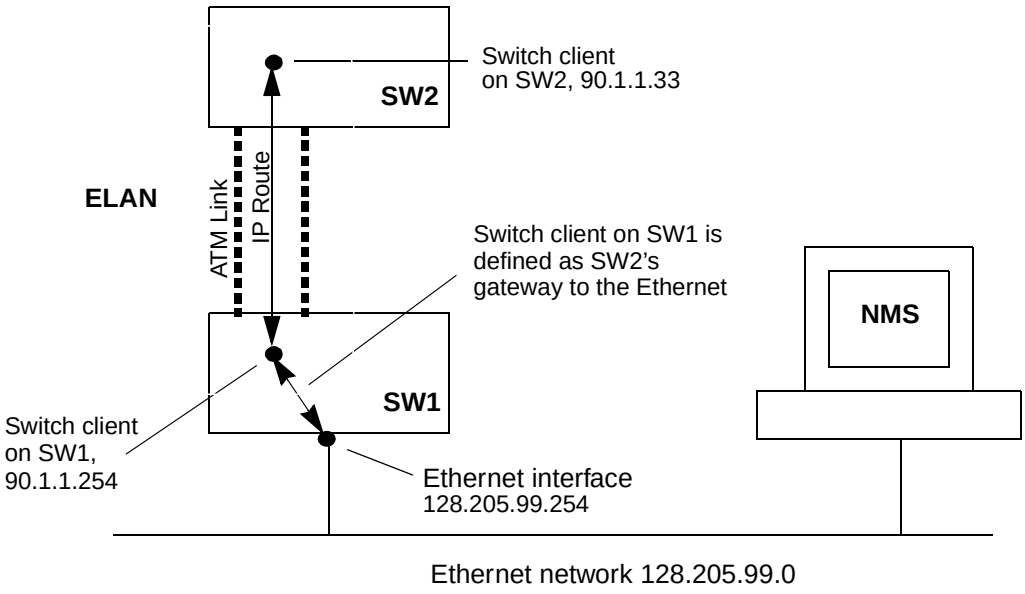

Figure 4-3 IP routing through SW1 for connectivity to the Ethernet network

Note The NMS must also contain a route that specifies the Ethernet interface of the Ethernet connected switch as the gateway to the ELAN subnet.

5 VIRTUAL PORTS AND STATIC CONNECTIONS

5.1 PVC CONNECTIONS

ATM SmartSwitches support Permanent Virtual Circuits (PVCs), both point-to-point and point-to-multipoint. Use PVCs to connect devices (that do not support SVCs) to a switch's local client. Also, use PVCs to make connections through an ATM SmartSwitch between devices that support only PVCs.

Use point-to-point PVCs to connect one end node to another for two-way communication. Use point-to-multipoint PVCs to connect a broadcast end node to a group of receiving end nodes; traffic is one way.

Note The examples in this chapter are carried out on a SmartSwitch 6500. Most of these examples will work with all other SmartSwitch ATM switches, however, the port numbering would be different. For example, instead of port **`7A1`** (SmartSwitch 6500) the port might be **`A1`** (on a 2500, 6A000, or 9A100).

Note PVCs use traffic descriptors to define their traffic characteristics. See Chapter 6, "Traffic Management," Section 6.1.1 for further information on traffic descriptors.

5.1.1 Point-to-Point PVCs

The procedure for setting up a PVC connection between two end nodes through an ATM SmartSwitch consists of specifying the ports and the Virtual Path Connection Identifier and Virtual Channel Identifiers (VPCI and VCI).

1. Use **`add trafficdescriptor`** to define a traffic descriptor to use with the PVC:

```
SmartSwitch # add trafficdescriptor
Executing this command : add TrafficDescriptor
TrafficType(UBR)                          : cbr
TrafficDescriptorType(2)                  :
PCRCLP01(100)                             :
QOSCLASS(1)                               :
AalType(5)                                :

SmartSwitch #
```

For this example, we specify CBR as the traffic type, then take the remaining defaults. Enter the **show trafficdescriptor** command to obtain the index number of the new traffic descriptor. In this example, the index number is two (2).

```
SmartSwitch # show trafficdescriptor
================================================================================

TD#  Traff    Desc  QoS  Peak Cell Rate  Sust Cell Rate  Max Burst Size  Min Cell Aal Type
     Type     Type         (Kb/s)           (Kb/s)          (Kb/s)      Rate
                         CLP_0  CLP_0+1  CLP_0  CLP_0+1  CLP_0  CLP_0+1  (Kb/s)
================================================================================
1    NRTVBR   7     0    0      10872    5436   0        2052   0        0         5
2    CBR      2     1    0      100      0      0        0      0        0         5
176  NRTVBR   2     1    0      1585     0      0        0      0        0         5

SmartSwitch #
```

2. Use **add pvc** to create the PVC; specify the ports through which the connection is established, the VPI/VCI pair to use with each port, and the traffic descriptor to use.

```
SmartSwitch #  add pvc
ConnType(PTP)                                 :
Port-1-Number()                               : 7a1   — Specify first port
Port-1-VPCI()                                 : 0     — Specify its VPCI
Port-1-VCI()                                  : 100   — Specify its VCI
Port-2-Number()                               : 7b2   — Specify second port
Port-2-VPCI()                                 : 0     — Specify its VPCI
Port-2-VCI()                                  : 100   — Specify its VCI
Port1-to-Port2TrafficDescriptorIndex()        : 2     — We use our traffic descriptor
Port2-to-Port1TrafficDescriptorIndex()        : 2

SmartSwitch #
```

The example above creates a PVC between ports 7a1 and 7b2 with VPCI/VCI = 0/100.

3. Plug the end nodes into the specified ATM SmartSwitch ports (7a1 and 7b2).

4. Configure each end node with the proper IP address, subnet mask, and VPCI/VCI pair = 0/100.

The end nodes can communicate with each other through the point-to-point PVC connection.

Note To create a PVC with a VPI greater than zero (0), you must change the default assignment of bits used to specify VPIs and VCIs. The number of VPI bits determine the available range of VPI numbers: *Largest VPI number* = $2^{VPIbits}$-1. For example, if the number of VPI bits is three, the highest VPI that can be specified is 2^3-1 = (8 - 1) = 7. To change the available VPI numbers, use the **set portconfig** command (on a per-port basis) to alter the **MaxVpiBits** parameter from its default of zero (0). Keep in mind that if VPI bits are increased VCI bits are accordingly decreased. Fewer VCI bits results in fewer available VCIs per VPI.

5.1.2 Point-to-Multipoint PVCs

Instructions in this section describe how to set up a point-to-multipoint connection through your ATM SmartSwitch.

Example: Create a point-to-multipoint connection between a broadcasting workstation on port **7a1** and three other workstations connected to ports **7a2**, **7a3**, and **7a4**.

1. Use **add trafficdescriptor** to create two new traffic descriptors, one for the forward direction, the other for the backward direction. For this example, for the forward traffic descriptor, we select UBR and accept the defaults.

```
SmartSwitch # add trafficdescriptor          — This is the forward descriptor
TrafficType(UBR)                          :  — We use UBR for this example
TrafficDescriptorType(11)                 :
PCRCLP01(100)                             :  — Take the default values
QOSCLASS(0)                               :
AalType(5)                                :

SmartSwitch #
```

However, on a point-to-multipoint connection there should be no traffic in the backward direction, so we define the backward traffic descriptor with its Cell Loss Priorities set to zero (0)

```
SmartSwitch # add trafficdescriptor
TrafficType(UBR)                          :  — This is the backward traffic descriptor
TrafficDescriptorType(11)                 :
PCRCLP01(100)                             : 0 — Set PCRCLP01 to zero
QOSCLASS(0)                               :
AalType(5)                                :

SmartSwitch #
```

2. Use **show trafficdescriptor** to obtain the new traffic descriptors' index numbers.

```
SmartSwitch # show trafficdescriptor

================================================================================

TD#  Traff    Desc  QoS  Peak Cell Rate  Sust Cell Rate  Max Burst Size  Min Cell Aal Type
     Type     Type         (Kb/s)           (Kb/s)          (Kb/s)      Rate
                         CLP_0  CLP_0+1  CLP_0  CLP_0+1  CLP_0  CLP_0+1  (Kb/s)
================================================================================
1    NRTVBR   7     0    0      10872    5436   0        2052   0        0         5
2    CBR      2     1    0      100      0      0        0      0        0         5
3    UBR      11    0    0      100      0      0        0      0        0         5
4    UBR      11    0    0      0        0      0        0      0        0         5
176  NRTVBR   2     1    0      1585     0      0        0      0        0         5

SmartSwitch #
```

In the example above, traffic descriptor three (3) will be used in the forward direction, and traffic descriptor four (4) will be used in the backward direction.

3. Use **add pvc** to successively create point-to-multipoint PVCs for ports **7a2**, **7a3**, and **7a4**.

```
SmartSwitch # add pvc
ConnType(PTP)                             : pmp
Port-1-Number()                           : 7a1
Port-1-VPCI()                             : 0
Port-1-VCI()                              : 101
Port-2-Number()                           : 7a2
Port-2-VPCI()                             : 0
Port-2-VCI()                              : 101
Port1-to-Port2TrafficDescriptorIndex()    : 3
Port2-to-Port1TrafficDescriptorIndex()    : 4

SmartSwitch #
```

4. Perform step 3 for ports **7a3** and **7a4**.

5. Connect the workstations to their respective ports.
6. Configure the workstations for the same subnet and VPCI/VCI pair = 0/101.

The broadcasting workstation on port **7a1** can send traffic to the receiving workstations on ports **7a2**, **7a3**, and **7a4**.

5.1.3 Connecting to Local Switch Client Through a PVC

All PVC connections to an ATM SmartSwitch local client use the CPU port. On a SmartSwitch 6500, this port is either **7B4** or **8B4** depending on the slot in which the master TSM/CPU module resides. Because of the SmartSwitch 6500's redundancy capability, the CPU port should always be designated as **CPU**. Using **CPU** assures that the PVC connects to the active CPU in the event of fail-over. On all other SmartSwitch ATM switches (2500, 6A000, or 9A100), the CPU port is **B4**, however, as with the SmartSwitch 6500, the value **CPU** can also be used.

Follow these instructions to connect an end node to an ATM SmartSwitch's local client through a point-to-point PVC.

1. Use **add pvc** to create the PVC.

```
SmartSwitch # add pvc
ConnType(PTP)                              :
Port-1-Number()                            : 7a1
Port-1-VPCI()                              : 0
Port-1-VCI()                               : 100
Port-2-Number()                            : cpu    — The CPU port
Port-2-VPCI()                              : 0
Port-2-VCI()                               : 101
Port1-to-Port2TrafficDescriptorIndex()     : 2
Port2-to-Port1TrafficDescriptorIndex()     : 2

SmartSwitch #
```

2. Use **add ipatmclient** to create an IP over ATM local client.

```
SmartSwitch # add ipatmclient
ClientNumber(0)                            : 2       — Set client number 2
ServerType(None)                           : local   — ARP server is on the switch
ServerAddress()                            :
IPAddress()                                : 100.1.1.0
NetMask(255.0.0.0)                         :
MTU(9180)                                  :

SmartSwitch #
```

3. Use **add ipatmpvc** to associate the end node's IP address with the PVC.

```
SmartSwitch # add ipatmpvc
ClientNumber(0)                            : 2        — Specify local client number
DestinationVPCI(0)                         :
DestinationVCI(33)                         : 101      —VCI to CPU port was specified as 101

SmartSwitch #
```

4. Connect the end node to port **7a1** of the ATM SmartSwitch.

5.2 PVP CONNECTIONS

Note PVP connections are supported only on the SmartSwitch 6500. However, because all ATM SmartSwitches support virtual ports, PVPs can be terminated using any SmartSwitch ATM switch.

The SmartSwitch 6500 supports the creation of Permanent Virtual Path (PVP) connections. The basic process for creating a PVP is as follows:

- Create a traffic descriptor for the PVP that meets its bandwidth and service category requirements.
- Use the **`set portconfig`** command to turn off signaling and ILMI on both ports to be connected by the PVP.

Note Dedicated PVP switches do not signal on their physical ports. However, if desired, you can leave signaling active on physical ports on the SmartSwitch 6500.

- Use the **`set portconfig`** command to specify a number of bits to be used for VPIs (**`MaxVpiBits`** parameter). Note that a PVP cannot use VPI zero. Consequently, the number of VPI bits must be greater than zero (0) on both ports. Determine the number of Available VPIs from the **`MaxVpiBits`** setting by using the following equation:

  $$\text{Available VPIs} = 2^{\text{MaxVpiBits}}-1$$

  For example if **`MaxVpiBits`** is set to **`3`**, then Available VPIs is:

  $$\text{Available VPIs} = 2^{3}-1 = 8 -1 = 7 \text{ VPIs (VPIs 1 through 7)}$$

  We have seven Available VPIs (and not eight) because the zero (0) VPI cannot be used for PVPs.
- Use the **`add pvp`** command to create the PVP connection.

The following is a practical example of creating a PVP connection between ports **`7a4`** and **`7b1`**.

1. Use the **`set portconfig`** command to turn off signaling and ILMI and to specify bits for VPIs on port **`7a4`**:

```
SmartSwitch # set portconfig
PortNumber()                          : 7a4       — Specify first port for PVP
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   : down      — Turn off ILMI
SigType()                             : nnipvc    — Turn off signaling
SigRole(network)                      :
InterfaceType(private)                :
MaxVpiBits(0)                         : 1         — 1 bit for VPIs: 2^1-1 = 1 VPI
MaxVciBits(12)                        :
MaxSvcVpci(1)                         :
MinSvcVci(32)                         :
MaxVccs(8192)                         :
MaxSvpVpci(1)                         :
MaxVpcs(1)                            :

SmartSwitch #
```

2. Use the **`set portconfig`** command to turn off signaling and ILMI and to specify bits for VPIs on port **`7b1`**:

```
SmartSwitch # set portconfig
PortNumber()                              : 7b1     — Specify the second port
PortAdminStatus(up)                       :
IlmiAdminStatus(up)                       : down
SigType()                                 : nnipvc
SigRole(network)                          :
InterfaceType(private)                    :
MaxVpiBits(0)                             : 1       — 1 bit for VPIs: 2^1-1 = 1 VPI
MaxVciBits(12)                            :
MaxSvcVpci(1)                             :
MinSvcVci(32)                             :
MaxVccs(8192)                             :
MaxSvpVpci(1)                             :
MaxVpcs(1)                                :

SmartSwitch #
```

3. Use the **`add pvp`** command to create the pvp connection:

```
SmartSwitch # add pvp
ConnType(PTP)                             :       — See note below
Port-1-Number()                           : 7a4   — Specify the first port
Port-1-VPI()                              : 1     — Specify its VPI
Port-2-Number()                           : 7b1   — Specify the second port
Port-2-VPI()                              : 1     — Specify its VPI
Port1-to-Port2TrafficDescriptorIndex()    : 2     — Set the traffic descriptors
Port2-to-Port1TrafficDescriptorIndex()    : 2

SmartSwitch #
```

Note Point-to-multipoint PVPs are currently not supported on the SmartSwitch 6500.

4. Use the **`show pvp`** command to display the PVP connection:

```
SmartSwitch # show pvp
PortNumber(ALL)                           :
CrossConnectId(ALL)                       :
CrossConnectSubId(ALL)                    :

=========================================================================
Conn Conn  |         Low      |         High      | Admin
Id   SubId | Port    VPI  Type| Port    VPI  Type | Status
=========================================================================
3    1       7A4     1    PTP   7B1     1    PTP    UP

Total number of PVPs = 1

SmartSwitch #
```

In the example above, we stopped ILMI and signaling on the ports used for the PVP. Stopping ILMI and signaling is characteristic of a “true” PVP connection. However, if necessary, a PVP can be created between ports running ILMI and signaling. In this case, the PVP coexists with the rest of the connections (if any) established across the connection.

5.2.1 Connecting PVPs

PVPs are physically connected to other devices in the following two ways:

- Physically connecting the PVP port to another PVP switch

  When connecting to another PVP switch, the VPI numbers assigned to the ports carrying the PVP on each switch must match. For example if a PVP exits switch 1 on port **7A1** and enters switch 2 on port **3B4**, the VPI number assigned to port **7A1** on switch 1 and port **3B4** on switch 2 must be the same (see Figure 5-1).

- Terminating the PVP port to a virtual port

  PVPs can be terminated on virtual ports (see Section 5.3). To terminate a PVP on a virtual port, the *virtual port number* must be the same as the VPI number for the PVP (see Figure 5-1). For example, to terminate a PVP with VPI number of 3, physically connect it to a port that contains a virtual port with virtual port number equal to three (**7a1.3, 5b2.3, A1.3, C5.3,** and so on).

