

SunATM™ SBus Adapters Manual



THE NETWORK IS THE COMPUTER™

Sun Microsystems Computer Company

A Sun Microsystems, Inc. Business
2550 Garcia Avenue
Mountain View, CA 94043 USA
415 960-1300 fax 415 969-9131

Part No.: 802-6504-10
Revision A, September 1996

Copyright 1996 Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, California 94043-1100 U.S.A.

All rights reserved. This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any.

Portions of this product may be derived from the UNIX[®] system and from the Berkeley 4.3 BSD system, licensed from the University of California. UNIX is a registered trademark in the United States and in other countries and is exclusively licensed by X/Open Company Ltd. Third-party software, including font technology in this product, is protected by copyright and licensed from Sun's suppliers.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19.

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

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

X Window System is a trademark of X Consortium, Inc.

THIS PUBLICATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

Copyright 1996 Sun Microsystems, Inc., 2550 Garcia Avenue, Mountain View, Californie 94043-1100 U.S.A.

Tous droits réservés. Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie et la décompilation. Aucune partie de ce produit ou de sa documentation associée ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Des parties de ce produit pourront être dérivées du système UNIX[®] et du système Berkeley 4.3 BSD licencié par l'Université de Californie. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays, et licenciée exclusivement par X/Open Company Ltd. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

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

Les utilisateurs d'interfaces graphiques OPEN LOOK[®] et Sun[™] ont été développés de Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox Corporation pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique, cette licence couvrant aussi les licenciés de Sun qui mettent en place les utilisateurs d'interfaces graphiques OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

Le système X Window est un produit du X Consortium, Inc.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" SANS GARANTIE D'AUCUNE SORTE, NI EXPRESSE NI IMPLICITE, Y COMPRIS, ET SANS QUE CETTE LISTE NE SOIT LIMITATIVE, DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DES PRODUITS A REpondre A UNE UTILISATION PARTICULIERE OU LE FAIT QU'ILS NE SOIENT PAS CONTREFAISANTS DE PRODUITS DE TIERS.



Your SBus card is marked to indicate its FCC, DOC, and VCCI class. Please read the appropriate section that corresponds to the marking on your SBus card before attempting to install it into your system.

FCC Class A Notice — United States

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.
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 generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Modifications

Modifications to this device, not approved by Sun Microsystems, Inc., may void the authority granted to the user by the FCC to operate this equipment.

FCC Class B Notice — United States

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.
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 B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/television technician for help.

Modifications

Modifications to this device, not approved by Sun Microsystems, Inc., may void the authority granted to the user by the FCC to operate this equipment.

DOC Class A Notice — Canada

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

Avis Concernant les Systèmes Appartenant à la Classe A du DOC — Canada

Le présent appareil numérique n'émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de la classe A prescrites dans le Règlement sur le brouillage radioélectrique édicté par le ministère des Communications du Canada.

DOC Class B Notice — Canada

This digital apparatus does not exceed the Class B limits for radio noise emission for a digital apparatus as set out in the Radio Interference Regulations of the Canadian Department of Communications.

Avis Concernant les Systèmes Appartenant à la Classe B du DOC — Canada

Le présent appareil numérique n'émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de la classe B prescrites dans le Règlement sur le brouillage radioélectrique édicté par le ministère des Communications du Canada.

Nippon—Japan

第一種VCCI基準に関するお知らせ

この装置は、第一種情報装置(商工業地域において使用されるべき情報装置)で商工業地域での電波障害防止を目的とした情報処理装置等電波障害自主規制協議会(VCCI)基準に適合しております。

この装置は、第一種または第二種ワークステーションのオプションです。本装置を使用する場合、システムとしての適合レベルは下記の通りです。

第一種ワークステーション：第一種情報装置

第二種ワークステーション：第一種情報装置

本装置を使用する第二種ワークステーションは、第一種情報装置(商工業地域において使用されるべき情報装置)となります。

従って、住宅地域またはその隣接した地域で使用すると、ラジオ、テレビジョン受信機等に受信障害を与えることがあります。

取扱説明書に従って正しい取り扱いをして下さい。

第二種VCCI基準に関するお知らせ

この装置は、第二種情報装置（住宅地域またはその隣接した地域において使用されるべき情報装置）で住宅地域での電波障害防止を目的とした情報処理装置等電波障害自主規制協議会（VCCI）基準に適合しております。

この装置は、第一種または第二種ワークステーションのオプションです。本装置を使用する場合、システムとしての適合レベルは下記の通りです。

第一種ワークステーション：第一種情報装置

第二種ワークステーション：第二種情報装置

本装置を使用する第一種ワークステーションは、第一種情報装置（商工業地域において使用されるべき情報処理装置）となります。従って、住宅地域またはその隣接した地域で使用すると、ラジオ、テレビジョン受信機等に受信障害を与えることがあります。

本装置を使用する第二種ワークステーションは、第二種情報装置（住宅地域またはその隣接地域において使用されるべき情報装置）となります。従って、本装置をラジオ、テレビジョン受信機に近接してご使用になると、受信障害の原因となることがあります。

取扱説明書に従って正しい取り扱いをして下さい。

Contents

1. Introducing and Installing the SunATM-622/MMF SBus Adapter 2.1	1-1
1.1 Hardware Requirements	1-2
1.2 Software Requirements	1-2
1.3 Installing the SunATM-622/MMF SBus Adapter 2.1	1-3
1.3.1 SunATM-622/MMF SBus Adapter 2.1 Installation.	1-3
1.3.2 SunATM-622/MMF SBus Adapter 2.1 Wiring Configuration	1-6
1.4 Testing the SunATM-622 SBus Adapter Before Booting	1-7
2. Introducing and Installing the SunATM-155 SBus Adapters 2.1 2-1	
2.1 Hardware Requirements	2-5
2.2 Software Requirements	2-6
2.3 Installing the SunATM-155 SBus Adapters	2-6
2.3.1 SunATM-155 SBus Adapter Installation.	2-7
2.3.2 SunATM-155/MMF SBus Adapter 2.1 Wiring Configuration	2-8

2.3.3	SunATM-155/UTP5 SBus Adapter 2.1 Wiring Configuration	2-9
2.4	Testing the SunATM-155 SBus Adapter Before Booting .	2-10
3.	Installing the SunATM Software	3-1
3.1	Before Installing the SunATM Software	3-2
3.2	Installing the SunATM 2.1 Software.....	3-2
3.2.1	Adding the Software Packages Using pkgadd	3-2
3.2.2	Using the pkgadd Utility.....	3-5
3.2.3	Checking the Package Installation Using pkgchk .	3-5
3.2.4	Checking the SunATM Software Installation Using pkginfo.....	3-6
3.2.5	Removing the Software Packages Using pkgrm ...	3-6
3.3	Rebooting the System and Examining Network Interfaces	3-6
4.	Configuring the SunATM Interface.....	4-1
4.1	SunATM SBus Adapter Versions	4-1
4.2	New Features in the SunATM 2.1 Software.....	4-2
4.2.1	Configuring Network Parameters.....	4-2
4.2.2	Support for the Full ATM MIBs in ILMI	4-2
4.2.3	SNMP Forwarding Agent	4-2
4.2.4	Logical Interface Names.....	4-2
4.2.5	Joining Multiple Emulated LANs.....	4-3
4.2.6	Mounting the /opt directory Over an ATM Interface	4-3
4.2.7	Per-Adapter Framing Interface Selection.....	4-3
4.3	Using the atmadm Configuration Program.....	4-4
4.3.1	Starting the atmadm Configuration Program ...	4-4