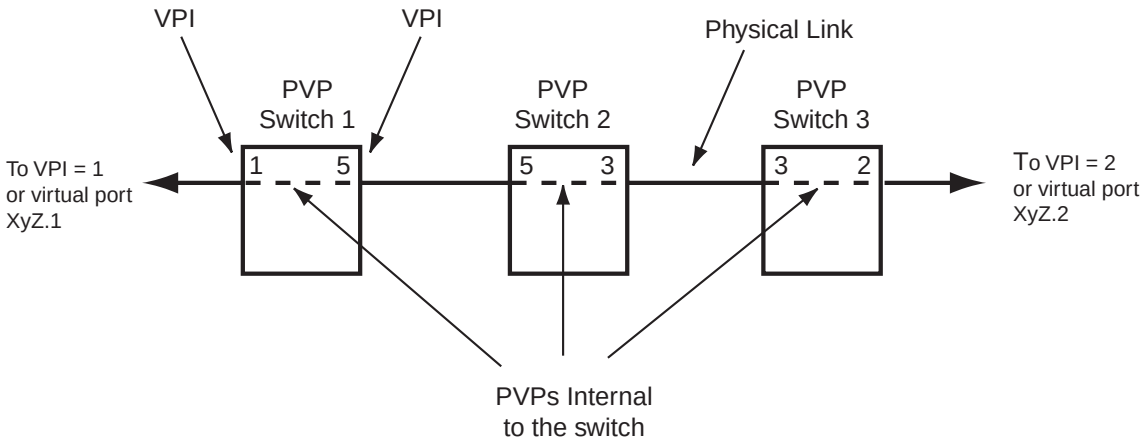

Figure 5-1 Terminating PVPs

5.3 VIRTUAL PORTS

ATM SmartSwitches support the ability to create virtual ports. Typically, virtual ports are used for terminating Permanent Virtual Path (PVP) connections. Virtual ports are designated by the following convention:

number of the physical port + a period + virtual port number

For example, **7a1.3**, **3a4.7**, **B2.5, A1.3**, and so on.

Note Zero (0) cannot be used as a a virtual port value. Virtual port zero (0) is reserved, and represents the physical port. For example, **7A1.0** and **B2.0** represent the physical ports **7A1** and **B2**, and are not available for designating virtual ports.

5.3.1 Creating Virtual Ports

Virtual ports are created on physical ports by first allocating a range of Virtual Path Identifiers (VPIs), and then distributing the VPIs among the virtual ports. The number of VPIs used depends on the number of virtual ports needed and the range of VPIs controlled by each virtual port.

When creating virtual ports, it's important to remember that the virtual port number represents the *Base VPI* used by the virtual port. For example, the virtual port **`5b1.3`** uses Base VPI = 3.

Creating virtual ports on an ATM SmartSwitch consists of the following basic process

- Create a traffic descriptor for the virtual port that meets its bandwidth and service category requirements.

Note To assure that virtual ports receives the exact bandwidth required, you may want to assign them traffic descriptors that specify CBR as the service class.

- Use the **`set portconfig`** command to turn off signaling on the physical port on which you are creating the virtual ports.

Note Signaling is usually not used on physical ports on which virtual ports are created. However, you can leave signaling active on the physical ports if necessary.

- Use the **`MaxVpiBits`** parameters of the **`set portconfig`** command to set the number of bits to use for VPIs for virtual ports on this physical port:

$$\text{Available VPIs} = 2^{\text{MaxVpiBits}} - 1$$

For example, if **`MaxVpiBits`** is set to 3, then the number of VPIs available for virtual ports is:

$$\text{Available VPIs} = 2^3 - 1 = 8 - 1 = 7$$

Note The value for Available VPIs is also the *highest* number that can be used to specify a virtual port on the physical port. For instance, in the example above, **`7a1.7`** is the highest virtual port that can be created using **`MaxVpiBits = 3`**.

- Use the **`add port`** command to create the virtual port and to specify the number of VPIs used by the virtual port. Note that the **`add port`** command also uses the **`MaxVpiBits`** parameter, however, here it's used to define the number of VPIs the virtual port uses, based on the equation:

$$\text{VPIs Used by Virtual Port} = \text{Base VPI} + (2^{\text{MaxVpiBits}}-1)$$

For example, if the virtual port number is **`5b2.1`** (Base VPI = 1), and **`MaxVpiBits = 1`**, then the total number of VPIs used by this virtual port is:

$$\text{Base VPI} + (2^1-1) = 1 + (2-1) = 1 + 1 = 2 \text{ VPIs}$$

So port **`5b2.1`** controls VPI 1 (the Base VPI) and VPI 2.

Note For PNNI, the number of VPIs used by each virtual port should be one (1). For virtual UNI, the number of VPIs used by each virtual port should correspond to the number of VPIs on the user side of the UNI connection (For information on virtual UNI, refer to the ATM Forum specification for ILMI 4.0.).

The following is a practical, step-by-step example of creating a virtual port on physical port **7A1** that controls a single VPI.

1. Use the **set portconfig** command to turn signaling off on physical port **7a1**:

```
SmartSwitch # set portconfig
PortNumber()                          : 7a1
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   :
SigType(autoConfig)                   : nnipvc  — Turn off signaling by setting SigType to nnipvc
SigRole(network)                      :
InterfaceType(private)                :
MaxVpiBits(0)                         :         — Default MaxVpiBits = 0
MaxVciBits(13)                        :         — Default MaxVciBits = 13
MaxSvcVpci(0)                         :
MinSvcVci(32)                         :
MaxVccs(8192)                         :
MaxSvpVpci(0)                         :
MaxVpcs(0)                            :

SmartSwitch #
```

2. Use the **set portconfig** command to assign two bits to **MaxVpiBits**.:

```
SmartSwitch # set portconfig
PortNumber()                          : 7a1
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   :
SigType(nniPvc)                       :
SigRole(network)                      :
InterfaceType(private)                :
MaxVpiBits(0)                         : 1       — Set to 1 — this translates to VPIs = 2^1-1 = 1
MaxVciBits(12)                        :         — Notice that MaxVciBits has reduced itself by 1 bit
MaxSvcVpci(7)                         :
MinSvcVci(32)                         :
MaxVccs(8192)                         :
MaxSvpVpci(7)                         :
MaxVpcs(7)                            :

SmartSwitch #
```

Note The command **set portconfig** is used here twice for the purposes of clarity only. Normally, you would turn off signaling and set the **MaxVpiBits** within the same instance of **set portconfig**.

3. Use the **PortNumber** and **MaxVpiBits** parameters of the **add port** command to create the virtual ports.

```
SmartSwitch # add port
PortNumber()                          : 7a1.1   — The .1 means our Base VPI is one (1)
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   :
SigType(autoConfig)                   :
SigRole(other)                        :
InterfaceType(private)                :
MaxVpiBits(0)                         : 0       — VPIs used = Base VPI + (2^0 - 1) = 1 + 0 = 1
MaxVciBits(10)                        :
MaxSvcVpci(1)                         :         — Confirms that we have only one VPCI for this virtual port
MinSvcVci(32)                         :
MaxVccs(2048)                         :
TrafficDescriptorIndex()              : 1       — Specify traffic descriptor to be used with virtual port

SmartSwitch #
```

Our virtual port is now created, and uses just one VPI: the Base VPI (**.1**).

The following is an example creates virtual port **7b2.4**, which uses seven VPIs, starting at Base VPI = 4.

1. Use the **set portconfig** command to turn off signaling and set the **MaxVpiBits** to **4**:

```
SmartSwitch # set portconfig
PortNumber()                          : 7b2       — Specify physical port to contain the virtual port
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   :
SigType(autoConfig)                   : nnipvc    — Turn off signaling
SigRole(network)                      :
InterfaceType(private)                :
MaxVpiBits(0)                         : 4         — Available VPIs are set to 2^4 - 1 = 16 - 1 = 15 VPIs
MaxVciBits(9)                         :           — MaxVciBits decrements by 4
MaxSvcVpci(15)                        :
MinSvcVci(32)                         :
MaxVccs(8192)                         :
MaxSvpVpci(15)                        :
MaxVpcs(15)                           :

SmartSwitch #
```

2. Use the **add port** command to create the port and to specify the number of VPIs:

```
SmartSwitch # add port
PortNumber()                          : 7b2.4   — Specify virtual port number (and Base VPI)
PortAdminStatus(up)                   :
IlmiAdminStatus(up)                   :
SigType(autoConfig)                   :
SigRole(other)                        :
InterfaceType(private)                :
MaxVpiBits(0)                         : 3       — VPIs used = Base VPI + (2^3 - 1) = 4 + 7 = 11
MaxVciBits(9)                         :
MaxSvcVpci(7)                         :         — Confirms that there are seven VPCI for this virtual port
MinSvcVci(32)                         :
MaxVccs(4096)                         :
TrafficDescriptorIndex()              : 1

SmartSwitch #
```

In the example above, the virtual port controls eight VPIs. Counting from the Base VPI, these are 4, 5, 6, 7, 8, 9, 10, and 11. Notice that other virtual ports can be created on this physical port because we haven't used all of the available VPI specified by the **set portconfig** command. For example, the next (higher) virtual port that's possible to create is **7b2.12** because the Base VPI is beyond the eight VPIs used by **7b2.4**.

Things To Watch Out For When Creating Virtual Ports

- Make certain that the virtual port number (Base VPI) plus the VPIs designated by **`MaxVpiBits`** does not exceed the Available VPIs as specified by **`MaxVpiBits`** in the **`set portconfig`** command.
- If you create more than one virtual port on a particular physical port, make certain that you do not run out of Available VPIs as specified by **`MaxVpiBits`** in the **`set portconfig`** command.
- If you create more than one virtual port on a particular physical port, make certain that no overlap occurs among the VPIs used by the virtual ports.
- Make sure the CAC policy is set correctly for the number of virtual ports.
- Make certain that the traffic descriptors used by the virtual ports were created with the appropriate bandwidth and category of service.
- Use the **`set cacserviceclassbw`** command (on a per-port basis) to allocate sufficient bandwidth to the specified service class

5.4 SOFT PVC AND PVP CONNECTIONS

The SmartSwitch 6500 supports both soft (or smart) PVC and soft PVP connections. Soft PVCs and PVPs are used to create PVC and PVP connections between ports on separate switches that are separated by a PNNI network. Normally, PVCs and PVPs must be configured manually from switch-to-switch across the network. However, soft PVCs and PVPs need to be configured only at the source and target switches. The connection is then routed through the PNNI network. Additionally, soft PVCs and PVPs take advantage of PNNI's self-healing and crank-back capabilities. With conventional PVCs (for example), it a link goes down on the network, the PVC connection is broken. With soft PVCs, however, if a link goes down, PNNI has the capability of finding an alternate path to the target, thereby reestablishing the PVC connection.

Note Soft PVPs are supported on the SmartSwitch 6500 ATM switch only.

Note Only point-to-point soft PVCs and soft PVPs are currently supported.

5.4.1 Soft PVC and Soft PVP differences

The differences between soft PVCs and soft PVPs are essentially the same as those between standard PVC and PVP connections:

- Soft PVCs are identified by a VPI number and VCI number
- Soft PVPs use only the VPI (VPCI)
- Soft PVPs must use a VPI > 0
- Soft PVPs must be eventually terminated on virtual ports

5.4.2 Making Soft PVC and PVP Connections

Creating soft PVC and PVP connections consists of the following general steps:

- Configure a target port and ATM target address on the target switch
- Create a traffic descriptor to be used by the connection
- Add a soft PVC (or PVP) on the source switch that specifies the port on the target switch as its end point

5.4.3 Creating a soft PVC

The following is a step-by-step example of creating a soft PVC from port **`7a1`** on the source switch to port **`6b3`** on the target switch. The two switches containing the soft PVC are separated by several switches, which are connected through PNNI (see Figure 5-2 and Figure 5-3).

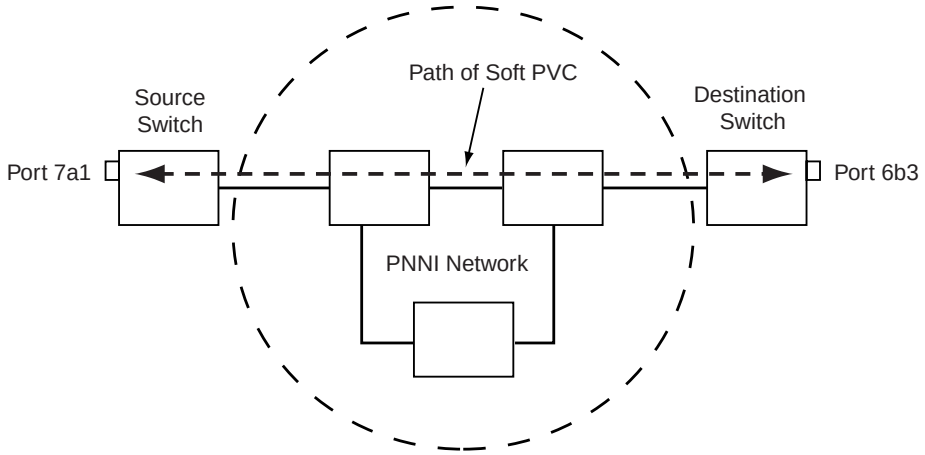

Figure 5-2 Soft PVC across PNNI network

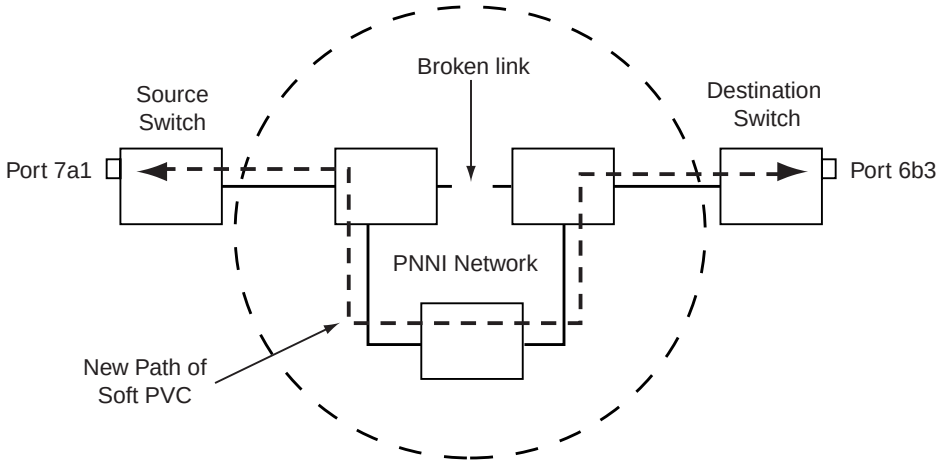

Figure 5-3 Soft PVC heals (is rerouted) to bypass broken link

1. Define a target ATM address to be used on the target switch. The target ATM address can be any address that is either eight (8) or twenty (20) bytes long and must not be identical to any address currently listed in the ATM routing table. Use the **`show atmroute`** command to check which addresses are currently defined.

```
SmartSwitch # show atmroute
Num(ALL)                                :

Num  Port Number ATM Address                                          Type Proto
================================================================================
1    --          39:00:00:00:00:00:00:00:00:00:14:41:80                  I PNNI
2    --          39:00:00:00:00:00:00:00:00:00:28:8d:00                  I PNNI
3    --          39:00:00:00:00:00:00:00:00:00:28:c1:80                  I PNNI
4    --          39:00:00:00:00:00:00:00:00:00:29:05:00                  I PNNI
5    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
6    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
7    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
8    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:34:77:81 I MGMT
9    --          39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:34:77:ff I MGMT
10   --          39:00:00:00:00:00:00:00:00:00:bf:ba:26                  I PNNI
11   CPU         47:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I MGMT
12   --          47:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I PNNI
13   CPU         c5:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I MGMT
14   --          c5:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I PNNI

SmartSwitch #
```

2. Use the **`add spvcaddress`** command on the target switch to specify the target port and ATM address.

```
SmartSwitch # add spvcaddress
PortNumber()                              : 6b3  — Port on target switch
AtmAddress()                            : 22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22

Added SPVC Address successfully.

SmartSwitch #
```

3. Use the **`show spvcaddress`** command to see the soft PVC port and ATM address on the target switch:

```
SmartSwitch # show spvcaddress
PortNumber(ALL)                         :
TargetAddress()                         :

Port          SPVC Target Address
================================================================================
6B3         22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22

Total number of SPVC Addresses = 1

SmartSwitch #
```

4. On the source switch, use the **`add trafficdescriptor`** command to create traffic descriptors for the forward and reverse directions of the connection (See Section 6.1.1for information about traffic descriptors).

5. On the source switch, use the **add spvc** command to create the soft PVC connection between the two switches:

```
SmartSwitch # add spvc
PortNumber()                              : 7a1 — Port on source switch
SourceVpi(0)                              : 0
SourceVci(32)                             : 101
DestinationSelectType(REQUIRED)           : — See note below
DestinationVPI(0)                         : 0
DestinationVCI(32)                        : 102
TargetAddress()                     : 22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22
TransmitTrafficDescriptorIndex()          : 3
ReceiveTrafficDescriptorIndex()           : 3
RetryInterval(10000)                      :
RetryLimit(3)                             :
RetryThreshold(1)                         :

SmartSwitch #
```

Note The **DestinationSelectType** determines which vpi/vci pair is used on the target switch. The possible settings are **REQUIRED** and **ANY**. If **DestinationSelectType** is set to **REQUIRED**, the specified target vpi/vci is set at the target switch. If **ANY** is specified, the soft PVC uses the first available vpi/vci pair it finds on the target switch. If **ANY** is specified, enter the **show spvctarget** command on the target switch to determine the vpi/vci pair used.

Enter the **show spvc** command on the target switch to see the soft PVC and its current state:

```
SmartSwitch # show spvc
PortNumber(ALL)                           :
SourceVpi(0)                              : 0
SourceVci(32)                             : 101

=====================================================
Port   Src VPI   Src VCI   Leaf Ref   Operation Status
=====================================================
7A1      0         101         1         connected

Total number of SPVCs = 1

SmartSwitch #
```

Note If you want to create soft PVCs that use VPI values other than zero (0), you must first use the **set portconfig** command to change the MaxVpiBits for the port from its default of zero (0) to a value that specifies a sufficient number of bits to create the VPI number. For example, if you want to use VPI = 3, change **MaxVpiBits** for that port to two (2). See Section 5.2 and Section 5.3 for more information about setting **MaxVpiBits**.

5.4.4 Creating a Soft PVP

Note Soft PVPs are supported only on the SmartSwitch 6500 ATM switch.

The following is an example of creating a soft PVP between port **7a1** on the source switch and port **6b3** on the target switch.

1. Use the **set portconfig** command on the target switch to increase the **MaxVpiBits**.

```
Smart6500_1 # set portconfig
PortNumber()                              : 7a1
PortAdminStatus(up)                       :
IlmiAdminStatus(up)                       :
SigType(autoConfig)                       :
SigRole(other)                            :
InterfaceType(private)                    :
MaxVpiBits(0)                             : 2     — Increase to two bits = 2^2-1 = 3 possible VPIs
MaxVciBits(11)                            :
MaxSvcVpci(3)                             :
MinSvcVci(32)                             :
MaxVccs(8192)                             :
MaxSvpVpci(3)                             :
MaxVpcs(3)                                :

Smart6500_1 #
```

2. On the target switch, define a target ATM address. The target ATM address can be any address that is either eight (8) or twenty (20) bytes long and must not be identical to any address currently listed in the ATM routing table. Use the **show atmroute** command to check which addresses are currently defined on the target switch.

```
SmartSwitch # show atmroute
Num(ALL)                                  :

Num  Port Number ATM Address                                                  Type Proto
================================================================================
1    --          39:00:00:00:00:00:00:00:00:00:14:41:80                        I PNNI
2    --          39:00:00:00:00:00:00:00:00:00:28:8d:00                        I PNNI
3    --          39:00:00:00:00:00:00:00:00:00:28:c1:80                        I PNNI
4    --          39:00:00:00:00:00:00:00:00:00:29:05:00                        I PNNI
5    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
6    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
7    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:00:1d:a3:87:0b I MGMT
8    CPU         39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:34:77:81 I MGMT
9    --          39:00:00:00:00:00:00:00:00:00:a3:87:0b:00:20:d4:34:77:ff I MGMT
10   --          39:00:00:00:00:00:00:00:00:00:bf:ba:26                        I PNNI
11   CPU         47:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I MGMT
12   --          47:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I PNNI
13   CPU         c5:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I MGMT
14   --          c5:00:79:00:00:00:00:00:00:00:00:00:00:a0:3e:00:00:01 I PNNI

SmartSwitch #
```

3. Use the **add spvcaddress** command on the target switch to specify the target port and ATM address.

```
SmartSwitch # add spvcaddress
PortNumber()                              : 6b3  — Port on target switch
AtmAddress()                            : 22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22

Added SPVC Address successfully.

SmartSwitch #
```

Note Both soft PVCs and Soft PVPs use the **add spvcaddress** command to specify the target switch's target ATM address. There is no separate "**add spvpaddress**" command.

4. Use the **show spvcaddress** command to see the soft PVP port and ATM address on the target switch:

```
SmartSwitch # show spvcaddress
PortNumber(ALL)                           :
TargetAddress()                           :

Port          SPVC Target Address
================================================================================
6B3         22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22

Total number of SPVC Addresses = 1

SmartSwitch #
```

5. On the source switch, use the **add trafficdescriptor** command to create traffic descriptors for the forward and reverse directions of the connection (See Section 5.1.1for information about traffic descriptors).

6. On the source switch, use the **add spvp** command to create the soft PVP connection between the two switches:

```
SmartSwitch # add spvp
PortNumber()                              : 7a1  — Port on source switch
SourceVpi(0)                              : 3
DestinationSelectType(REQUIRED)           :      — See note below
DestinationVPI(0)                         : 3    — We use VPI= 3
TargetAddress()                         : 22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22:22
TransmitTrafficDescriptorIndex()          : 3
ReceiveTrafficDescriptorIndex()           : 3
RetryInterval(10000)                      :
RetryLimit(3)                             :
RetryThreshold(1)                         :

SmartSwitch #
```

Note The **DestinationSelectType** determines which vpi is used on the target switch. The possible settings are **REQUIRED** and **ANY**. If **DestinationSelectType** is set to **REQUIRED**, the specified target vpi is set at the target switch. If **ANY** is specified, the soft PVP uses the first available vpi it finds on the target switch. If **ANY** is specified, enter the **show spvptraget** command on the target switch to determine the vpi used.

Enter the **show spvp** command on the target switch to see the soft PVP and its current state:

```
SmartSwitch # show spvp
PortNumber(ALL)                          :7a1
SourceVpi(0)                             : 3

=======================================================
Port   Src VPI   Leaf Ref   Operation Status
=======================================================
7A1      0          1         connected

Total number of SPVCs = 1

SmartSwitch #
```

6 TRAFFIC MANAGEMENT

6.1 TRAFFIC MANAGEMENT CAPABILITIES

ATM SmartSwitches have extensive abilities for managing traffic flow. Traffic management includes all operations performed by the ATM SmartSwitch that ensures optimum switch throughput, where throughput is based on rate of packet loss, available bandwidth, and traffic processing overhead. Under most conditions, an ATM SmartSwitch can efficiently and automatically manage switch traffic. However, if necessary, you can adjust the switch traffic management parameters. For example, it might be necessary to adjust parameters for a port that carries a large amount of CBR traffic or a very large number of simultaneous connections.

ATM SmartSwitches provide console commands that affect traffic flow on a global, port, or category of service level. These console commands affect switch traffic flow by controlling

- Bandwidth allocation
- Call Admission Control (CAC) policies
- The service category for a connection
- Buffer memory allocation
- Threshold settings for anti-congestion routines

Caution Do not change traffic control settings unless you have expert-level experience with ATM switching. Back up the switch configuration before making changes. Also, make notes of the changes you make to the traffic control parameters.

6.1.1 Traffic Descriptors

Traffic characteristics of an ATM source are signaled through a set of traffic descriptors during connection establishment. ATM SmartSwitches use traffic descriptors for resource allocation during call set up to guarantee the quality of service (QoS) across the connection. The source traffic descriptor is a set of parameters that describes the expected class of service and bandwidth utilization of a connection. Depending on the class of service specified in the traffic descriptor you can set the following parameters:

- Peak Cell Rate (PCR)
- Sustainable Cell Rate (SCR)
- Maximum Burst Size (MBS)
- Minimum Cell Rate (MCR) — signaled through UNI4.0 signaling only
- AAL type

If a connection is bi-directional, a traffic descriptor has to be assigned to each direction and need not be the same in both directions.

ATM SmartSwitch user data cells are classified according to the state of a cell loss priority (CLP) bit in the header of each cell. A CLP 1 cell has a lower priority than a CLP 0 cell and is discarded first. Source traffic descriptors can specify CLP 0 cell traffic, CLP 1 cell traffic, or the aggregate CLP 0+1 traffic.

Use the **trafficdescriptor** commands to view, create, and delete traffic descriptors.

For example, enter the **show trafficdescriptor** command to view all currently defined traffic descriptors.

```
SmartSwitch # show trafficdescriptor
================================================================================

TD#  Traff    Desc  QoS  Peak Cell Rate  Sust Cell Rate  Max Burst Size  Min Cell Aal Type
     Type     Type        (Kb/s)          (Kb/s)          (Kb/s)      Rate
                        CLP_0  CLP_0+1  CLP_0  CLP_0+1  CLP_0  CLP_0+1  (Kb/s)
================================================================================
1    NRTVBR   7     0    0      10872    5436   0        2052   0        0         5
2    CBR      2     1    0      100      0      0        0      0        0         5
176  NRTVBR   2     1    0      1585     0      0        0      0        0         5

SmartSwitch #
```

Note You cannot use the default traffic descriptors for user-defined PVCs. All traffic descriptors used to define PVCs must be created by the user.

The Descriptor Type parameter in the example above corresponds to the traffic descriptor types defined in the UNI3.0/UNI3.1 specification. Descriptor types are specified numerically and correspond to the descriptions in Table 6-1.

Table 6-1 Traffic descriptor type number explanation

Type	Valid Service Category	Descriptor Characteristics
1		No Traffic Descriptor
2	CBR	PeakCellRate CLP0+1
3	CBR	PeakCellRate CLP0+1, PeakCellRate CLP0
4	CBR	PeakCellRate CLP0+1, PeakCellRate CLP0, Tag CLP = 1
5	VBR	PeakCellRate CLP0+1, SustCellRate CLP0+1, MaxBurstSize CLP0+1
6	VBR	PeakCellRate CLP0+1, SustCellRate CLP0, MaxBurstSize CLP0
7	VBR	PeakCellRate CLP0+1, SustCellRate CLP0, MaxBurstSize CLP0, Tag CLP = 1
8	ABR	PeakCellRate CLP0+1, Minimum Cell Rate
11	UBR	BestEffort

A user-defined PVC must have user-defined traffic descriptors. For instance, if a video link over a PVC requires a peak cell rate of 8000 kb/s, create a traffic descriptor for CBR traffic that specifies 8000 as the peak cell rate.

```
SmartSwitch # add trafficdescriptor
TrafficType(UBR)                          : cbr
TrafficDescriptorType(2)                  :3
PCRCLP01(100)                             :8000
QOSCLASS(1)                               :
AalType(5)                                :

SmartSwitch #
```

Each traffic descriptor is identified by a unique index number. Use the index number to specify which traffic descriptor to use when setting up a PVC. For example, the **add pvc** command prompts you for the traffic descriptor index.

```
SmartSwitch #  add pvc
ConnType(PTP)                             :
Port-1-Number()                           : 7a1
Port-1-VPCI()                             : 0
Port-1-VCI()                              : 100
Port-2-Number()                           : 7b2
Port-2-VPCI()                             : 0
Port-2-VCI()                              : 100
Port1-to-Port2TrafficDescriptorIndex()    : 3     — Forward traffic descriptor
Port2-to-Port1TrafficDescriptorIndex()    : 2     — Backward traffic descriptor

SmartSwitch #
```

Notice in the example above that you can use different traffic descriptors for forward and backward traffic provided that both traffic descriptors used belong to the same service category.

6.1.2 Call Admission Control Policy

Call Admission Control (CAC) policy defines the bandwidth allocation scheme used by the CAC when setting up connections. ATM SmartSwitches offer three schemes that can be set on a per-port, per-service class basis,

- Conservative
- Moderate
- Liberal

Under conservative policy, the CAC allocates bandwidth closest to the requested bandwidth and QoS parameters. Conversely, liberal policy causes the CAC to allocate the least amount of bandwidth. And the CAC under moderate policy allocates intermediate amounts of bandwidth.

Depending on the type of traffic on your network, each of these CAC policies has its advantages. For instance, liberal policy allows a larger number of connections over that of the conservative or moderate policy. Liberal policy assumes that the traffic pattern of individual VCs does not overlap most of the time. For example, if VC1 and VC2 are created under the liberal CAC policy, it's assumed that the probability of both VCs sending large bursts of cells at the same time is relatively low. On the other hand, conservative policy assumes that there might be a larger overlap of traffic from different VCs, and provides each VC with bandwidth closer to the requested bandwidth. This higher bandwidth provides a guarantee of quality for each VC.

Use the command **show caceqbwallocscheme** to view the current CAC policies used by each port for each class of service.

```
SmartSwitch # show caceqbwallocscheme
PortNumber(ALL)                          :
=========================================================
Port#                  Alloc Scheme
                             for
            CBR     RTVBR   NRTVBR   UBR    ABR
=========================================================
7A1         CON      CON     CON     LIB    CON
7A2         CON      CON     CON     LIB    CON
7A3         CON      CON     CON     LIB    CON
7A4         CON      CON     CON     LIB    CON
7B1         CON      CON     CON     LIB    CON
7B1.3       CON      CON     CON     LIB    CON
7B2         CON      CON     CON     LIB    CON
7B3         CON      CON     CON     LIB    CON
CPU         CON      CON     CON     LIB    CON
CPU.1       CON      CON     CON     LIB    CON

SmartSwitch #
```

Note The CAC affects both physical and virtual ports as indicated in the example above (**7b1.3** is a virtual port).

If there are a large number of connections of a particular class of service on a particular port, and these connections begin to slow down and show signs of congestion, use the **set caceqbwallocscheme** command to change the CAC policy to moderate or conservative.

```
SmartSwitch # set caceqbwallocscheme
PortNumber()                             : 7a1
SeriveCategory(CBR)                      : ubr
AllocScheme(LIBERAL)                     : moderate

SmartSwitch #
```

Use the **set cacserviceclassbw** command to change the amount of bandwidth on a per-port basis that the CAC recognizes as available for each class of service. Available bandwidth for a class of service is specified as a percent of total port bandwidth. For example, to increase the bandwidth for CBR calls on port **7a1** to 20 percent of total port bandwidth, enter the following

```
SmartSwitch # set cacserviceclassbw
PortNumber()                             : 7a1
MaxBandWidth_In_Percentage-CBR(1)        : 20     — Increase to 20%
MaxBandWidth_In_Percentage-RT_VBR(1)     :
MaxBandWidth_In_Percentage-NRT_VBR(7)    :
MaxBandWidth_In_Percentage-UBR(89)       : 70     — Decrease by 20%
MaxBandWidth_In_Percentage-ABR(1)        :

SmartSwitch #
```

Notice in the example above that the total percentage for all service classes on the port must not exceed 100 percent. Furthermore, if the **set cacserviceclassbw** command is used to alter a physical port, the change also affects any virtual ports on that physical port.

6.1.3 Queue Buffers

ATM SmartSwitches perform buffering using a shared-memory architecture. Buffer space is divided into queues for each class of service. In turn, ports are allocated a portion of each of the service class queues. This allocation is controlled on a per-port basis by the **porttrafficcongestion** commands.

Quality of service is defined on an end-to-end basis in terms of cell loss ratio, cell transfer delay, and cell delay variation.