4.3.2	atmadmin Main Menu	4-5
4.3.3	atmadmin and the SunATM Configuration Files in the /etc directory	4-7
4.4	atmadmin Parameter Groups	4-8
4.4.1	Physical Layer Parameter Group	4-10
4.4.2	Signalling Parameter Group	4-11
4.4.3	ILMI Parameter Group	4-12
4.4.4	Classical IP Parameter Group	4-12
4.4.5	LAN Emulation Parameter Group	4-16
5.	Editing the SunATM Configuration Files	5-1
5.1	Editing the /etc/atmconfig File	5-2
5.1.1	Changing the Framing Interface in the /etc/atmconfig File	5-3
5.1.2	Example of an /etc/atmconfig File	5-3
5.2	Configuring a Classical Internet Protocol Interface	5-4
5.2.1	Editing the /etc/aarconfig File	5-4
5.2.2	Using Variables in the /etc/aarconfig File	5-8
5.2.3	Sample Classical IP Configurations	5-11
5.3	Configuring a LAN Emulation Interface	5-14
5.3.1	Editing the /etc/laneconfig File	5-14
5.3.2	Using Variables in the /etc/laneconfig File	5-18
5.3.3	Sample LAN Emulation Configurations	5-19
5.4	Supporting Logical Interfaces	5-20
5.5	Supporting Multiple Emulated LANS on a Single Interface 5-21	

5.6	Tuning Your System for Better SunATM Performance . . .	5-23
6.	Classical IP and LAN Emulation Protocols	6-1
6.1	ATM Addresses and Address Registration	6-2
6.1.1	ATM Address Registration Daemon (ILMID)	6-2
6.2	Classical Internet Protocol	6-3
6.2.1	ATM Address Resolution	6-3
6.2.2	ATM ARP Address Resolution Tables	6-4
6.3	LAN Emulation	6-5
6.3.1	LAN Emulation Services	6-6
6.3.2	Resolving an IP Address to an ATM Connection	6-7
6.3.3	LAN Emulation Connections	6-9
6.4	ATM and SNMP	6-10
A.	Wiring Scheme and Pin Descriptions	A-1
B.	SunATM SBus Adapters Specifications	B-1
B.1	SunATM-155 SBus Adapters 2.1 Specifications	B-1
B.1.1	Performance Specifications	B-1
B.1.2	Power Specifications	B-2
B.1.3	Physical Dimensions	B-2
B.1.4	Environmental Specifications	B-2
B.2	SunATM-622/MMF SBus Adapter 2.1 Specifications	B-3
B.2.1	Performance Specifications	B-3
B.2.2	Power Specifications	B-3
B.2.3	Physical Dimensions	B-3
B.2.4	Environmental Specifications	B-4

C. Troubleshooting and Error Messages	C-1
C.1 Troubleshooting While Starting a SunATM Interface ...	C-1
C.1.1 Generic Configuration	C-2
C.1.2 Classical IP Configuration	C-3
C.1.3 LAN Emulation Configuration	C-6
C.2 Error Messages	C-10
C.2.1 Error Messages from <code>S00sunatm</code>	C-10
C.2.2 Error Messages from <code>aarsetup</code> and <code>lanesetup</code> .	C-13
C.2.3 Error Messages from the Kernel Drivers	C-17
D. Managing SunATM Interfaces with SNMP	D-1
D.1 Installing the SunATM SNMP Software	D-1
D.2 Setting Up the Management Console.	D-1
D.3 Setting Up Agent Systems.	D-3
E. Application Programmers' Interface	E-1
E.1 Using the SunATM API with the q93b and the ATM Device Drivers	E-2
E.1.1 Establishing a Connection to the q93b Driver	E-2
E.1.2 Setting up an ATM Connection Over a Switched Virtual Circuit (SVC)	E-3
E.1.3 Connecting, Sending, and Receiving Data with the ATM Device Driver.	E-7
F. Running Diagnostic Tests	F-1
F.1 SunVTS Validation and Test Suite	F-1
F.2 Using the OpenBoot PROM Selftest.	F-2
Glossary.	Glossary-1

Index	Index-1
-------------	---------

Figures

Figure 1-1	Sun Ultra 1 Series System Back Panel	1-4
Figure 1-2	SunATM-622/MMF SBus Adapter	1-4
Figure 1-3	Installing the SunATM-622/MMF SBus Adapter	1-5
Figure 1-4	SunATM-622/MMF SBus Adapter's Backplate	1-5
Figure 2-1	SunATM-155/MMF SBus Adapter 2.1 and Back Panel	2-3
Figure 2-2	SunATM-155/UTP5 SBus Adapter 2.1 and Back Panel	2-4
Figure 2-3	SunATM-155/MMF SBus Adapter 2.1	2-8
Figure 2-4	SunATM-155/UTP5 SBus Adapter 2.1	2-9
Figure 4-1	atmadmin Main Menu	4-5
Figure 4-2	System Parameter Group Menu	4-6
Figure 4-3	Interface Configuration Menu	4-7
Figure 4-4	Physical Layer Parameter Menu	4-10
Figure 4-5	Signalling Parameter Menu	4-11
Figure 4-6	ILMI Configuration Menu	4-12
Figure 4-7	Classical IP Parameter Menu	4-13
Figure 4-8	LAN Emulation Instance Menu	4-17

Figure 4-9	LAN Emulation Per-Instance Parameters Menu	4-18
Figure 5-1	Example <code>/etc/atmconfig</code> File.	5-3
Figure 6-1	ATM Address Fields.	6-2
Figure 6-2	Using <code>atmsnmpd</code> as a Forwarding Agent	6-11
Figure A-1	Designation T568B.	A-1
Figure E-1	ATM Signalling	E-2
Figure E-2	Message Format	E-4
Figure E-3	Normal Call Set up and Tear Down	E-10

Tables

Table 1-1	Platform Architecture with Examples of Systems	1-2
Table 2-1	Platform Architecture with Examples of Systems	2-5
Table 4-1	Supported SunATM SBus Adapters	4-1
Table 4-2	Basic Navigational Commands in <code>atmadmin</code>	4-5
Table 4-3	Configurable Parameters in the SunATM Software	4-8
Table 4-4	Predefined SunATM Variables	4-15
Table 5-1	<code>/etc/atmconfig</code> Field Descriptions	5-2
Table 5-2	<code>/etc/aarconfig</code> File Flags	5-5
Table 5-3	<code>/etc/aarconfig</code> File Flag Options	5-7
Table 5-4	Predefined SunATM Variables	5-9
Table 5-5	<code>/etc/laneconfig</code> Entry Descriptions	5-15
Table 5-6	<code>/etc/laneconfig</code> Flag Descriptions	5-16
Table 5-7	<code>laneconfig</code> Flag Requirements and Options	5-17
Table 5-8	Predefined SunATM Variables	5-18
Table 6-1	LAN Emulation Connections.	6-9
Table A-1	Pin Descriptions for the 96-Pin SBus Connector	A-2

Table B-1	Performance Specifications	B-1
Table B-2	Power Specifications	B-2
Table B-3	Physical Dimensions	B-2
Table B-4	Environmental Specifications	B-2
Table B-5	Performance Specifications	B-3
Table B-6	Power Specifications	B-3
Table B-7	Physical Dimensions	B-3
Table B-8	Environmental Specifications	B-4
Table E-1	Messages Between the User and the q93b Driver.	E-3
Table E-2	Fields in the M_PROTO mblock	E-4
Table E-3	qcc Functions	E-5
Table E-4	atm_util Function Overview	E-7

Preface

SunATM™ SBus Adapters Manual provides installation instructions for the SunATM-622/MMF SBus Adapter 2.1, the SunATM-155/MMF SBus Adapter 2.1, and the SunATM-155/UTP5 SBus Adapter 2.1. This manual also describes how to install and configure the SunATM 2.1 software.