For example, enter the **show porttrafficcongestion** command to view current buffer utilization.

```
SmartSwitch # show porttrafficcongestion
PortNumber(ALL)                          :

PortID QueueId ServiceClass MinIndex MinValue MaxIndex MaxValue

================================================================================
CPU    1       CBR          10       64       15       1024
CPU    2       RTVBR        8        256      13       4096
CPU    3       NRTVBR       8        256      13       4096
CPU    4       ABR          8        256      12       8192
CPU    5       UBR          8        256      12       8192

PortID QueueId ServiceClass MinIndex MinValue MaxIndex MaxValue

================================================================================
7A1    1       CBR          10       64       15       1024
7A1    2       RTVBR        8        256      13       4096
7A1    3       NRTVBR       8        256      13       4096
7A1    4       ABR          8        256      12       8192
7A1    5       UBR          8        256      12       8192
PortID QueueId ServiceClass MinIndex MinValue MaxIndex MaxValue

================================================================================
7A2    1       CBR          10       64       15       1024
7A2    2       RTVBR        8        256      13       4096
7A2    3       NRTVBR       8        256      13       4096
7A2    4       ABR          8        256      12       8192
7A2    5       UBR          8        256      12       8192

PortID QueueId ServiceClass MinIndex MinValue MaxIndex MaxValue

================================================================================
7A3    1       CBR          10       64       15       1024
7A3    2       RTVBR        8        256      13       4096
7A3    3       NRTVBR       8        256      13       4096
7A3    4       ABR          8        256      12       8192
7A3    5       UBR          8        256      12       8192

More(<space>/q)?:
```

MinValue and **MaxValue** are thresholds set on a per-queue, per-port basis and are measured in cells (53 bytes). The **MinValue** threshold is the amount of buffer space guaranteed to a call of a particular service class on the corresponding port. The **MaxValue** threshold is the maximum amount of buffer space that a call of a particular service class is allowed on the corresponding port.

QoS corresponds to the queues as follows:

- Queue 1 — Constant Bit Rate (CBR)
- Queue 2 — Real Time Variable Bit Rate (rt-VBR)
- Queue 3 — Non-real time Variable Bit Rate (Nrt-VBR)

- Queue 4 — Available Bit Rate (ABR)
- Queue 5 — Unspecified Bit Rate (UBR)

If calls of a particular service class are being dropped on a particular port, use the **`set porttrafficcongestion`** command to raise the port's queue Min threshold.

For example, to change both the Min and Max amounts of buffer space used for CBR calls on port **`7a3`**, first enter the **`show porttrafficcongestion`** command to determine the current minimum threshold level:

```
SmartSwitch # show porttrafficcongestion
PortNumber(ALL)                          : 7a3

PortID QueueId ServiceClass MinIndex MinValue MaxIndex MaxValue

================================================================================
7A3    1       CBR          10       64       15       1024
7A3    2       RTVBR        8        256      13       4096
7A3    3       NRTVBR       8        256      13       4096
7A3    4       ABR          8        256      12       8192
7A3    5       UBR          8        256      12       8192

SmartSwitch #
```

CBR on port **`7a3`** is currently using 64 (**`MinIndex 10`**) as its minimum threshold. Use the **`show minmax`** command to determine a new minimum threshold for CBR:

```
SmartSwitch # show minmax
----------------------------------------
MinIndex  MinValue  MaxIndex  MaxValue
----------------------------------------
0         65536     0         1048576
1         32768     1         786432
2         16384     2         524288
3         8192      3         393216
4         4096      4         262144
5         2048      5         196608
6         1024      6         131072
7         512       7         98304
8         256       8         65536
9         128       9         49152
10        64        10        32768
11        32        11        16384
12        16        12        8192
13        8         13        4096
14        4         14        2048
15        0         15        1024

SmartSwitch #
```

From the table, we'll select 128 (**`MinIndex 9`**). Use the set porttrafficcongestion command to assign this value to CBR for port **`7a3`**.

```
SmartSwitch # set porttrafficcongestion
Port(ALL)                                : 7a3
QueueNumber()                            : 1     — Corresponds to CBR
MinIndexNumber()                         : 9     — MinIndex for 128
MaxIndexNumber()                         : 15    — Specify the current MaxIndex
```

6.1.4 EFCI, EPD, and RM Cell Marking

To control switch congestion, ATM SmartSwitches implement standard resource management cell (RM-cell) marking, explicit forward congestion indicator cell marking (with backward RM cell marking), and early packet discard (EPD). These congestion control schemes are triggered when the number of cells within shared memory reaches user-definable thresholds. Use the **switchtrafficcongestion** commands to view and set these thresholds.

For example, enter the **show switchtrafficcongestion** command.

```
SmartSwitch # show switchtrafficcongestion

Switch Traffic Congestion Parameters
================================================================================
Low EPD Threshold                       : 209715 cells
High EPD Threshold                      : 104857 cells
CLP1 Discard Threshold                  : 131072 cells
RM Cell Marking Enable                  : OFF
EFCI Cell Marking Enable                : OFF
Explicit Rate Marking Enable            : OFF

SmartSwitch #
```

For most types of traffic, EPD triggering is tied to the low EPD threshold. Signaling traffic, however, is tied to the high EPD threshold; this assures that signaling packets are discarded only when congestion is most severe.

Use the **set switchtrafficcongestion** command to change thresholds for EPD and to enable or disable RM and EFCI cell marking. For example:

```
SmartSwitch # set switchtrafficcongestion
LowEPDWatermark(4096)                    :
HighEPDWatermark(4096)                   :
CLP1_DiscardWatermark(4096)              :
RMCellMarkingEnable(enable)              :
ExplicitRateMarkingEnable(enable)        :
EFCIMarkingEnable(enable)                :

SmartSwitch #
```

7 FIRMWARE UPGRADES AND BOOTLINE COMMANDS

7.1 UPDATE FIRMWARE COMMANDS

You can upgrade the operating firmware of an ATM SmartSwitch while the switch is running its current firmware. This procedure is known as a hot upgrade and is accomplished by the **`update firmware`** command.

When an ATM SmartSwitch is started (or rebooted), it copies its operating firmware from flash RAM to the CPU's program memory. When a hot upgrade is performed, the image in flash RAM is erased and replaced with the new firmware image. While the upgrade is occurring, the switch continues to run its copy in program memory. When the switch is rebooted, the new firmware image residing in flash RAM is copied into system memory and then run.

To use the hot upgrade feature, the ATM SmartSwitch must have network access to an end station running TFTP server software. The ATM SmartSwitch operating firmware file must reside within the directory specified by the TFTP server software. Often, this directory is /tftpboot. However, it may be different with your TFTP server software.

The following is an example of a hot upgrade:

```
SmartSwitch # update firmware
ServerIP()                                : 206.61.237.127            — IP address of TFTP server
Path(public/server.ima)                   : builds/luxor2/server.ima  — Path and name of file to download

You are updating the code image in the flash.

Are you sure this is what you want to do?
Confirm(y/n)?:y                                                       — Specify Yes to start download process

Verifying bootfile builds/luxor2/server.ima on 206.61.237.127
...passed.

Erasing Flash.

Using TFTP to get and program bootfile builds/luxor2/server.ima from 206.61.237.127.
4904K (5021760 bytes) received.

Flash update succeeded.

You will have to reboot for the new image to take effect.

SmartSwitch #
```

Notice that the **`update firmware`** command does not use Bootp to find the TFTP server. Instead, the **`update firmware`** command requires that you specify the IP address of the TFTP server, the path to the image file, and the file name.

Unsuccessful Update

If the **`update firmware`** command fails, DO NOT turn off or attempt to reboot your ATM SmartSwitch. In its current state, the operating firmware normally stored in flash RAM is erased. The switch is functioning only because it is running the image of the operating firmware that resides in volatile system memory.

If possible, determine why the **update firmware** command failed. Possible causes are:

- The ATM SmartSwitch lost network connectivity before it finished its download
- The wrong file or a corrupt file was downloaded into memory

If you can correct the problem, enter the **update firmware** command to continue with the upgrade process. However, if you are unable to correct the problem, use the **df** (download flash) command and a TFTP/Bootp server to replace the operating firmware on your ATM SmartSwitch. Follow the procedure outlined below:

1. Set up TFTP/Bootp server software on a workstation.
2. Connect both the TFTP/Bootp server and the ATM SmartSwitch to your Ethernet network. Make sure that the TFTP/Bootp server can be reached by ATM SmartSwitch Ethernet interface.
3. Connect a dumb terminal (or workstation running terminal emulation software) to the SmartSwitch Terminal port.
4. Copy the ATM SmartSwitch operating firmware image into the appropriate location on the TFTP/Bootp server.
5. Set up the TFTP/Bootp server tables (or equivalent file) with the ATM SmartSwitch MAC address and IP address. You may also need to specify the path to the image file to be downloaded.
6. From the terminal connection, enter the **reboot** command.
7. When the following message appears,

   ```
   "Press any key to exit to bootline prompt. "
   ```

   stop the countdown by pressing any key. The bootline prompt (=>) appears on the terminal screen.
8. Enter the **df s** command. The ATM SmartSwitch contacts the TFTP/Bootp server and downloads the operating firmware into its flash RAM.

```
=>df s
You've requested a Switch Software download
Are you sure?(Y/N)y
Initializing ethernet...
Starting Bootp...
   Boot file: c:\tftpboot\images\server.ima
Using TFTP to get bootfile "c:\tftpboot\images\server.ima" .
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................
Validity checks of the Switch Software Downloaded file...
All Validity checks OK
Programming downloaded image into Switch Software section, please wait...
New Switch Software programmed successfully
=>
```

9. Enter the **go** command to start the ATM SmartSwitch.

7.2 BOOTLINE COMMANDS

This section describes the low-level bootline commands. Bootline commands are used for setting switch start-up behavior and for performing firmware downloads. Use the bootline commands to:

- Set which copy of the boot load firmware is the default copy
- Perform a "low-level" format of the flash file system
- Check boot load firmware version numbers

- Load switch firmware upgrades
- Set whether power-on system tests (POST) are automatically run at start-up
- Change the master/slave relationship for TSM/CPUs and CSMs on SmartSwitch 6500s

7.2.1 Accessing the Bootline Prompt

Bootline commands are executed from the bootline prompt. The bootline prompt is not part of the switch console, and is accessible only after a reboot and before the switch firmware is loaded. Consequently, the bootline commands can be used only through a terminal connection.

Perform the following steps to gain access to the bootline prompt:

1. Connect a dumb terminal (or workstation running terminal emulation software) to the RJ-45 terminal port on the front of your ATM SmartSwitch.
2. Enter the **`reboot`** command from the terminal.
3. Wait for the following message to appear:

   ```
   "Press any key to exit to bootline prompt."
   ```
4. Before the countdown reaches zero, press a key to access the bootline prompt. Notice that the bootline prompt (=>) differs from the prompt used by the switch console.

7.2.2 Bootline Commands Explanations

The following table describes the commands available from the bootline prompt, their use, and their associated parameters.

Table 7-1 Bootline commands

Command	Action	Parameters
chpi	**Change default boot load image:** Sets one of two images of the boot load firmware as the default. Default boot load image is executed at start-up.	**chpi 0** = set boot load image 0 as default **chpi 1** = set boot load image 1 as default
clfs	**Clear flash file system:** Clear flash file system of all switch configuration information.	none
dcfg	**Display boot load configuration:** Displays revision numbers of both boot load images, the switch MAC address, and the file space (in hexadecimal) available for additional MAC addresses. Shows whether POST is set to run at switch start-up.	none
df	**Download Firmware:** Downloads firmware images from a TFTP/Bootp server. Different components of the switch firmware are downloaded, depending on the parameter used with this command.	**df B** = download boot load firmware **df S** = download switch operating firmware **df P** = download diagnostics (POST) **df** (none) = download switch operating firmware
go	**Run switch firmware:** Exit the bootline prompt, and run switch operating firmware.	**go V** = run switch firmware, do not run POST **go P** = run POST before running switch firmware **go** (none) = run switch firmware, do not run POST
he	**Show help:** Displays help for a bootline command or displays list of all bootline commands.	**he** [<command>] = display help for command specified **he** = display list of all bootline commands
ponf	**POST on or off:** Changes start-up action: either run POST before running switch firmware or skip POST and go directly to switch firmware.	**ponf V** = run switch firmware after start-up timeout **ponf P** = run POST before running switch firmware

Table 7-1 Bootline commands (Continued)

Command	Action	Parameters
scsm	**Switch to the redundant CSM:** Tells the SmartSwitch 6500 to transfer CSM mastership to the slave CSM.	none
swms	**Switches CPU mastership to other TSM/CPU:** Changes the slave TSM/CPU to the master.	none

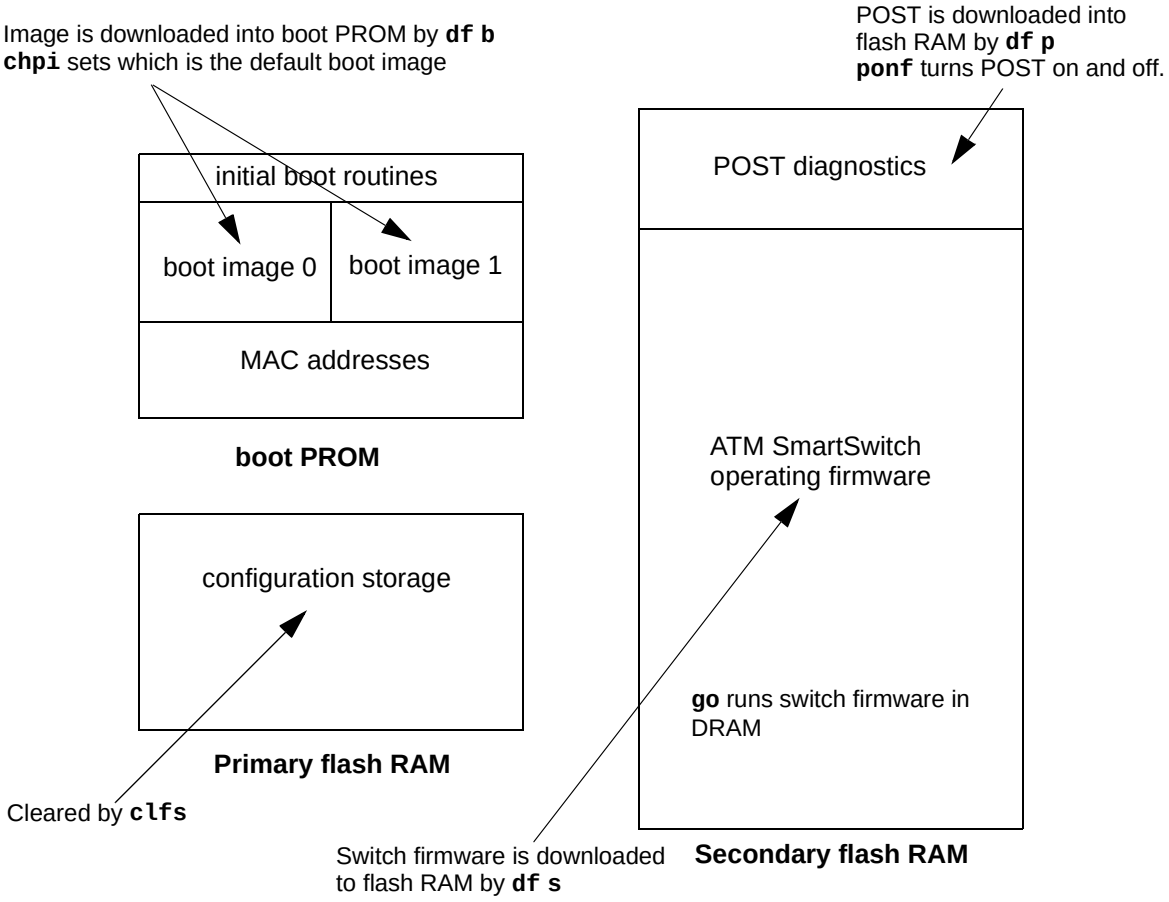

Figure 7-1 Memory locations affected by the bootline commands

7.2.3 Upgrading Boot Load firmware

Two images of the boot load firmware reside in flash RAM. The two images are identified as boot load image 0 and boot load image 1. Both boot load images can be upgraded by using a TFTP/Bootp server. However, an upgrade is always written over the boot load image that is not currently running. This insures that if a boot load upgrade fails, there is still one good boot load image to fall back on.

Follow the steps below to upgrade the switch boot load firmware.

1. Set up the TFTP/Bootp server software on a workstation.
2. Connect both the TFTP/Bootp server and the ATM SmartSwitch to your Ethernet network. Make sure that the TFTP/Bootp server can be reached by the ATM SmartSwitch Ethernet interface.
3. Connect a dumb terminal (or PC running terminal emulation software) to the ATM SmartSwitch Terminal port.
4. Copy the ATM SmartSwitch boot load firmware image into the appropriate location on the TFTP/Bootp server. (In this example, the firmware is copied to c:\tftpboot\images\boot.ima.)
5. Set up the TFTP/Bootp server tables (or equivalent file) with:
   - ATM SmartSwitch MAC address
   - IP address of the ATM SmartSwitch Ethernet interface
   - path to the boot image file on the TFTP/Bootp server
6. From the terminal connection, enter the **`reboot`** command.
7. When the following message appears,

   ```
   "Press any key to exit to bootline prompt."
   ```

   stop the countdown by pressing any key. The bootline prompt (=>) appears on the terminal screen.
8. Enter the **`df B`** command. The ATM SmartSwitch contacts the TFTP/Bootp server and downloads the file into the boot load image location that corresponds to the boot load image not currently running. For example, if boot load image 0 is running, **`df B`** downloads the file into boot load image 1, leaving boot load image 0 untouched.

```
=>df b
You've requested a Boot Load Software download
Are you sure?(Y/N)y
Initializing ethernet...
Starting Bootp...
   Boot file: c:\tftpboot\images\boot.ima
Using TFTP to get bootfile "c:\tftpboot\boot\boot.ima" .
..................................................
.........................................
Validity checks of the Boot Load Software Downloaded file...
All Validity checks OK
Programming downloaded image into Boot Load Software1 area, please wait...
New Boot Load Software programmed successfully.
Modifying Control/Stat field to reflect new image change, please wait...
Control/Stat field programmed successfully.
Please reboot to execute new Boot Load Software
=>
```

9. If the new boot load firmware passes the validity checks, it is marked as the new default image. In the example above, boot load image 1 becomes the new default image.
10. Reboot the ATM SmartSwitch to run the new boot load firmware. Notice that the boot load message at start-up indicates that the ATM SmartSwitch is now loading and running boot load image 1.

Changing the Default Boot Load Image

Continuing with the example above, perform the following steps to set boot load image 0 back to being the default.

1. Reboot the ATM SmartSwitch.
2. When the following message appears

   ```
   "Preparing to run Default Primary Image: 1
   Enter 0 or 1 to override and force one of these primary image sectors to run:"
   ```

   press the zero (0) key. The ATM SmartSwitch loads boot load image 0.
3. Use the **chpi** command to make boot load image 0 the default.

```
=>chpi 0
Old Default Primary Image Number: 1
Erasing Sector in Primary Flash sector4
Programming control/stat info into Primary Flash sector4
New Default Primary Image Number: 0
=>
```

4. Reboot the ATM SmartSwitch. Boot load image 0 is now used as the default image.

```
Preparing to run Default Primary Image: 0
Enter 0 or 1 to override and force one of these primary image sectors to run:
```

7.2.4 Upgrading POST Diagnostic firmware

1. Set up the TFTP/Bootp server software on a workstation.
2. Connect both the TFTP/Bootp server and the ATM SmartSwitch to your Ethernet network. Make sure that the TFTP/Bootp server can be reached by the ATM SmartSwitch Ethernet interface.
3. Connect a dumb terminal (or workstation running terminal emulation software) to the ATM SmartSwitch Terminal port.
4. Copy the ATM SmartSwitch diagnostic firmware image into the appropriate location on the TFTP/Bootp server. (In this example, the firmware is located at c:\tftpboot\images\post.ima.)
5. Set up the TFTP/Bootp server tables (or equivalent file) with:
   - ATM SmartSwitch MAC address
   - IP address of the ATM SmartSwitch Ethernet interface
   - path to the POST file on the TFTP/Bootp server
6. From the terminal connection, enter the **reboot** command.
7. When the following message appears,

   ```
   "Press any key to exit to boot load prompt."
   ```

   stop the countdown by pressing any key. The boot load prompt (=>) appears on the terminal screen.
8. Enter the **df P** command. The ATM SmartSwitch contacts the TFTP/Bootp server and downloads the diagnostic firmware into flash RAM.

```
=>df p
You've requested a POST Software download
Are you sure?(Y/N)y
Initializing ethernet...
Starting Bootp...
   Boot file: c:\tftpboot\images\post.ima
Using TFTP to get bootfile "c:\tftpboot\images\post.ima" .
.....................................................................
```

```
.....................................................................
.....................................................................
.....................................................................
.....................................
Validity checks of POST software Downloaded file...
All Validity checks OK
Programming downloaded image into POST Software section, please wait...
New POST Software programmed successfully
=>
```

9. Check whether the diagnostic download is successful by entering the **`go P`** command. This forces the ATM SmartSwitch to run POST before starting the switch firmware.

7.2.5 Upgrading Switch Operating firmware

Note ATM SmartSwitch operating firmware can also be updated using the switch console **`update firmware`** command (see Section 7.1).

1. Set up the TFTP/Bootp server software on a workstation.

2. Connect both the TFTP/Bootp server and the ATM SmartSwitch to your Ethernet network. Make sure that the TFTP/Bootp server can be reached by the ATM SmartSwitch Ethernet interface.

3. Connect a dumb terminal (or workstation running terminal emulation software) to the ATM SmartSwitch Terminal port.

4. Copy the ATM SmartSwitch operating firmware image into the appropriate location on the TFTP/Bootp server. (In this example, the firmware is located at c:\tftpboot\images\server.ima.)

5. Set up the TFTP/Bootp server tables (or equivalent file) with:

- ATM SmartSwitch MAC address
- IP address of the ATM SmartSwitch Ethernet interface
- path to the operating firmware file on the TFTP/Bootp server

6. From the terminal connection, enter the **`reboot`** command.

7. When the following message appears,

```
"Press any key to exit to bootline prompt."
```

stop the countdown by pressing any key. The bootline prompt (=>) appears on the terminal screen.

8. Enter the **`df s`** command. The ATM SmartSwitch contacts the TFTP/Bootp server and downloads the switch operating firmware into flash RAM.

```
=>df s
You've requested a Switch Software download
Are you sure?(Y/N)y
Initializing ethernet...
Starting Bootp...
   Boot file: c:\tftpboot\images\server.ima
Using TFTP to get bootfile "c:\tftpboot\images\server.ima" .
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
```

```
.................................................
Validity checks of the Switch Software Downloaded file...
All Validity checks OK
Programming downloaded image into Switch Software section, please wait...
New Switch Software programmed successfully
=>
```

9. Start the ATM SmartSwitch by entering the **go** command.

8 ATM FILTERING AND CLOCKING

8.1 PORT ATM ADDRESS FILTERS

SmartSwitch ATM switches support ATM address filtering. Address filtering provides a way to control call setups through SVCs. Filtering is a process of stating whether entities with particular ATM source or destination addresses (or ranges of addresses) are admitted or denied access through a port or set of ports.

Note Address filters can be created that include only a source or destination address. Filters do not necessarily have to specify both addresses.

8.1.1 Creating ATM Address Filters

The process for using ATM address filtering is summarized below

1. Create and name a filter that specifies a source address (or range of addresses) and/or a destination address (or range of addresses) and the action to be taken (admit or deny)
2. Create and name a filter set whose members are existing filters
3. Assign a filter set (by name) to an incoming port and an outgoing port

8.1.2 How ATM Address Filters Work

It's important to understand that a filter set is essentially a set of "IF" statements. When a call is received on a port on which a filter set has been assigned, the call's source address, destination address, or both are compared to the first member of the filter set. If the addresses contained within the call match the addresses of the first filter in the filter set, the specified action is taken (admit or deny). If the addresses do not match, the next filter in the filter set is tested, and so on. Ultimately, if none of the filters apply (no addresses match), no action is taken and the call is allowed to proceed.

8.1.3 ATM Address Filter Example

The following is an example of creating a filter, a filter set, and assigning the filter set to an incoming and outgoing port.

1. Use the **add atmfilter** command to create filters on source and/or destination addresses

```
SmartSwitch # add atmfilter
FilterName(FILTER1)                     : Domain1
Src-ATMAddr()                           : 39:00:00:00:00:00:00:00:00:00:1d:a3:
44:00:1d:a3:44:20:11:00
SrcAddrMask(FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF):
Dst-ATMAddr()                           : 39:00:00:00:00:00:00:00:00:00:1d:b4:
d5:00:1d:b4:d5:14:31:00
DstAddrMask(FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF):
FilterType()                            : deny

SmartSwitch #
SmartSwitch # add atmfilter
FilterName(FILTER2)                     : domain2
Src-ATMAddr()                           : 39:00:00:00:00:00:00:00:00:00:1d:71:
04:00:1d:71:04:55:36:00
SrcAddrMask(FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF):
Dst-ATMAddr()                           : 39:00:00:00:00:00:00:00:00:00:1d:7a:
12:00:1d:7a:12:01:57:00
DstAddrMask(FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF):
FilterType()                            : deny

SmartSwitch #
```

2. Use the **add atmfilterset** command to create a filter set that uses the filters **domain1** and **domain2**

```
SmartSwitch # add atmfilterset
FilterSetName(SET1)                     : Denied_domains
FilterName()                            : domain1
FilterName()                            : domain2
FilterName()                            :     — Press the Enter key when finished specifying filters
Created Filter Set (Denied_domains) With 2 Filters

SmartSwitch #
```

3. Use the **create portfilterset** command to assign the filter set to an incoming and outgoing port.

```
SmartSwitch # create portfilterset
InComingPort()                          : 8a1
OutGoingPort()                          : 8a2
FilteSetName()                          : Denied_domains

SmartSwitch #
```

Once the filter set is assigned to the incoming and outgoing ports, any call setup attempted through ports **8a1** and **8a2** are rejected if they contain the source and destination addresses specified in the filters **domain1** and **domain2**.

Source and Destination Address Masks

When creating an ATM address filter, the **add atmfilter** command prompts for an address mask (**SrcAddrMask** and **DstArrdMask**). When an entity attempts a call through a port, the address masks determines which bits of the addresses presented by the entity are to be compared against which bits of the ATM addresses specified in the filter. This bit-filtering is performed by applying the mask to *both* the call's address and the specified address in the filter.

By setting the mask appropriately, a filter could either admit or deny access to all but a few addresses within a range. For example, if a filter's mask is set to **00:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF**, the filter disregards the first byte when comparing addresses. As another example, if the filter's mask is set to **FC:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:FF**, the filter disregards the last two bits of the first byte of the address (**FC = 11111100**) when comparing addresses.

If a filter's mask is shorter than its corresponding ATM address, the mask starts at the most significant bit, and pads the remaining length (equal to the length of the specified ATM address) with zero bytes (00). For example, if a filter address is specified as **39:00:00:00:00:00:00:00:00:00:14:41:80:00:20:D4:14:41:80:00**, and the mask for that address is specified as **FF:FF:FF:FF:FF:FF:FF:FF:FF:FF**, the SmartSwitch ATM switch treats the mask as **FF:FF:FF:FF:FF:FF:FF:FF:FF:FF:00:00:00:00:00:00:00:00:00:00**.

8.1.4 Filter Considerations Regarding LANE and IP over ATM

It's important to remember that ATM address filters and filter sets cannot restrict communication between clients who are members of the same ELAN. For example, client 1 and client 2 are members of the same ELAN. For some reason it's necessary to restrict client 1 from communicating with client 2. A filter is created and assigned to the port through which client 1 connects the SmartSwitch ATM switch. The filter denies client 1 access to client 2 by rejecting the call set up to client 2. However, once the call fails, client 1 resorts to broadcasting to client 2 through the ELAN's BUS. In turn, the BUS forwards the broadcast packets to client 2 and contact between client 1 and client2 is established.

ATM address filtering under LANE is more effective if the filter denies a client the ability to join an ELAN. In the example above, client 1 could be kept from communicating with client 2 if client 1 first needed to join client 2's ELAN. In this case, a filter is created that denies client 1 the destination of the LANE servers. As a result client 1 cannot join client 2's ELAN and the two are kept from communicating.

ATM address filtering are more effective in an IP over ATM VLAN environment. Clients connect to each other by obtaining address information form the ARP server. Once the address information is obtained, clients connect directly to each other through the switch's ports. Because of the client-to-client connection method of IP over ATM, filter sets assigned to strategic ports, can effectively control (admit or deny) entities attempting to set up calls through the VLAN.

8.2 PORT CLOCK CONFIGURATION

Note The port clock features described below are supported by the SmartSwitch 6500 only.

The SmartSwitch 6500 allows the specifying source of clocking on a per-port basis. The following describes the possible clock modes:

- Local — The port derives its clocking signal from its own oscillator
- Loopback — The port derives its clocking signal from the clock signal transmitted to it from the device (switch, etc.) to which it's attached
- Network — The port derives its clocking signal from a clock signal received on some port of the switch and made available through the backplane to all ports

By default, the clock mode for all SmartSwitch 6500 ports is **local**. Use the **set portclockmode** command to change a ports clocking source. For example, the following sets port **5a3** into loopback mode.

```
SmartSwitch # set portclockmode
PortNumber(ALL)                          : 5a3
PortClkMode(local)                       : loop

SmartSwitch #
```

Note Never configure two connecting port to both be in loopback mode. Without at least one of the connecting ports generating a clocking signal, connectivity will go out of sync and communication will be lost.

8.2.1 Network Clocking

Network clocking allows your SmartSwitch 6500 to obtain an external, high-quality, precise clocking signal and make it available for use by all ports. Typically, network clocking is configured when a high-quality clock signal is available (for example from a service provider connection) and the SmartSwitch is supporting traffic from applications that are time-sensitive, such as voice and video. The port connected to the high-precision clock signal is specified as the network source using the **set networkclock** command. When set, the port is essentially placed in loopback mode, however, the port also places the incoming, high-precision signal on the SmartSwitch 6500's backplane, where it becomes available to all other ports.

The following is an example of network clocking configuration. It is assumed in this example that the SmartSwitch 6500 is connected through port **7a1** to a service provider's switch that produces a high-precision clocking signal.

1. Use the **set networkclock** command to specify the port through which the network clocking signal is to be obtained

```
SmartSwitch # set networkclock
PortNumber(none)                         : 7a1

SmartSwitch #
```

2. Use the **set portclockmode** command to instruct ports (either all ports or on a per-port basis) to use the clocking signal obtained from port **7a1**

```
SmartSwitch # set portclockmode
PortNumber(ALL)                          : — In this example, we set all ports to use the network clock
PortClkMode(local)                       : network

SmartSwitch #
```

Once the **set portclockmode** command is entered with the **PortClkMode** parameter set to **network**, the ports specified on the SmartSwitch 6500 now use the clocking signal received on port **7a1** as their port clock source.

9 TROUBLESHOOTING

This chapter provides basic troubleshooting for diagnosing and fixing problems with VLAN, emulated LANs, PNNI links, and ATM traffic congestion.

9.1 TROUBLESHOOTING IP OVER ATM

You have configured an IP over ATM VLAN, but your network applications are not working. Use these questions and tests to help determine the cause of the problem.

1. Check for connectivity: Try pinging between end nodes and from the ATM SmartSwitch (using **`ping`**) to its end nodes. If you cannot ping, check physical connectivity (disconnected cable and so on).
2. Check IP routes and addresses.

- Use the **`show route`** command to check the ATM SmartSwitch route table.
  - Are the destination addresses correct for the specified gateways?
  - Are there any routing loops?
  - Are one or more of the destination addresses mapped to the wrong subnet?
- Use **`show client`** (ARP server is on the ATM SmartSwitch) to check the local client.
  - Does the client have the correct IP address?
  - Is the subnet correct? Is the ATM address correct?
  - Is the server type correct?
- Check end node configurations.
  - Are end nodes configured correctly?

3. Check ARP statistics.

- Use **`show ipatmarp`** (if the ARP server is on the ATM SmartSwitch).
  - Are there entries in the table?
  - Are the ATM addresses correct?
- Use **`show clientarp`** (ARP server is not on the SmartSwitch) to check local client's ARP Table.
  - Are there entries in the table? If not, recheck client and end node configuration.
  - Are the ATM addresses correct?

4. Check ILMI, UNI routes, and PVCs (if applicable).

- If using SVCs, use **`show ATMRoute`** to check whether static UNI routes are correct and whether dynamic UNI routes are established and correct. If dynamic routes are incorrect or missing, try creating static routes instead.
- If using PVCs, use **`show pvc`** to check if PVCs connect the correct resources through the correct ports.
- If using PVCs, use **`show ipatmpvc`** to check if local switch clients are mapped to the correct end node IP addresses.

5. If working through these questions does not solve the problem, contact Cabletron Systems Customer Service. (see Appendix B, "Technical Support").

9.2 TROUBLESHOOTING LAN EMULATION

You have configured an Emulated LAN and your network applications are not working. Use these questions and tests to help determine the cause of the problem.

1. Check for connectivity. Try pinging between end nodes. Ping from the ATM SmartSwitch (using **`ping`**) to its end nodes. If you cannot ping, check physical connectivity (disconnected cable and so on).
2. Execute the **`show lecs`** command on the switch that contains the LECS. If the LECS is down, start it by executing the **`start lecs`** command.
   - If running distributed LANE services (LECS on one switch and LES and BUS on another switch) execute the **`show les`** command on the switch running the LES and BUS. If the LES and BUS are down, start the LES and BUS by executing the **`start les`** command.
3. Check IP routes and addresses.