These instructions are designed for an experienced system administrator with networking knowledge.

How This Book Is Organized

Chapter 1, “Introducing and Installing the SunATM-622/MMF SBus Adapter 2.1,” introduces the SunATM-622 adapter and describes how to install and verify the adapter.

Chapter 2, “Introducing and Installing the SunATM-155 SBus Adapters 2.1,” introduces the SunATM-155 adapters and describes how to install and verify the adapters.

Chapter 3, “Installing the SunATM Software,” describes how to install the SunATM software packages using the `pkgadd` utility.

Chapter 4, “Configuring the SunATM Interface,” describes some of the new features in the SunATM software and how to configure the software using the `atmadmin` configuration program.

Chapter 5, “Editing the SunATM Configuration Files,” shows how you can edit the SunATM 2.1 configuration files to best suit your network.

Chapter 6, “Classical IP and LAN Emulation Protocols,” provides background information about both the Classical Internet Protocol (IP) and the local area network (LAN) emulation protocol, which allows IP traffic to run over Asynchronous Transfer Mode (ATM) interfaces.

Appendix A, “Wiring Scheme and Pin Descriptions,” describes the wiring scheme for the T568 and 96-pin SBus connectors.

Appendix B, “SunATM SBus Adapters Specifications,” lists the specifications for all of the SunATM adapters.

Appendix C, “Troubleshooting and Error Messages,” provides a troubleshooting section for the software installation and configuration, and defines the software error messages.

Appendix D, “Managing SunATM Interfaces with SNMP,” describes how to install and to set up Simple Network Management Protocol (SNMP) agents.

Appendix E, “Application Programmers’ Interface,” defines the SunATM application programmer’s interface (API).

Appendix F, “Running Diagnostic Tests,” provides pointers to the SunVTS™ test suite, and it describes how to run the OpenBoot™ selftest.

The **Glossary** contains list of words and acronyms found in this book with their definitions.

UNIX Commands

This document may not include specific software commands or procedures. Instead, it may name software tasks and refer you to operating system documentation or the handbook that was shipped with your new hardware.

The type of information that you might need to use references for includes:

- Shutting down the system
- Booting the system
- Configuring devices
- Other basic software procedures

See one or more of the following:

- *Solaris 2.x Handbook for SMCC Peripherals* contains Solaris™ 2.x software commands.
- On-line AnswerBook™ for the complete set of documentation supporting the Solaris 2.x software environment.
- Other software documentation that you received with your system.

Typographic Conventions

The following table describes the typographic changes used in this book.

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

Shell Prompts

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

Shell	Prompt
C shell	machine_name%
C shell superuser	machine_name#
Bourne shell and Korn shell	\$
Bourne shell and Korn shell superuser	#

Ordering Sun Documents

The SunDocs Order Desk is a distribution center for Sun Microsystems technical documents. You can use major credit cards and company purchase orders. You can order documents in the following ways:

Country	Telephone	Fax
United States	1-800-873-7869	1-800-944-0661
United Kingdom	0-800-89-88-88	0-800-89-88-87
France	05-90-61-57	05-90-61-58
Belgium	02-720-09-09	02-725-88-5
Luxembourg	32-2-720-09-09	32-2-725-88-5
Germany	01-30-81-61-91	01-30-81-61-92
The Netherlands	06-022-34-45	06-022-34-46
Sweden	020-79-57-26	020-79-57-27
Switzerland	155-19-26	155-19-27
Japan	0120-33-9096	0120-33-9097

World Wide Web: <http://www.sun.com/sunexpress/>

Sun Welcomes Your Comments

Please use the *Reader Comment Card* that accompanies this document. We are interested in improving our documentation and welcome your comments and suggestions.

If a card is not available, you can email or fax your comments to us. Please include the part number of your document in the subject line of your email or fax message.

- Email: smcc-docs@sun.com
- Fax: SMCC Document Feedback
1-415-786-6443

Notes, Cautions, and Warnings



Warning – This equipment contains lethal voltage. Accidental contact can result in serious injury or death.



Caution – Improper handling by unqualified personnel can cause serious damage to this equipment. Unqualified personnel who tamper with this equipment may be held liable for any resultant damage to the equipment.

Individuals who remove any outer panels to access this equipment must observe all safety precautions and ensure compliance with skill level requirements, certification, and all applicable local and national laws.

Procedures contained in this document must be performed by qualified service-trained maintenance providers.

Note – Before you begin, carefully read each of the procedures in this manual. If you have not performed similar operations on comparable equipment, *do not attempt* to perform these procedures.

Introducing and Installing the SunATM-622/MMF SBus Adapter 2.1



The SunATM-622/MMF SBus Adapter 2.1 is a double-wide adapter that conforms to the specifications of the Asynchronous Transfer Mode (ATM) Forum. The adapter offers 622 Mbps network bandwidth over a multimode fiber optic cable.

The SunATM-622/MMF SBus Adapter 2.1 is designed for operation in systems that run the Solaris environment, revision 2.4 or later. To use the SunATM-622 SBus adapter, the system must also contain an OpenBoot™ PROM (OBP), level 2.0 or later. An on-board FCode PROM provides the configuration support that identifies the adapter to the system.

The highlights of the SunATM-622/MMF SBus Adapter 2.1 include:

- Conforms to IEEE 1496
- Supports 622 Mbps operation over 62.5/125 μ Multimode fiber
- Integrate SBus/SAR (Segmentation And Reassembly) ASIC SAHI (SBus to ATM Host Interface) implemented in standard CMOS
- SAR function aligned with ATM Forum specified and International Telecommunications Union - Telecommunication Sector (ITU-TS) approved ATM Adaptation Layer (AAL) 5
- Supports SONET/SDH (Synchronous Optical NETWORK/Synchronous Digital Hierarchy) physical layer framing structure
- Up to 126 simultaneous transmit channels and up to 1024 simultaneous open receive channels
- Compatible with relevant emerging standards (including existing ATM Forum baseline specifications and ITU-TS)

1.1 Hardware Requirements

You will need an ATM switch to build an ATM network. To connect the SunATM-622 SBus adapter to the ATM switch, you will need a multimode fiber cable with an SC connector.

Refer to the manual supplied with the ATM switch for the specific instructions about cable connections.

Table 1-1 shows the Sun-4U, Sun-4m, and Sun-4d architecture systems that support the SunATM-622 SBus adapter.