- Use **`show route`** command to check the ATM SmartSwitch route table.
  - Are the destination addresses correct for the specified gateways?
  - Are there any routing loops?
  - Are one or more of the destination addresses mapped to the wrong subnet?
- Use **`show client`** to check the ATM SmartSwitch local ELAN client.
  - Does the client have the correct IP address?
  - Is the subnet correct?
  - Is the ATM address correct?
  - Is the server type correct?
- Check end nodes configurations.
  - Are end nodes configured correctly?

4. If the ELAN spans multiple switches, check the following:
   - Is the LECS address correct on all switches?
   - Can all switches reach the switch providing LECS support?
   - If using the Well Known LECS Address, are all switches correctly mapped?
5. Check the LECS database.

- Use **`show lecselan`** to check the names and numbers of ELANs.
  - Are ELAN names correct?
  - Is the ATM address of the LES correct?

6. Check whether LES is connected.

- Use **`show lesclient`** to check whether devices are registered with the LES. If clients are registered, check end node configuration. If not registered, check multi-point signaling.
- Use **`set leselan`** to turn off multi-point signaling on a per-ELAN basis.

- Do devices begin to register with the LES and BUS once multi-point signaling is turned off?

7. Check whether BUS is connected.

- Use **`show busclient`** to check whether devices are registered with the BUS. If clients are registered, check end node configuration. If not registered, check multi-point signaling.
- Use **`set leselan`** to turn off multi-point signaling on a per-ELAN basis.
  - Do devices begin to register with the LES and BUS once multi-point signaling is turned off?
- Check IISP routes to the switch containing the LES and BUS.
  - Are all IISP routes correct?
  - Does a new IISP route need to be added so devices can reach the LES and BUS?

8. If working through these questions does not solve the problem, contact Cabletron Systems Customer Service. (see Appendix B, "Technical Support").

9.3 TROUBLESHOOTING PNNI LINKS

You have physically connected another company's ATM switch with your ATM SmartSwitch. Each switch supports PNNI, but there is no connectivity between the two devices. When dealing with PNNI connectivity, two possible configurations must be considered:

- The ATM SmartSwitch and the other switch are in the same peer group
- The ATM SmartSwitch and the other switch are is different peer groups

Use the following procedures to diagnose and resolve PNNI connectivity problems.

9.3.1 Switches in Same Peer Group

1. Check the physical connection. Make sure that the switches are connected correctly.
2. Check that both switches are in the same peer group. On the ATM SmartSwitch, enter the **`show pnninode`** command to view the peer group ID. If not the same peer group, perform the following:
   - Set the peer group ID on either switch to match the other. On the ATM SmartSwitch, use the **`set pnnipeergroup`** command to change the peer group ID.
3. Check the signalling type of each switch. If either switch does not show PNNI as the signaling type on the connecting port. Perform the following:
   - Turn off ILMI and manually set the signaling type to PNNI. On the ATM SmartSwitch, enter the **`show portconfig`** command to view signaling type for all ports. If necessary, use the **`set portconfig`** command to turn off ILMI and manually set signaling to **`pnni10`**.
4. If none of the above actions have corrected the problem, contact Cabletron Systems Customer Service (see Appendix B, "Technical Support").

9.3.2 Switches in Different Peer Groups

1. Check the physical connection between the peer groups. Make sure that the switches are connected correctly.

2. Make certain that the switches in the other peer group support multi-level PGLs and border nodes. If not, the other switches must be placed in the same peer group as the ATM SmartSwitch if you want them to connect.
3. Are the switches within the peer groups communicating with each other? If not, fix the connectivity problem within the peer group (see Section 9.3.1).
4. Has the Peer Group Leader (PGL) been elected in both groups? If not, start the election process. On the ATM SmartSwitch, use the **`set pnniplgelection`** command to start the PGL election process.
5. Do both peer groups have a parent node (grandparent node, great grandparent, etc.) in a common peer group?
   - If not, create a parent node within a higher-level peer group that's common to both peer groups. On the ATM SmartSwitch, use the **`add pnninode`** command to create the parent node.
   - If they do, contact Cabletron Systems Customer Service (see Appendix B, "Technical Support")

9.4 TROUBLESHOOTING CONGESTION

If the bandwidth of your ATM SmartSwitch begins to decrease, and if connections are being lost or packets are being dropped at a high rate, it's possible that your switch is becoming congested. Congestion can occur on the port level, the global switch level, or both levels.

If you suspect that your ATM SmartSwitch is experiencing congestion, follow the steps outlined below to diagnose and resolve the cause of congestion.

9.4.1 Diagnosing Congestion

1. Enter the **`show portstats`** command, and take the default of (**`all`**).
2. If cells are being dropped only on specific ports, proceed to the "Port Congestion" section.
3. If cells are being dropped on all ports, the indication is global congestion. Proceed to the "Global Congestion" section.

9.4.2 Global Congestion

1. Is the total cell drop rate equal to the Unknown VC cell drop rate?

- If yes, the switch is improperly set up. Check the switch configuration.
- If no, this indicates global congestion. Continue.

2. Set the **`porttrafficcongestion`** values to those recommended in the table below.

Table 9-1 Settings for Class of Service Queues

Service Class	Recommended Settings
CBR	Fewer than 100 connections on a port: Min = 64, Max = 1024
CBR	More than 100 connections on a port: Min = 128, Max = 1024

Table 9-1 Settings for Class of Service Queues (Continued)

Service Class	Recommended Settings
rt-VBR	Bandwidth* utilization less than 20%: Min = 16, Max = 1024
rt-VBR	Bandwidth* utilization greater than 20%: Min = 128, Max = 4096
Nrt-VBR	Min = 256, Max = 4096
UBR	Min = 256, Max = 8192
ABR	Min = 256, Max = 8192

*Use the **`show portconfig`** command to view bandwidth utilization

3. Has the congestion subsided?

- If yes, you are done.
- If no, continue.

4. Have you changed the EPD threshold (**`set switchtrafficcongestion`** command)?

- If yes, replace it to the default setting. If congestion subsides, you are done.
- If no, continue.

5. Enter the **`show cacinfo`** and **`show portconfig`** commands for each port. Is the allocated bandwidth small and is the traffic mostly UBR?

- If no, go back to step 4 and check next port.
- If yes, continue.

6. Enter the **`show porttrafficcongestion`** command. Is the UBR queue **`MaxValue`** large?

- If no, go back to step 4.
- If yes, continue.

7. Reduce the UBR queue **`MaxValue`** by a small amount, then wait a few minutes.

8. Enter the **`show portstats`** command, and take the default of **`all`**. Is the number of cells dropped increasing for this port, and quickly decreasing for all other ports?

- If yes, proceed to the “Port Congestion” section.
- If no, continue.

9. Is the number of cells being dropped by all other ports decreasing somewhat?

- If no, go back to step 6.
- If yes, continue.

10. Enter the **`set caceqbwallocscheme`** command and set call admission control for this port to a more conservative policy (**`moderate`** or **`conservative`**).

11. Go back to step 4 until all ports have been checked.

9.4.3 Port Congestion

1. Enter the **`show portstats`** command a few times, noting the value for cells dropped and unknown VCs dropped. Is the number of cells dropped equal to the number of VCs dropped?

- If yes, the switch is improperly set up. Check the switch configuration.
- If no, this indicates port congestion. Continue.

2. Enter the **`show cacinfo`** command for this port. Note the bandwidth allocated for each Quality of Service on this port.
3. For each class of service, enter the **`set porttrafficcongestion`** command. Set the **`MaxValue`** to the value recommended in Table 9-1, “Setting for Class of Service Queues.”
4. Have you performed step 3 for every class of service for this port?

- If no, go to step 3.
- If yes, continue.

5. Enter the **`set caceqbwallocscheme`** command for this port. Set call admission control for this port to a more conservative policy (**`moderate`** or **`conservative`**).
6. Check VC statistics for this port using either the **`show pvc /d`** or **`show svc /d`** command, whichever is appropriate. If the port belongs to the high virtual channel link (VCL), read the forward statistics. If the port belongs to the low VCL, read the backward statistics. If the port belongs to both high and low VCLs, read both statistics.
7. Is the number of cells received increasing?

- If no, go through step 6 a few more times. If cells received still do not increase and congestion persists, contact Cabletron Customer support.
- If yes, continue.

- Enter the **`show cacinfo`** command for this port. Is the Allocated Bandwidth less than the Cell Reception Rate obtained from **`show pvc /d`** or **`show svc /d`** in step 6?
- If no, go through step 6 a few more times. If cells received still do not increase and congestion persists, contact Cabletron Customer support.
- If yes, this VC is misbehaving. Take appropriate action, for example, terminate the VC.

9.5 EVENTS AND ALARMS

ATM SmartSwitches record and report their operation in real-time through the use of events and alarms. An event is an occurrence of a significant activity. For instance, a port going down or a client joining an ELAN are examples of events. Alarms are a specific class of events defined as “events that the user needs to know about or attend to immediately.” Alarms do not always indicate switch faults. Alarms may also be informational events. For instance, “LECS Operational” is an example of an alarm that is not a switch fault, but is an activity that the user should know about immediately.

9.5.1 Event Categories

Events are grouped into the following categories:

- Critical — Impacts the entire switch, leaving the system unavailable or in a degraded state
- Major — Impacts a feature of the switch, leaving the feature unavailable or in a degraded state
- Minor — Impacts the system or feature, leaving it in a sub-optimal state
- Informational — An occurrence of an activity that the user should know about

Both events and alarms are stored within circular memory buffers. When the buffers become full, older events and alarms are overwritten by newer entries. Both events and alarms are stored in shared RAM. However, the 40 most recent alarms are also stored in flash RAM. Storing these 40 alarms in flash RAM makes them persistent between reboots of the ATM SmartSwitch, and provides information about the state of the switch prior to reboot.

Note Alarms are collected and stored in flash RAM in groups of four. As a result, some of the most recent alarms may not be persistent. For example, there are 24 (6 times 4) alarms stored in flash RAM. If a 25th alarm occurs, and the switch is rebooted, only the 24 alarms are persistent. The 25th alarm is dropped because the number of alarms (after 24) did not reached the next multiple of four (28).

9.5.2 Viewing Events and Alarms

Use the **`show events`** command to view a list of the currently logged events. For example,

```
SmartSwitch # show events
Index(ALL)                                  :
0  33554474  MAJOR EVENT        000:00:04:311
-------------------------------------------------
 LES ReadServerConfig:  Unable to open config file les.db

1  33554653  INFO EVENT        000:00:04:320
-------------------------------------------------
LECS Database non existing - creating default ELAN

2  117571585  MINOR EVENT        000:00:07:341
-------------------------------------------------
SAAL connection has become active, initiated locally
Port ID 0x01c41000
Protocol 0x02

3  117571585  MINOR EVENT        000:00:07:585
-------------------------------------------------
SAAL connection has become active, initiated locally

More(<space>/q)?:
```

Events are displayed in the following format:

- Event number — The index number of the event in the circular buffer
- Event ID — A unique ID assigned to the event
- Category — Whether this event is critical, major, minor, or informational
- Time — Time of event, in switch up-time in hours, minutes, seconds, and milliseconds
- Object — The object affected by the event (port, LEC, and so on)
- Description — Brief message describing the event

Event messages can be automatically displayed on the ATM SmartSwitch console. Use the **`set eventdisplay`** command to display events on the console as they occur:

```
SmartSwitch # set eventdisplay
EventDisplay(OFF)                          : on
SmartSwitch #
```

Note Depending on the activity of your ATM SmartSwitch, the appearance of events on the ATM SmartSwitch may be too frequent to use the console comfortably. It is recommended that you turn on the automatic display of events only when troubleshooting.

Use the **show alarms** command to view a list of the currently logged alarms. For example,

```
SmartSwitch # show alarms
Index(ALL)                                :
0  33554702  000:07:05:300
-------------------------------------------------
pvcm_cac_admit: failed  501037


1  33554652  023:56:23:317
-------------------------------------------------
LECS Operational

2  117506049  024:01:54:083
-------------------------------------------------
Failed to re-establish SAAL connection
Port ID 0x01c81000
T309    10000

3  117506049  024:01:54:430
-------------------------------------------------

More(<space>/q)?:
```

Alarms are displayed in the following format:

- Alarm number — The index number of the alarm in the circular buffer
- Alarm ID — A unique ID assigned to the alarm
- Time — Time of alarm, in switch up-time in hours, minutes, seconds, and milliseconds
- Object — The object affected by the alarm (port, LEC, and so on)

Alarm messages can be automatically displayed on the ATM SmartSwitch console. Use the **set alarmdisplay** command to display alarms on the console as they occur:

```
SmartSwitch # set alarmdisplay
alarmDisplay(OFF)                         : on
SmartSwitch #
```

9.5.3 Deleting Events and Alarms

To delete events or alarms currently logged within your ATM SmartSwitch, use the **delete events** and **delete alarms** commands, respectively.

9.6 SAVING CORE DUMPS

The ATM SmartSwitch core dump feature allows you to specify a local Ethernet host where, in the event of a system failure, the ATM SmartSwitch sends a copy of its memory. ATM SmartSwitch system memory is saved to two files, one containing CPU memory (`core_cpu`), the other common memory (`core_cmn`). These files can then be sent to Cabletron customer support for analysis.

Note To use the core dump feature, the local Ethernet host must be running TFTP server software, and you must have write access to the TFTP directory.

Enter the **`set coredump`** command to enable the core dump feature. For example,

```
SmartSwitch # set coredump
EnableCoreDump(n)                : y                    — "y" to enable core dump feature
ServerIP()                       : 204.95.77.240        — IP address of my TFTP server
CoreDumpFile()                   : /tftpboot/bobr/core  — full path name for core dump files
userName()                       : bobr                 — login name on the server
UserPassword()                     :                    — password
SmartSwitch #
```

Note The **`set coredump`** command uses FTP to create the `core_cpu` and `core_cmn` files. If your server does not run FTP, create these files manually. Then execute the **`set coredump`** command.

Note On UNIX systems, make sure that the permissions are set correctly so that data can be written.

Note For security, the **`set coredump`** command retains your password only long enough to create the core dump files. Your password is then dropped from system memory.

To see the current core dump configuration, enter the **`show coredump`** command.

```
SmartSwitch # show coredump

Core Dump Enabled   : Yes
Core Dump Server IP : 204.95.77.240
Core Dump File      : /tftpboot/bobr/core

SmartSwitch #
```

If a system failure occurs while the core dump feature is enabled, the ATM SmartSwitch console appears similar to the example below. The ATM SmartSwitch then begins sending images of its memory to the core dump files on the TFTP server.

```
Illegal access. Bus Error.
IP: e0103288    PFP: e04be080
r0(pfp): e04be040         r1(sp): e04be0c0        r2(rip): e00dd7dc
r3     : 00000000         r4    : e00f8f0c        r5     : e0409f10
r6     : 00000003         r7    : e00f8f0c        r8     : e0409f40
r9     : 00000003         r10   : 00000030        r11    : e00f8f0f
r12    : 00000008         r13   : 00000001        r14    : e00d22f0
r15    : 00000008
d2000000:  Core Dump
Common DRAM dumped to /tftpboot/bobr/core_cmn
CPU DRAM dumped to /tftpboot/bobr/core_cpu
ffffffff ffffffff ffffffff ffffffff   *................*
d2000010:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000020:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000030:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000040:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000050:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000060:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000070:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000080:  ffffffff ffffffff ffffffff ffffffff   *................*
d2000090:  ffff
SmartSwitch Start-up Code
Cabletron Systems Inc.
```

Copy the information displayed on the console and send it to your Cabletron customer support representative along with the core dump files. (See Appendix B, "Technical Support")

APPENDIX A AGENT SUPPORT

This appendix briefly describes the support provided for managing an ATM SmartSwitch using Simple Network Management Protocol (SNMP).

A.1 MIB, SMI, MIB FILES AND INTERNET MIB HIERARCHY

A MIB (Management Information Base) is the term used to represent a virtual store of management data on a device. Given the structure of management data, it can be operated upon (retrieved, created or modified) using the SNMP protocol. The structure of that data is defined using a subset of a notation called Abstract Syntax Notation (ASN.1). This subset is called SMI (Structure of Management Information). A file containing the definition of that structure is called a MIB file. To provide for a uniform naming convention for all MIBs, from all vendors, for all kinds of data, a standard format is used. This format is a hierarchy and is termed the Internet MIB Hierarchy.

The MIB structure is logically represented by a tree hierarchy (see Figure A-1). The root of the tree is unnamed and splits into three main branches: Consultative Committee for International Telegraph and Telephone (CCITT), International Organization for Standardization (ISO), and joint ISO/CCITT.

These branches and those that fall below each category have short text strings and integers to identify them. Text strings describe object names, while integers allow computer software to create compact, encoded representations of the names. For example, the ZeitNet MIB variable znIpAtmClient is an object name and is also represented by the number one.

An object identifier in the Internet MIB hierarchy is the sequence of numeric labels on the nodes along a path from the root to the object. The object for the Internet Standard for MIB II is represented by the object identifier 1.3.6.1.2.1. It also can be expressed as iso.org.dod.internet.mgmt.mib (see Figure A-1).

Note For the authoritative reference on the concepts described in this section, refer to RFCs 1901 through 1908.

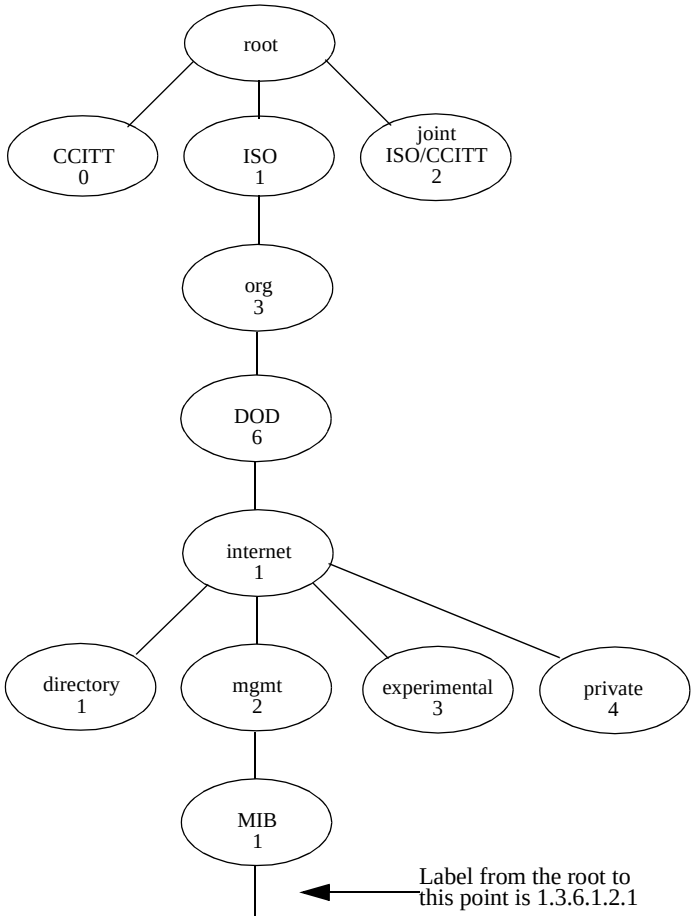

Figure A-1 Internet MIB hierarchy

A.1.1 CSI ZeitNet Proprietary MIBs

The location of some of ZeitNet proprietary MIBs in the Internet hierarchy is shown in Figure A-2. All nodes starting with “zn” represent Zeitnet objects.

The private ZeitNet MIB is represented by the object identifier 1.3.6.1.4.1.1295, or iso.org.dod.internet.private.enterprise.zeitnet. The ZeitNet proprietary MIBs include the subtrees shown in Figure A-2.

.

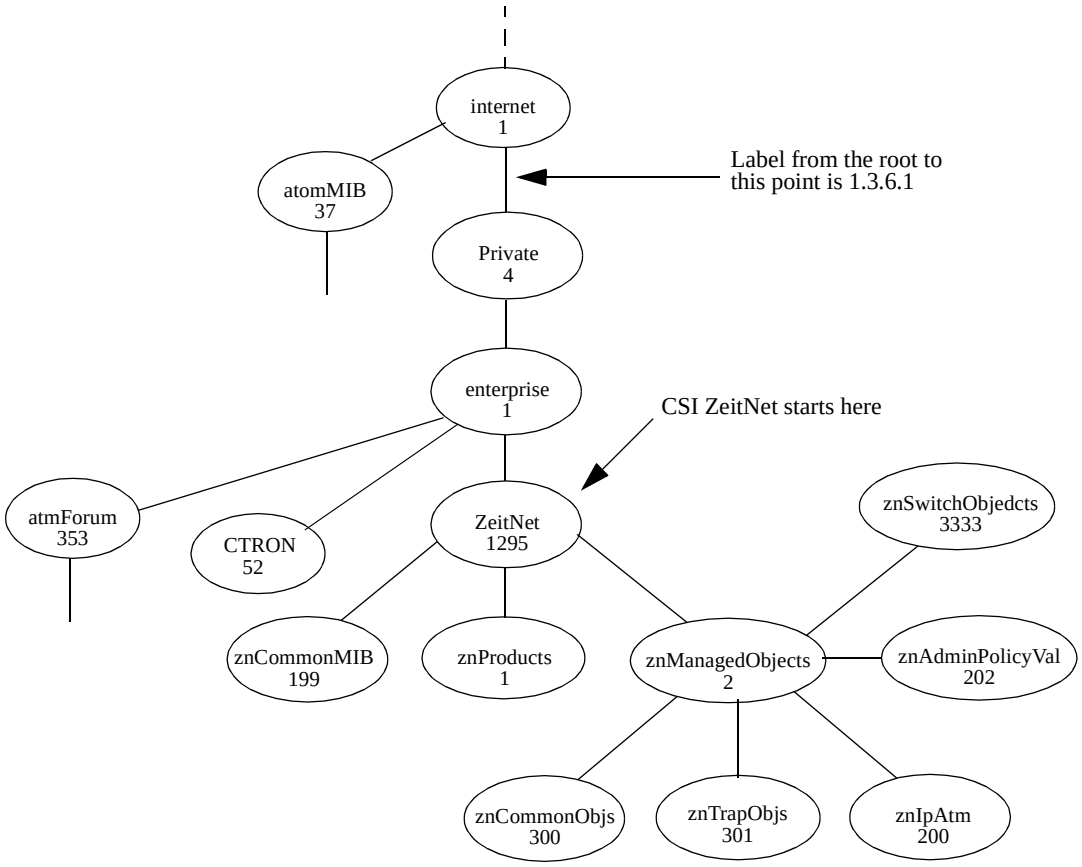

Figure A-2 CSI ZeitNet Private MIBs

In Figure A-2, the ZeitNet proprietary group is identified by 1.3.6.1.4.1.1295; its subgroup, called znProducts, is identified by 1; and the first variable is znManagedObjects with a value of 2. Therefore, the object znManagedObjects has an object identifier of 1.3.6.1.4.1.1295.2.

A.1.2 Relation Between Object Identifier and the Represented Value

In Figure A-3, the znLec object (representing LAN Emulation Client information) has an Object Identifier of 1.3.6.1.4.1.1295.2.3333.9.1.1. The znLecDDCount object representing the number of Data direct connections maintained by one LEC (Lan Emulation Client) has a object identifier of 1.3.6.1.4.1.1295.2.3333.9.1.1.1.1. Querying for the value represented by this object identifier (using the SNMP protocol), returns the actual number of data direct connections for the identified LEC.

:

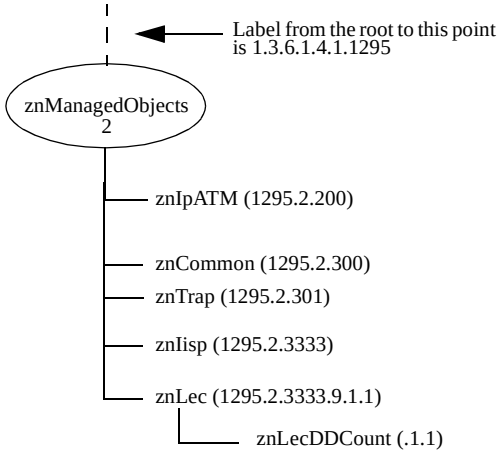

Figure A-3 Cabletron ATM SmartSwitch object identifier example

A.1.3 Supported protocols

All ATM SmartSwitches support Simple Network Management Protocol (SNMP). Both the SNMPv1 and SNMPv2c formats of the protocol are supported.

A.1.4 Supported SMI Formats

Cabletron Zeitnet proprietary MIBs are defined using SNMPv2c format of the SMI.

A.1.5 CSI ZeitNet Proprietary MIB Groups

The following table of CSI Zeitnet proprietary MIB groups lists group name, object identifier, and group function.

Table A-1 CSI Zeitnet proprietary MIB groupings

Name	Object Identifier	Function
zeitnet	1.3.6.1.4.1.1295	All Zeitnet Proprietary Objects
znProducts	1.3.6.1.4.1.1295.1	ZeitNet product specific
znManagedObjects	1.3.6.1.4.1.1295.2	Various classes of Managed entities
znIpAtm	1.3.6.1.4.1.1295.2.200	IP ATM services
znIpAtmClient	1.3.6.1.4.1.1295.2.200.1	IP ATM Client Services

Table A-1 CSI Zeitnet proprietary MIB groupings (Continued)

Name	Object Identifier	Function
znIpAtmServer	1.3.6.1.4.1.1295.2.200.2	IP ATM Server Services
znCommonObjs	1.3.6.1.4.1.1295.2.300	Zeitnet Specific Information
znTrapObjs	1.3.6.1.4.1.1295.2.301	ZeitNet Traps
znSwitchObjects	1.3.6.1.4.1.1295.2.3333	Switch/hardware specific information
znSystem	1.3.6.1.4.1.1295.2.3333.1	Hardware and software system level information
znSwitchDiscoveryTable	1.3.6.1.4.1.1295.2.3333.1.34	Neighbor switch configuration
znConfig	1.3.6.1.4.1.1295.2.3333.2	Switch software configuration management.
znModule	1.3.6.1.4.1.1295.2.3333.3	Switch Module information.
znPort	1.3.6.1.4.1.1295.2.3333.4	Switch Port Information.
znPortTrafficCongTable	1.3.6.1.4.1.1295.2.3333.4.3	Traffic management
znSignalling	1.3.6.1.4.1.1295.2.3333.5	Signalling timer information
znSar	1.3.6.1.4.1.1295.2.3333.8	SAR specific information.
znVlan	1.3.6.1.4.1.1295.2.3333.9	Zeitnet Lane Services Group
znLanEmulation	1.3.6.1.4.1.1295.2.3333.9.1	Zeitnet LAN Emulation Group
znLec	1.3.6.1.4.1.1295.2.3333.9.1.1	LAN Emulation Client Specific
znLes	1.3.6.1.4.1.1295.2.3333.9.1.2	Lan Emulation Server Specific
znBus	1.3.6.1.4.1.1295.2.3333.9.1.3	Broadcast and Unknown Server information.
znLecs	1.3.6.1.4.1.1295.2.3333.9.1.4	Lan Emulation Configuration Server Info
znSSCOP	1.3.6.1.4.1.1295.2.3333.12	SSCOP Configuration
znEventTable	1.3.6.1.4.1.1295.2.3333.13.2	Event table
znEventAlarmTable	1.3.6.1.4.1.1295.2.3333.13.5	Alarm table
znTrafficDescrExtTable	1.3.6.1.4.1.1295.2.300.13	Proprietary extensions to atmTrafficDescrParamTable
znCacStats	1.3.6.1.4.1.1295.2.3333.4.5	CAC Statistics Group
znSwitchHW	1.3.6.1.4.1.1295.2.3333.14	Hardware Characteristics of the Switch Group
znSlotTable	1.3.6.1.4.1.1295.2.3333.14.4	Table of I/O Slots
znCpuPortTable	1.3.6.1.4.1.1295.2.3333.14.13	Table of CPU Ports
znIOModuleTable	1.3.6.1.4.1.1295.2.3333.14.15	Table of I/O Modules
znPortExtTable	1.3.6.1.4.1.1295.2.3333.14.10	Extensions to znPortTable
CTRON	1.3.6.1.4.1.52.4.1.	Cabletron Enterprise-specific Container MIB

A.1.6 ATM SmartSwitch MIB Support

The ATM SmartSwitch is shipped with the following MIBs:

- MIB II (RFC 1213)
- Interface Table MIB (RFC 1573)
- AToM MIB (RFC 1695)
- AToM2 MIB
- LANE MIB (ATM Forum)
- ILMI 4.0 MIB (ATM Forum)
- PNNI MIB (ATM Forum)
- IP over ATM MIB
- ATM SmartSwitch MIBs (proprietary)
- Soft PVC MIB

Note Along with the MIBs, the CD-ROM also contains a README file and the release note.

A.1.7 MIB Exceptions

With the current implementation of MIB files, conformance to ATM standards for the ATM SmartSwitch includes the following exceptions.

Non-Conformance

- atmInterfaceIlmiVpi — Read-only
- atmInterfaceIlmiVci — Read-only
- aal5VccTable — Not supported
- atmSvcVcCrossConnectRowStatus Set — Not supported
- atmConfigSigType — The values given below are not supported:
  - ituDss2
  - atmfBici2Dot0
- znIpAtmClientDDVcType — Accepts only pvc(2) in sets
- lecMulticastSendType — Accepts only best effort (1)
- lecMulticastSendAvgRate — Accepts values only up to 370370
- lecMulticastSendPeakRate — Accepts values only up to 370370
- leArpEntryType — Accepts only staticVolatile (4) and staticNonVolatile (5)
- lesControlTimeout — Read-only
- atmTrafficDescrParamIndexNext — Not supported
- atmVplCastType — The values given below are not supported:

  - p2mpRoot
  - p2mpLeaf

- atmVplReceiveTrafficDescrIndex — Doesn't accept ABR traffic descriptor
- atmVplTransmitTrafficDescrIndex — Doesn't accept ABR traffic descriptor

Not Supported

The following MIB objects are not supported. If used, these objects return either the value zero or the message, "Not supported."

- atmInterfaceDs3PlcpTable
- atmInterfaceTCTable
- atmSvcVpCrossConnectTable
- atmSigSupportTable
- atmSigDescrParamTable
- atmIfAdminAddrTable
- atmVclAddrBindTable
- atmAddrVclTable
- atmVclGenTable
- atmfMyOsiNmNsapAddress
- lecRouteDescrTable
- leRDArpTable

A.2 MANAGING AN ATM SMARTSWITCH

Your ATM SmartSwitch must be IP reachable by the NMS before it can be managed. The default connection between the ATM SmartSwitch and the NMS is the Ethernet interface of the ATM SmartSwitch. Use the **`show switchconfig`** command to find the IP address of the ATM SmartSwitch. An NMS can use this IP address to reach the ATM SmartSwitch through Ethernet. An NMS can also manage an ATM SmartSwitch through one of its ATM ports if the ATM SmartSwitch has a client connection into a VLAN or emulated LAN.

Note that the ATM SmartSwitch itself, is not reachable through ATM until a client for the switch is created and participates as a member of a VLAN or ELAN. Your NMS uses that switch client's address to access and manage the switch.

To create a client for the switch, use the **`add ipatmclient`** command for VLANs and **`add laneclient`** for emulated LANs.

Use the **`set mynmaddr`** command to tell the ATM SmartSwitch which interface to use when communicating with your NMS. For detailed information about these commands, see the SmartSwitch ATM Reference Manual.

A.2.1 Console Commands that Affect the Agent

The following is a list of the console commands that affect the operation of the ATM SmartSwitch SNMP agent. For detailed descriptions of these commands, see the SmartSwitch ATM Reference Manual.

- Community: Sets the community strings for the ATM SmartSwitch
- TrapCommunity: Specifies the NMS to which traps are sent
- MyNMAddr: Specifies the IP address through which the switch is managed
- TrustedNMS:Specifies the IP address of the NMS allowed to perform the following commands:
  - **`update firmware`**
  - **`backup`**
  - **`restore`**
  - **`reboot`**

A.2.2 Default Community Strings

The following is a list of the default community strings used by the ATM SmartSwitch:

- public — Used for all standard SNMP communication
- ILMI — Used by ILMI channels between switches
- zeitnet — Used by the SmartSwitch ATM Administrator program

Caution If the community string *zeitnet* is changed on the ATM SmartSwitch it must also be changed at the SmartSwitch ATM Administrator. Failure to do so, makes the ATM SmartSwitch unreachable by the SmartSwitch ATM Administrator program.

APPENDIX B TECHNICAL SUPPORT

This appendix tells you what to do if you need technical support for your ATM SmartSwitch.