Table 1-1 Platform Architecture with Examples of Systems

Platform Architecture	System Type
Sun-4U	Sun Ultra™ 1 Creator Series Sun Ultra 1 Series Sun Ultra 2 Series Ultra Enterprise™ 1 Ultra Enterprise 2 Ultra Enterprise 3000 Ultra Enterprise 4000 Ultra Enterprise 5000 Ultra Enterprise 6000
Sun-4m	SPARCstation™ 20 Series
Sun-4d	SPARCserver™ 1000 and 1000E SPARCcenter™ 2000 and 2000E

1.2 Software Requirements

Note – Before installing the SunATM 2.1 software packages, you must first install the SunATM-622 SBus adapter into the system.

The SunATM-622 SBus adapter and the SunATM 2.1 software are supported on systems running the Solaris environment, revision 2.4 or later.

The SunATM 2.1 CD-ROM that shipped with the adapter contains the *required* driver software that must be installed in order to connect a SunATM-622 adapter to a network.

1.3 Installing the SunATM-622/MMF SBus Adapter 2.1

Before installing the SunATM 2.1 software packages, you must install the SunATM-622 SBus adapter into the system.

Note – Only one SunATM-622 SBus adapter is supported per SBus. For example, on desktop machines that have only one SBus per system (even though there may be multiple SBus slots), no more than one SunATM-622 SBus adapter is supported per system.

1.3.1 SunATM-622/MMF SBus Adapter 2.1 Installation

Refer to the hardware installation or service manual that shipped with your system for information about installing SBus adapters.



Caution – Make sure you use a wrist strap. It provides grounding for static electricity between your body and the system chassis. If you do not wear a wrist strap, the system components can be damaged by harmful electrical discharge.

1. Install the SunATM-622 SBus adapter according to the SBus installation procedures in the hardware installation or service manual for your system.

Although your system's service manual describes how to install SBus adapters, you may require more information on how to install the double-wide SunATM-622 SBus adapter into your system. Please read the additional notes, cautions, and steps below.

Note – The SunATM-622/MMF SBus Adapter 2.1 is shipped with a rubber plug that keeps the connector free of dust during shipping and storage. Before installing the adapter, you must first remove the plug from the SC fiber receptacle (Figure 1-2).



Caution – If you are installing the SunATM-622 SBus adapter into a Sun Ultra 1 Creator series or a Sun Ultra 1 series system, you must install the adapter in the top SBus slot (slot 1) of the system (see Figure 1-1).

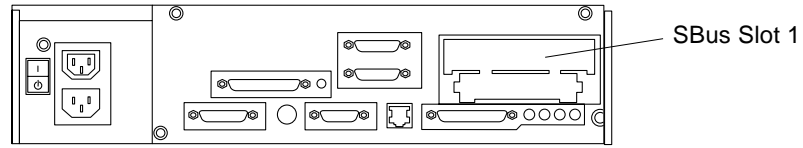


Figure 1-1 Sun Ultra 1 Series System Back Panel

Note - If you are installing the SunATM-622 SBUS adapter into a Sun Ultra 1 system, you will not be able to connect the adapter's two SBUS connectors into two SBUS sockets. Instead, you must install the right connector into the top SBUS socket. The adapter will operate properly in a Sun Ultra 1 system if only the right connector is installed (Figure 1-2).

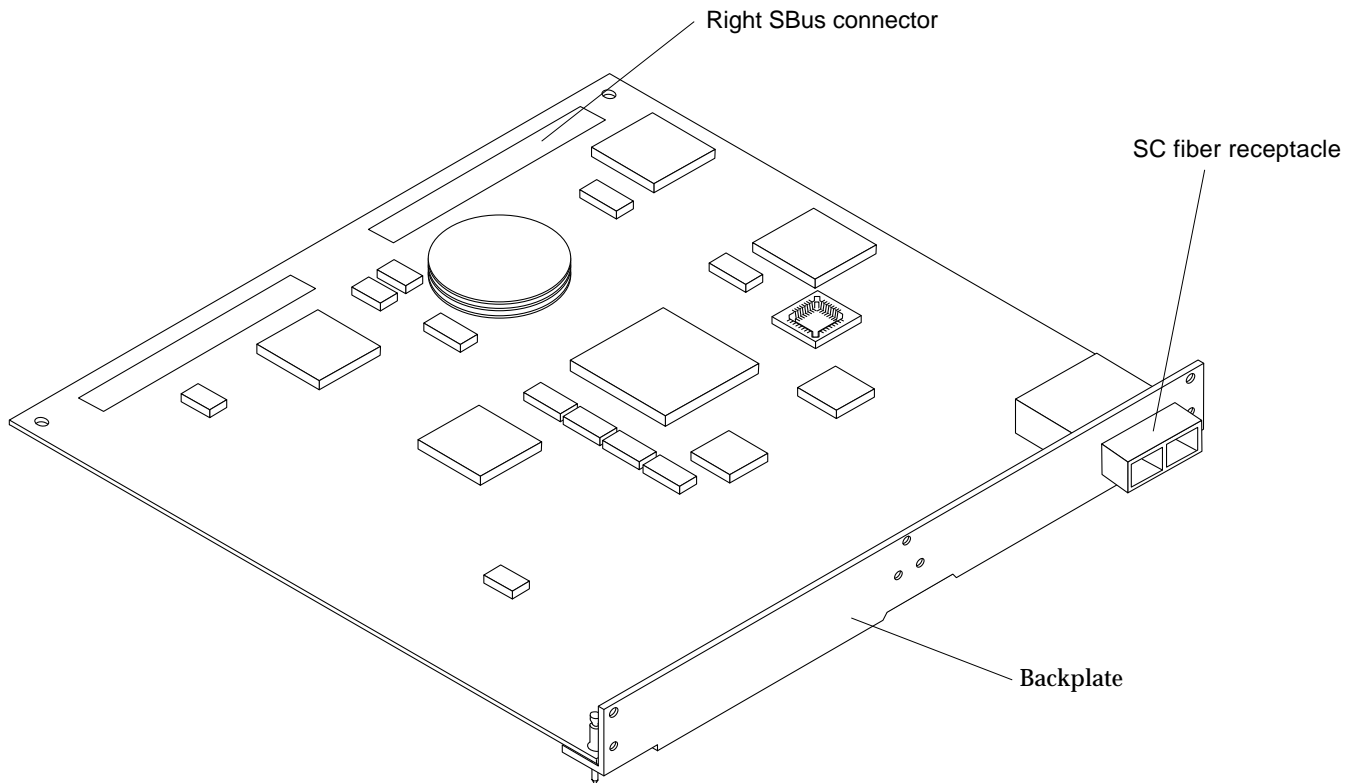


Figure 1-2 SunATM-622/MMF SBUS Adapter

Because of the large size of the SunATM-622 Sbus adapter's backplate (see Figure 1-2), you may need to follow these additional steps to correctly install the adapter.

- a. **Insert the SunATM-622 SBus adapter at an angle into the system by hooking the adapter under the top tabs on the system's back panel (Figure 1-3).**

Be sure the SC fiber receptacle goes through the slot on the back panel.