Cabletron offers several support and service programs that provide high-quality support to our customers. For technical support, first contact your place of purchase. If you need additional assistance, contact Cabletron Systems, Inc. There are several easy ways to reach Cabletron Customer Support and Service.

B.1 TELEPHONE ASSISTANCE

Our Technical Support Center is available Monday through Friday, 8am to 8pm Eastern Time, by calling 603-332-9400.

B.2 FAX SERVICE

You can fax support questions to us any time at 603-337-3075.

B.3 ELECTRONIC SERVICES

You can contact Cabletron's Bulletin Board Service by dialing 603-335-3358.

Our internet account can be reached at support@ctron.com.

You can also check our home pages on the World Wide Web.

- http://www.Cabletron.com
- http://www.ctron.com

B.4 PLACING A SUPPORT CALL

To expedite your inquiry, please provide the following information:

- Your Name
- Your Company Name
- Address
- Email Address
- Phone Number
- FAX Number
- Detailed description of the issue (including history, what you've tried, and conditions under which you see this occur)

- Hardware model number, software version, and switch configuration (that is, what part types are in what slots)

B.5 HARDWARE WARRANTY

Cabletron warrants its products against defects in the physical product for one year from the date of receipt by the end user (as shown by Proof of Purchase). A product that is determined to be defective should be returned to the place of purchase. For more detailed warranty information, please consult the Product Warranty Statement received with your product.

B.6 SOFTWARE WARRANTY

Cabletron software products carry a 90-day software warranty. During this period, customers may receive updates and patches for verified, reported software issues.

B.7 REPAIR SERVICES

Cabletron offers an out-of-warranty repair service for all our products at our Santa Clara Repair Facility. Products returned for repair will be repaired and returned within 5 working days. A product sent directly to Cabletron Systems, Inc. for repair must first be assigned a Return Material Authorization (RMA) number. A product sent to Cabletron Systems, Inc., without an RMA number displayed outside the box will be returned to the sender unopened, at the sender's expense.

To obtain an RMA number, contact the Cabletron Technical Support. When you call for an RMA number, your support representative will spend a few minutes with you, making sure the board is defective. Once they confirm the board is defective, they will assign an RMA number. Payment, shipping instructions, and turnaround time will be confirmed when the RMA number is assigned.

INDEX

A

accessing the boot load prompt 7-3
address filters 8-1
    example 8-2
address masking 8-2
administrative weight 3-9
agent support A-1
aggregation tokens 3-10
alarm categories 9-7
alarms 9-6
    deleting 9-8
allocating queue buffers 6-5
ARP server 2-2
ATM address filter sets 8-1
ATM address filters 8-1
    address masking 8-2
    ARP server 8-3
    BUS multicast 8-3
    creating 8-1
    example 8-2
    IP over ATM 8-3
    LANE 8-3
    process 8-1
atmroute command 4-1
Available VPIs 5-8

B

bandwidth on class of service 6-4
Base VPI 5-8
Best Effort 2-11
Boot Load Commands
    chpi 7-4
    clfs 7-4
    dcfg 7-4
    df 7-4
    go 7-4
    he 7-4
    memory affected by 7-5
    ponf 7-4
    scsm 7-5
    swms 7-5
boot load firmware 7-6
boot load prompt 7-3
BUS 2-5
    logical multicasting 2-14
    physical multicasting 2-14

C

Cabletron technical support B-1
CAC 6-3
    allocating bandwidth 6-4
    conservative 6-3
    liberal 6-3
    moderate 6-3
CAC policies
    defined 6-3
CAC policy by class of service 6-4
call admission control policies
    defined 6-3
Call Admission Control policy 6-3
CCITT A-1
cell marking 6-7
changing default boot load image 7-7
chpi 7-4
class of service CAC policy 6-4
class of service queue buffers 6-5
clfs 7-4
commands
    add atmfilter 8-2
    add atmfilterset 8-2
    add atmroute 4-1, 4-5
    add elan 2-4, 2-12
    add ipatmclient 2-1, 5-4
    add ipatmpvc 5-4
    add laneclient 2-4
    add lecselanlec 2-11, 2-14
    add lecselannametable 2-12
    add lecselanpolicy 2-13
    add lecsneighbor 2-22
    add lecspacketsize 2-12
    add lecstlvset 2-14
    add pnninode 3-5
    add port 5-10
    add pvc 5-2, 6-3
    add route 4-9
    add spvc 5-14

add spvcaddress 5-13, 5-16
add spvp 5-16
add trafficdescriptor 6-3
create portfilterset 8-2
reboot 7-2
set caceqbwallocscheme 6-4
set cacserviceclassbw 6-4
set coredump 9-9
set eventdisplay 9-7
set linkmonitortimeout 3-12, 4-4, 4-6
set lnniinfo 2-20
set lnnistatus 2-20, 2-24
set networkclock 8-4
set pnniinterface 3-10
set pnnipeergroupid 3-4
set pnnipglelection 3-5
set portclockmode 8-4
set portconfig 4-2, 4-6, 5-5, 5-9
set porttrafficcongestion 6-6
show alarms 9-8
show atmroute 4-2, 4-6
show caceqbwallocscheme 6-4
show client 2-2, 2-5
show events 9-7
show ipatmarp 2-2
show lecselanpolicy 2-13
show lecsneighborinfo 2-22
show lecsserverlist 2-25
show lnnistatus 2-20
show minmax 6-6
show netprefix 4-1
show pnniinterface 3-9
show pnnilink 3-5
show pnnimetric 4-8
show pnninode 3-2, 3-3
show pnnipglelection 3-6
show porttrafficcongestion 6-5
show pvp 5-6
show route 4-10
show spvc 5-14
show spvcaddress 5-13
show spvctarget 5-14
show spvp 5-17
show spvptraget 5-16
show switchtrafficcongestion 6-7
show trafficdescriptor 5-2, 6-2
start lecs 2-4
that affect the agent A-7
update firmware 7-1, 7-8
community A-8
congestion management
diagnosing congestion 9-4
global congestion 9-4
port congestion 9-5
troubleshooting 9-4
connecting PVPs 5-7
core dump files 9-9
core dump security 9-9
core dumps 9-9
creating a soft PVC 5-12
creating a VLAN 2-1
CSI ZeitNet MIB A-1
CSI ZeitNet proprietary MIBs A-2

D

dcfg 7-4
default client address 2-3
default ELAN 2-4
default IP over ATM client 2-3
default LECID 2-20
default netprefix 2-3
deleting events and alarms 9-8
destination type
any 5-14, 5-16
required 5-14, 5-16
df 7-4
diagnosing congestion 9-4
distributed LANE services 2-9

E

EFCI 6-7
ELAN 2-4
default 2-4
over WANs 2-14
ELAN join policies 2-11
ELAN policy
adding a policy 2-13
Best Effort 2-11, 2-12
By ATM Address 2-12
By ELAN Name 2-12
By LAN Type 2-12
By MAC Address 2-12
By Packet Size 2-12
By Route Descriptor 2-12

identifying clients 2-14
index number 2-13
priority value 2-13
ELANs across multiple switches 2-8
Emulated LAN 2-4
enabling EFCI marking 6-7
enabling RM cell marking 6-7
EPD 6-7
EPD threshold 6-7
event categories 9-6
event persistence 9-6
event queue 9-6
events 9-6
deleting 9-8
events and alarms 9-6
viewing 9-7
exterior route 4-1

F

filter mask
destination 8-2
filter masks
source 8-2
filter sets 8-1
filters 8-1
address masking 8-2
creating 8-1
example 8-2
firmware 7-1

G

go 7-4

H

hardware warranty B-2
he 7-4

I

IISP
controlling fail-over timing 4-4
fail-over timing 4-4
route type parameter 4-1
IISP link timing 4-4
IISP routes 4-1
IISP routing considerations 4-2
IISP routing example 4-3
ILMI 4-6
over PVPs 5-6
ILMI 4.0 5-9
internet MIB hierarchy A-1
IP over ATM 2-1, 9-1
ARP server 2-1, 2-2
ARP table 2-2
ATM address filters 8-3
ATM addressing 2-3
client 2-2
creating VLAN 2-1
viewing ARP table 2-2
IP over ATM client 2-3
IP Routing 4-9
IPATM
ATM address filters 8-3
ISO/CCITT A-1

L

LAN emulation
across multiple switches 2-8
add an ELAN 2-4
adding a client 2-4
ATM addressing 2-6
BUS 2-5
creating an ELAN 2-4
default ELAN 2-4
distributed LANE services 2-9
ELAN join policies 2-11
LES 2-5
starting the LECS 2-4
switch clients 2-9
LANE 2-1, 9-2
ATM address filters 8-3
over PVPs 2-14
tunneling 2-14
LANE over WAN circuits 2-14
LANE service 2-4
distributed 2-9
LECID 2-20
default 2-20
LECS 2-4
adding neighbors 2-22
LNNI configuration 2-19
LECSELANLEC table 2-13
LES 2-5

LES/BUS
connectivity 2-19
LES/BUS load sharing 2-17
LGN 3-3
link timing 3-11, 4-4, 4-6
LNNI 2-16
configuring 2-19
distributed LES/BUS servers 2-22
full-mesh topology 2-19
LANE service redundancy 2-16
LECID 2-20
LECS 2-16
LES/BUS 2-23
load sharing 2-17
locally attached LES 2-25
multiple LECS 2-19
neighbor LECS 2-22
SCSP 2-22
SMS servers 2-25
load sharing 2-17
local port clocking 8-3
logical group node 3-3
logical link 3-6
logical multicasting 2-14, 2-15
loopback port clocking 8-3

M

MaxIndex 6-6
MaxVpiBits 5-2, 5-5, 5-8, 5-14
metrics 4-7
MIB
CSI ZeitNet proprietary A-2
exceptions A-6
object identifier A-3
zeitnet A-1
MIB exceptions A-6
MIB groupings
CTRON A-5
zeitnet A-4
znBus A-5
znCacStats A-5
znCommonObjs A-5
znConfig A-5
znCpuPortTable A-5
znEventAlarmTable A-5
znEventTable A-5
znIOModuleTable A-5
znIpAtm A-4
znIpAtmClient A-4
znIpAtmServer A-5
znLanEmulation A-5
znLec A-5
znLecs A-5
znLes A-5
znManagedObjects A-4
znModule A-5
znPort A-5
znPortExtTable A-5
znPortTrafficCongTable A-5
znProducts A-4
znSar A-5
znSignalling A-5
znSlotTable A-5
znSSCOP A-5
znSwitchDiscoveryTable A-5
znSwitchHW A-5
znSwitchObjects A-5
znSystem A-5
znTrafficDescrExtTable A-5
znTrapObjs A-5
znVlan A-5
MIBs
non-conformance A-6
not supported A-7
objects not supported A-7
supported A-6
MinIndex 6-6
multi-level PNNI topology 3-3

N

neighbor LECS 2-22
netprefix 2-3
network clocking 8-3
defined 8-4
node ATM address 3-2

O

object identifier A-3

P

parallel links 3-9, 3-10
permanent virtual circuits 5-1
PGL 3-3

physical multicasting 2-14
PNN
  managing parallel links 3-9
PNNI
  adding higher-level peer groups 3-7
  adding nodes 3-5
  administrative weight 3-9
  aggregation tokens 3-10
  class of service 3-9
  connecting multiple peer groups 3-3
  controlling fail-over timing 3-12
  default node ATM address 3-2
  example 3-3
  fail-over timing 3-11
  logical link 3-6
  multi-level topology 3-3
  node address 3-1
  parallel links 3-9, 3-10
  physical connections and peer groups 3-7
  setting peer group ID 3-4
  soft PVCs 5-11
  soft PVPs 5-11
  starting PGL election 3-5
  troubleshooting 9-3
  viewing links 3-5
  viewing PGL 3-6
PNNI link timing 3-11
PNNI node addressing 3-1
PNNI routing 3-1
point-to-multipoint PVCs 5-2
point-to-point PVCs 5-1
ponf 7-4
port clock 8-3
  loopback 8-3
port clock configuration 8-3
port clocking
  local 8-3
  network 8-3
port config
  MaxVpiBits 5-2
port congestion 9-5
PVC
  available VPIs 5-2
  MaxVpiBits 5-2
  soft 5-11
PVCs 5-1
  backward traffic descriptor 5-3
  connecting to local switch client 5-4
  creating 5-1, 5-2
  point-to-multipoint 5-2
  point-to-point 5-1
  traffic descriptor 5-1
PVP
  available VPIs 5-5
  MaxVpiBits 5-5
  running ILMI 5-6
  soft 5-11
PVPs 5-5
  add pvp 5-6
  connecting 5-7
  creating 5-6
  disabling signaling 5-5
  MaxVpiBits 5-5
  set portconfig 5-5
  viewing 5-6

Q

queue buffer allocation 6-5
queue buffers 6-5

R

redundancy configuration
  scsm 7-5
  swms 7-5
redundancy for CSM 7-5
redundant LECS 2-16
redundant LES/BUS 2-19
RFCs A-1
RM cell marking 6-7
route metrics 4-7
Routing
  IISP 4-1
routing 4-1
  IISP considerations 4-2
  IISP example 4-3
  ILMI 4-2, 4-6
  incoming metric 4-7
  IP 4-9
  metrics 4-7
  outgoing metric 4-7
  reaching an NMS 4-9
  reaching the Ethernet interface 4-9
  UNI 4-5

S

scsm 7-5
SCSP 2-22, 2-24
security 9-9
SmartSwitch 6500
  SNMP agent A-1
  supported MIBs A-6
  traffic management 6-1
SmartSwitch ATM Administrator
  default community strings A-8
SMI Formats supported A-4
SNMP A-1
  community A-8
  console commands that affect the agent A-7
  default community strings A-8
  managing the SmartSwitch 6500 ..... A-7
SNMP agent support A-1
SNMPv1 ..... A-4
SNMPv2c A-4
soft PVC 5-11
soft PVP 5-11
software warranty B-2
SPVC 5-11
  checking route table 5-13
  configuring 5-15
  connections 5-12
  creating 5-12
  destination type 5-14
  MaxVpiBits 5-14
  target ATM address 5-13
  target switch 5-13
  target VPI/VCI 5-14
SPVC target 5-14
SPVP 5-11
  add spvcaddress 5-16
  adding an SPVP 5-16
  connections 5-12
  creating 5-15
  DestinationSelectType 5-16
  target 5-16
  target VPI 5-16
swms 7-5

T

target ATM address 5-13
target VPI/VCI 5-14
technical support B-1
  electronic services B-1
  fax service B-1
  hardware warranty B-2
  placing a support call B-1
  repair services B-2
  software warranty B-2
TLV set 2-13
Traffic Descriptors 6-1
traffic descriptors
  characteristics 6-2
  creating 6-1
  type number 6-2
Traffic Management 6-1
traffic management
  cell marking 6-7
  changing EPD thresholds 6-7
  EFCI 6-7
  enabling EFCI marking 6-7
  enabling RM cell marking 6-7
  EPD 6-7
  EPD threshold 6-7
  MaxIndex 6-6
  MinIndex 6-6
  queue buffers 6-5
  RM cell marking 6-7
Troubleshooting 9-1
troubleshooting
  congestion 9-4
  core dumps 9-9
  diagnosing congestion 9-4
  event categories 9-6
  events and alarms 9-6
  global congestion 9-4
  IP over ATM 9-1
  LAN emulation 9-2
  PNNI links 9-3
  port congestion 9-5
  switches in different peer groups 9-3
  switches in same peer group 9-3
tunneling 2-14

U

UNI
  controlling fail-over timing 4-6
  fail-over timing 4-6
UNI link timing 4-6
UNI routes 4-5

update firmware 7-8
upgrades 7-1
upgrading
    boot load firmware..... 7-6
    changing default boot load image..... 7-7
    POST diagnostics 7-7
    switch operating firmware 7-8
    unsuccessful update 7-1
    update firmware..... 7-1
Upgrading and Changing Firmware..... 7-1

V

VCI..... 5-2
viewing alarms 9-7
viewing events..... 9-7
virtual port..... 5-7
    root port 5-7
virtual ports 5-7
    assigning 5-7
    assigning VPIs 5-8
    Available VPIs..... 5-8
    Base VPI 5-8
    creating 5-8
    ILMI 4.0..... 5-9
    MaxVpiBits 5-8
    numbering convention 5-7
    things to watch out for 5-11
    virtual UNI..... 5-9
    VPIs used..... 5-8
virtual UNI 5-9
VLAN
    creating 2-1
    IP over ATM..... 2-1
VPI 5-2
VPI/VCI pair 5-2

W

warranty
    hardware B-2
    software B-2