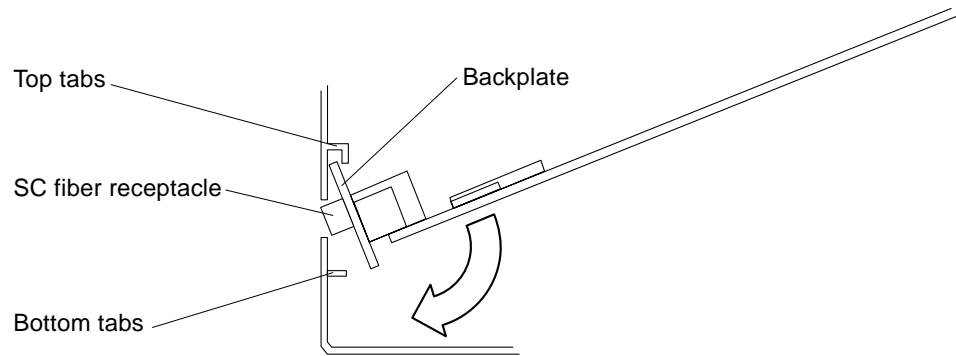


Figure 1-3 Installing the SunATM-622/MMF SBus Adapter

- b. **Align the adapter's right SBus connector with the system's right (or only) SBus socket, and gently lower the SBus adapter over the SBus socket.**
- c. **Carefully slide the adapter to the left and to the right until the system's bottom tabs (see Figure 1-3) slide into the bottom recesses (see Figure 1-4) of the adapter's backplate.**

Once the bottom tabs are in place, you should be able to lower the adapter easily into the system.

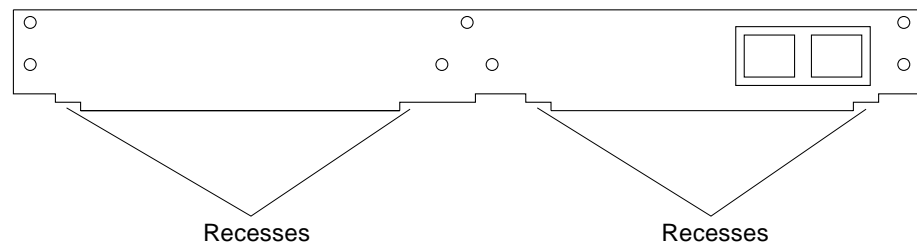


Figure 1-4 SunATM-622/MMF SBus Adapter's Backplate

d. Gently press the SunATM-622 SBus adapter's connector(s) into the system's socket(s).

Do not force the adapter, or you may damage the pins on the connectors. If you are installing the adapter into a Sun Ultra 1 system, press only the right connector into the top SBus socket. On all other supported systems, press the two connectors into the two SBus sockets.

Note – Refer to your system's installation or service manual for more information about installing SBus adapters.

2. Verify the hardware installation by executing a test command.



Caution – Do not boot the operating system until SunATM installation is verified. See “Testing the SunATM-622 SBus Adapter Before Booting” on page 1-7.

1.3.2 SunATM-622/MMF SBus Adapter 2.1 Wiring Configuration

The SunATM-622/MMF SBus Adapter 2.1 is shipped with the SC fiber receptacle already keyed (Figure 1-2). As you hold the adapter with the receptacle pointed toward you, “transmit” is on the left and “receive” is on the right.

To connect the SunATM-622/MMF SBus Adapter 2.1 to the network:

- ◆ **Connect one end of the multimode fiber cable into the fiber receptacle on the adapter and connect the other end to the ATM switch (Figure 1-2).**

Refer to the installation or users manual supplied with the ATM hardware interface for additional information.



Caution – Do not boot the operating system until the installation is tested and verified. See Section 1.4, “Testing the SunATM-622 SBus Adapter Before Booting,” for more information.

1.4 Testing the SunATM-622 SBus Adapter Before Booting

After you have installed the SunATM-622 SBus adapter, *but before booting the system*, verify installation by executing the `show-devs` command at the OpenBoot prompt.



Caution – Do not change the SBus slot in which a SunATM-622/MMF SBus adapter is installed once the system has been booted. The Solaris 2.x software environment remembers the location of each SBus adapter that has been installed. Switching SBus slots will cause the operating system to assume that you removed your original SunATM-622 SBus adapter and added a second adapter to the system. Refer to the `path_to_inst` man page for more information.

1. Use the `show-devs` command to display the device information.

The `show-devs` *device path* command displays all devices known to the system directly beneath a given level in the device hierarchy.

Note – The SunATM-622/MMF SBus adapters shipped with the SunATM 2.1 software will be identified by the driver name `ba`.

The `show-devs` command used by itself shows the entire device tree. Code Example 1-1 and Code Example 1-2 below show information for a Ultra Enterprise 3000 server with an OpenBoot PROM level 3.2.3.

Note – The `/SUNW,ba@2,0` bold entries in the responses to both commands indicate that the system recognizes the SunATM-622 SBus adapter plugged into SBus slot 2.

Code Example 1-1 `show-devs` Results From a Sun Ultra Enterprise 3000 Server

```
ok show-devs /sbus
/sbus@3,0/SUNW,soc@0,0
//sbus@3,0/SUNW,ba@2,0
/sbus@3,0/SUNW,fas@3,8800000
/sbus@3,0/SUNW,hme@3,8c00000
/sbus@3,0/SUNW,soc@d,10000
/sbus@3,0/SUNW,fas@3,8800000/st
/sbus@3,0/SUNW,fas@3,8800000/sd
```

Code Example 1-2 show-devs Results From a Ultra Enterprise 3000 Server

```
ok show-devs
/counter-timer@3,3c00
/sbus@3,0
/fhc@2,f8800000
/SUNW,UltraSPARC@7,0
/SUNW,UltraSPARC@6,0
/fhc@6,f8800000
/central@1f,0
/virtual-memory
/memory@0,0
/aliases
/options
/openprom
/chosen
/packages
/sbus@3,0/SUNW,soc@0,0
/sbus@3,0/SUNW,ba@2,0
/sbus@3,0/SUNW,fas@3,8800000
/sbus@3,0/SUNW,hme@3,8c00000
/sbus@3,0/SUNW,soc@d,10000
/sbus@3,0/SUNW,fas@3,8800000/st
/sbus@3,0/SUNW,fas@3,8800000/sd
/fhc@2,f8800000/sbus-speed@0,500000
/fhc@2,f8800000/eeprom@0,300000
/fhc@2,f8800000/flashprom@0,0
/fhc@2,f8800000/environment@0,400000
/fhc@2,f8800000/ac@0,1000000
/fhc@6,f8800000/flashprom@0,0
/fhc@6,f8800000/sram@0,200000
/fhc@6,f8800000/environment@0,400000
/fhc@6,f8800000/simm-status@0,600000
/fhc@6,f8800000/ac@0,1000000
/central@1f,0/fhc@0,f8800000
/central@1f,0/fhc@0,f8800000/clock-board@0,900000
/central@1f,0/fhc@0,f8800000/zs@0,904000
/central@1f,0/fhc@0,f8800000/zs@0,902000
/central@1f,0/fhc@0,f8800000/eeprom@0,908000
/openprom/client-services
/packages/terminal-emulator
```

2. Boot the system.

Refer to the *Solaris 2.x Peripheral's Handbook* for more information.

Introducing and Installing the SunATM-155 SBus Adapters 2.1



The SunATM-155/MMF SBus Adapter 2.1 and the SunATM-155/UTP5 SBus Adapter 2.1 are single-wide SBus adapters that conform to the specifications of the Asynchronous Transfer Mode (ATM) Forum. The adapters offer 155 Mbps network bandwidth over either Multimode fiber optic cable or Category 5 unshielded twisted pair (UTP) copper wire.

The SunATM-155 SBus adapters are designed for operation in systems that run under the Solaris environment, revision 2.4 or later. To use the SunATM-155 adapters, the system must also contain an OpenBoot™ PROM (OBP), level 2.0 or later. An on-board FCode PROM provides the configuration support that identifies the SunATM-155 SBus adapters to the system.

The highlights of the SunATM-155 SBus adapters 2.1 include:

- Conforms to IEEE 1496
- Supports 155-Mbps operation over:
 - 62.5/125 μ Multimode fiber (SunATM155/MMF Adapter) or
 - UTP Category 5 wire (SunATM-155/UTP5 Adapter)
- Integrate SBus/SAR (segmentation and reassembly) ASIC SAHI (SBus to ATM host interface) implemented in standard CMOS
- SAR function aligned with ATM Forum specified and International Telecommunications Union - Telecommunication Sector (ITU-TS) approved ATM Adaptation Layer (AAL) 5
- Supports the SONET/SDH (Synchronous Optical NETWORK/Synchronous Digital Hierarchy) physical layer framing structure

- Up to 126 simultaneous transmit channels and up to 1024 simultaneous open receive channels
- Compatible with relevant emerging standards (including existing ATM Forum baseline specifications and ITU-TS)

Note – Level 2.x OpenBoot PROMs (or later) are *required* for systems using the SunATM-155 adapters. If lower-level boot PROMs are installed on your system, you must upgrade the boot PROMs before using SunATM-155 adapters.

To find the OpenBoot PROM (OBP) revision level on your system, type `.version` at the `<#0> ok` prompt.

Figure 2-1 shows the SunATM-155/MMF SBus Adapter 2.1 and Figure 2-2 shows the SunATM-155/UTP5 SBus Adapter 2.1.

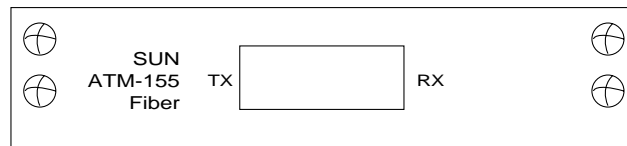
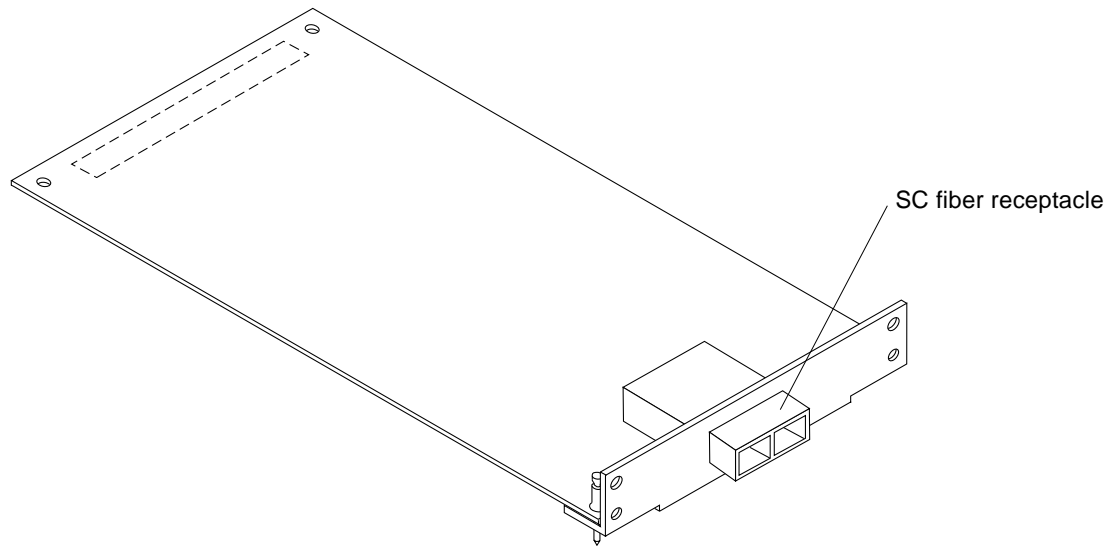


Figure 2-1 SunATM-155/MMF SBus Adapter 2.1 and Back Panel

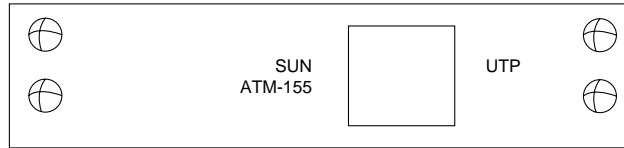
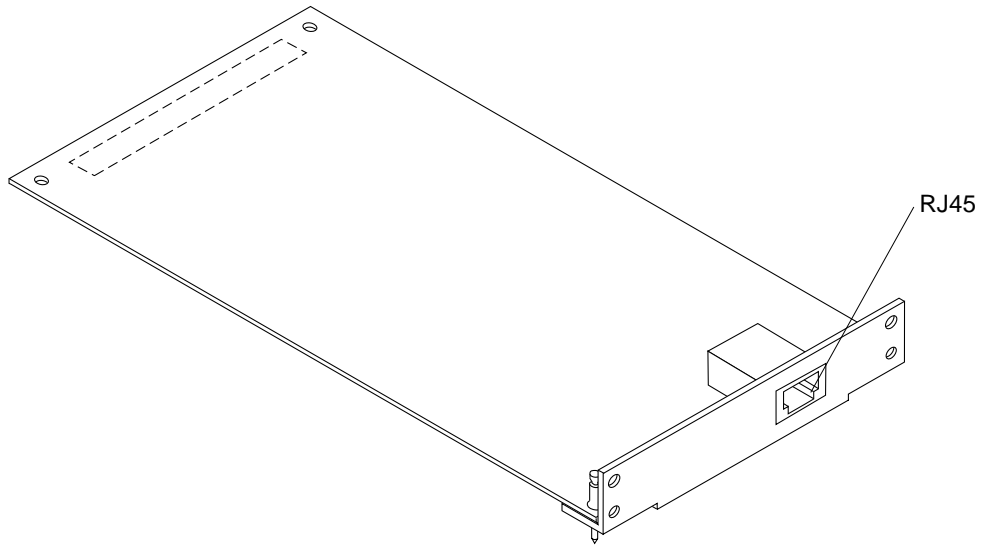


Figure 2-2 SunATM-155/UTP5 SBus Adapter 2.1 and Back Panel

2.1 Hardware Requirements

You need an ATM switch to build an ATM network. To connect the SunATM-155 SBus adapters to the ATM switch, you need the following cables:

- SunATM-155/MMF SBus Adapter 2.1 - Multimode fiber cable with an SC connector
- SunATM-155/UTP5 SBus Adapter 2.1 - Category 5 UTP with a RJ-45 connector

Refer to the manual supplied with the ATM switch for specific instructions about cable connections.

Table 2-1 shows the SBus based Sun-4U, Sun-4m, Sun-4d, and Sun-4c architecture systems that support the SunATM-155 SBus adapters 2.1.

Table 2-1 Platform Architecture with Examples of Systems

Platform Architecture	System Type
Sun-4U	Sun Ultra™ 1 Creator Series Sun Ultra 1 Series Sun Ultra 2 Series Ultra Enterprise™ 1 Ultra Enterprise 2 Ultra Enterprise 3000 Ultra Enterprise 4000 Ultra Enterprise 5000 Ultra Enterprise 6000
Sun-4m	SPARCstation™ Classic SPARCstation LX SPARCstation 4 SPARCstation 5 SPARCstation 10 SPARCstation 10SX SPARCstation 20 Series SPARCstation 600 Series
Sun-4d	SPARCserver™ 1000 and 1000E SPARCcenter™ 2000 and 2000E
Sun-4c	SPARCstation 2 SPARCstation IPX™

2.2 *Software Requirements*

Note – Before installing the SunATM 2.1 software packages, you must first install the SunATM SBus adapter into the system.

The SunATM-155 SBus adapters are supported on systems running under the Solaris environment, revision 2.4 or later.

The SunATM 2.1 CD-ROM that shipped with the SBus adapter contains the *required* driver software that must be installed in order to connect a SunATM-155 SBus adapter to a network.

2.3 *Installing the SunATM-155 SBus Adapters*

Before installing the SunATM 2.1 software packages, you must install the SunATM-155 SBus adapter into the system.

Note – SunATM-155 adapters are supported on systems running the Solaris software environment, revision 2.4 or later.

Up to two SunATM-155 adapters are supported per SBus. For example, on desktop machines that have only one SBus per system (even though there may be multiple SBus slots), no more than two SunATM-155 adapter are supported per system.

2.3.1 SunATM-155 SBus Adapter Installation

An extender plate (part number: 560-1977-01) is included with the SunATM-155 SBus adapter. You must attach this extender plate to the SBus adapter before installing the adapter in some older systems. Refer to the hardware installation or service manual that shipped with your system for information about installing SBus adapters.



Caution – Make sure you use a wrist strap. It provides grounding for static electricity between your body and the system chassis. If you do not wear a wrist strap, the system components can be damaged by harmful electrical discharge.

Note – The SunATM-155/MMF SBus adapter is shipped with a rubber plug that keeps the connector free of dust. To install the adapter, the plug must first be removed.

1. **Install the SunATM-155 SBus adapter according to the SBus installation procedures in the hardware installation or service manual for your system.**
2. **Verify the hardware installation by executing a test command.**



Caution – Do not boot the operating system until SunATM installation is verified. See “Testing the SunATM-155 SBus Adapter Before Booting” on page 2-10.

The following two sections provide an introduction to the physical connectors and wiring characteristics of the SunATM-155/MMF and SunATM-155/UTP5 SBus adapters, respectively.

2.3.2 SunATM-155/MMF SBus Adapter 2.1 Wiring Configuration

The SunATM-155/MMF SBus adapter is shipped with the SC connector already keyed. As you hold the SBus adapter with the connector pointed toward you, “transmit” is on the left and “receive” is on the right.

To connect the SunATM-155/MMF adapter to the network:

- ◆ **Connect one end of the multimode fiber cable into the fiber receptacle on the SBus adapter and connect the other end to the ATM switch (Figure 2-3).**

Refer to the installation or users manual supplied with the hardware interface for additional information.

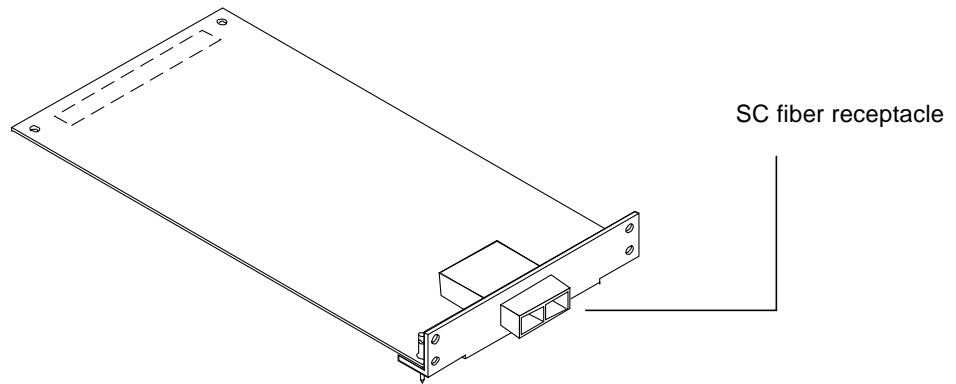


Figure 2-3 SunATM-155/MMF SBus Adapter 2.1



Caution – Do not boot the operating system until the installation is tested and verified.

2.3.3 SunATM-155/UTP5 SBus Adapter 2.1 Wiring Configuration

The SunATM-155/UTP5 SBus adapter is shipped with the RJ45 connector already keyed for “transmit” (Pair 2, pins 1 and 2) and “receive” (Pair 4, pins 7 and 8) in accordance with the EIA/TIA (T568B) wiring scheme (see Appendix A, “Wiring Scheme and Pin Descriptions”).

To connect the SunATM-155/UTP5 SBus adapter to the network:

- ◆ **Plug one end of the Category 5 UTP network cable into the RJ45 receptacle on the SBus adapter and connect the other end to the ATM switch (Figure 2-4).**

Refer to the installation or users manual supplied with the hardware interface for additional information.

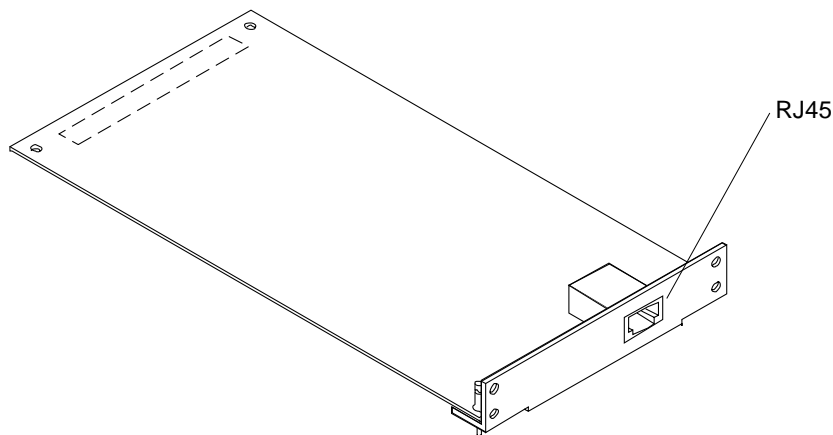


Figure 2-4 SunATM-155/UTP5 SBus Adapter 2.1



Caution – Do not boot the operating system until the installation is tested and verified.

2.4 Testing the SunATM-155 SBus Adapter Before Booting

After you have installed the SunATM-155 SBus adapter, *but before booting the system*, verify installation by executing the `show-devs` command.



Caution – Do not change the SBus slot in which a SunATM-155 SBus adapter is installed once the system has been booted. The Solaris 2.x software environment remembers the location of each SBus adapter that has been installed. Switching SBus slots will cause the operating system to assume that you removed your original SunATM-155 SBus adapter and added a second adapter to the system. Refer to the `path_to_inst` man page for more information.

1. Use the `show-devs` command to display the device information.

The `show-devs device path` command displays all devices known to the system directly beneath a given level in the device hierarchy.

The `show-devs` command used by itself shows the entire device tree. Examples below show information for a SPARCstation 10 system.

Note – The SunATM-155 SBus adapters 2.1 shipped with the SunATM 2.1 software will be identified by the driver name `ba`.

Note – The `/SUNW,ba@3,0` bold entries in the responses to both commands indicate that the system recognizes the SunATM-155 SBus adapter plugged into SBus slot 3.

```
<#0> ok show-devs /iommu/sbus
/iommu@f,e0000000/sbus@f,e0001000/SUNW,ba@3,0
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,bpp@f,4800000
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000/mmcodec
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010/le@f,c00000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/st
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd
<#0> ok
```



```
<#0> ok show-devs
/TI,TMS390Z50@f,f8ffffffc
/eccmemctl@f,0
/virtual-memory@0,0
/memory@0,0
/obio
/iommu@f,e0000000
/openprom
/aliases
/options
/packages
/obio/power@0,a01000
/obio/auxio@0,800000
/obio/SUNW,fdtwo@0,700000
/obio/interrupt@0,400000
/obio/counter@0,300000
/obio/eprom@0,200000
/obio/zs@0,0
/obio/zs@0,100000
/iommu@f,e0000000/sbus@f,e0001000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,ba@3,0
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,bpp@f,4800000
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000/mmcodec
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010/le@f,c000000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/st
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd
/packages/obp-tftp
/packages/deblocker
/packages/disk-label
<#0> ok
```

2. Boot the system.

Refer to the *Solaris 2.x Peripheral's Handbook* for more information.

Installing the SunATM Software



Before installing and configuring the SunATM software, you must first install the SunATM SBus adapter into the system. See Chapter 1, “Introducing and Installing the SunATM-622/MMF SBus Adapter 2.1,” and Chapter 2, “Introducing and Installing the SunATM-155 SBus Adapters 2.1,” for more information.

After you have installed the SunATM software, but *before* you reboot your system, you will need to configure the SunATM software. See Chapter 4, “Configuring the SunATM Interface,” for instructions on how to use the `atmadmin` configuration program.

3.1 Before Installing the SunATM Software

Before installing the SunATM 2.1 software, you must first remove any previous version of the SunATM software that may be present on your system. If you attempt to add the software packages over existing SunATM packages, the installation will fail.

- ◆ **Check for any SunATM software packages by using the `pkginfo` command:**

```
# /usr/bin/pkginfo | grep SUNWatm
```

If you find any SunATM packages, you must remove them.

- ◆ **Remove any existing SunATM software packages by using the `pkgrm` command:**

```
# /usr/sbin/pkgrm SUNWatm SUNWatma SUNWatmu
```

3.2 Installing the SunATM 2.1 Software

3.2.1 Adding the Software Packages Using `pkgadd`

1. **Become superuser (root).**
2. **Insert the SunATM 2.1 CD-ROM into the CD-ROM player connected to your system.**
 - If your system is running the Volume Management software, it should automatically mount the CD on this directory: `/cdrom/sunatm_2_1`
 - If your system is not running the Volume Management software, type the following to mount the CD-ROM:

```
# mkdir /cdrom
# mount -F hsfs -o ro /dev/dsk/c0t6d0s2 /cdrom
```

3. Add the SunATM 2.1 software packages.

- If your system is running the Volume Management software, type:

```
# /usr/sbin/pkgadd -d /cdrom/sunatm_2_1 SUNWatm SUNWatmu SUNWatma
```

- If your system is not running the Volume Management software, and you have mounted the CD-ROM as described in Step 2, type:

```
# /usr/sbin/pkgadd -d /cdrom SUNWatm SUNWatmu SUNWatma
```

Note – For basic ATM functionality, the SUNWatm package is the only required software package. The SUNWatmu package contains the man pages and the files required to configure an ATM Simple Network Management Protocol (SNMP) management system. The SUNWatma package provides the SunATM interim Application Programmers' Interface (API) libraries and header files.

The SunATM packages will be installed in the following directories:

- SunATM Device Drivers and Utilities (SUNWatm) will go into /kernel/mod, /kernel/drv and /etc/opt/SUNWatm
- SunATM Runtime Support Software (SUNWatmu) will go into /opt/SUNWatm
- SunATM Interim API (SUNWatma) will go into /usr/include/atm and /usr/lib

Note – Man pages contained in the SUNWatmu package will go into /opt/SUNWatm/man. (Add this path to your system's \$MANPATH environment variable.) Interim API examples will go into /opt/SUNWatm/examples.

4. Eject the SunATM 2.1 CD-ROM.

- If your system is running the Volume Management software and a window interface, click on the Eject Disk button on the /cdrom/sunatm_2_1 File Manager.
- If your system is running the Volume Management software without a window interface, type:

```
# cd /  
# eject cdrom
```

- If you are not running the Volume Management software, and you mounted the CD-ROM as described in Step 2, type:

```
# cd /  
# umount /cdrom  
# eject cdrom
```

Note – For more information about the Volume Management software, refer to the Solaris documentation.

5. Configure your SunATM interface(s).

You must complete the network configuration of your SunATM interface before you can use the interface. An interactive program, /etc/opt/SUNWatm/bin/atmadmin, is included with the SunATM software, and it may be used to configure your SunATM interfaces. See Chapter 4, “Configuring the SunATM Interface,” for instructions on how to use the atmadmin configuration program.



Caution – You *must* configure the SunATM software before rebooting your system.

6. Perform a reconfiguration boot on your system, and check the network.

See Section 3.3, “Rebooting the System and Examining Network Interfaces,” for more information.

3.2.2 Using the `pkgadd` Utility

When the device on which the package resides is not specified, `pkgadd` checks the default spool directory (`/var/spool/pkg`). If the package is not there, installation fails. The `-d` option allows you to specify a different spool directory, and the name specified after `-d` must be a full pathname to a device or directory (as shown in the examples).

When `pkgadd` encounters a problem, information about the problem is displayed with the following prompt:

```
Do you want to continue with this installation?
```

You should respond with either `yes`, `no`, or `quit`. If more than one package has been specified, `no` stops the installation of the package being installed but informs `pkgadd` to continue with installation of the other packages. `quit` tells `pkgadd` to stop installation of all packages.

Note – For more information about the `pkgadd` utility, refer to the `pkgadd(1M)` man page.

3.2.3 Checking the Package Installation Using `pkgchk`

Once the package is installed, you can use the `pkgchk` command to see if the installation was complete:

```
# /usr/sbin/pkgchk SUNwatm
```

Multiple packages can be specified at the command line by separating the package names with a space. If no package identifier is specified, the entire contents of the machine are checked.

3.2.4 Checking the SunATM Software Installation Using `pkginfo`

Check the ATM software installation by using the `pkginfo` command:

```
# /usr/bin/pkginfo | grep SUNWatm
system      SUNWatm      SunATM Device Drivers
application SUNWatma     SunATM Interim Api Support Software
application SUNWatmu     SunATM Runtime Support Software
```

3.2.5 Removing the Software Packages Using `pkgrm`

You can remove one or more packages with the following command:

```
# /usr/sbin/pkgrm SUNWatm SUNWatma SUNWatmu
```

In this example, `pkgrm` removes the packages identified as `SUNWatm` (SunATM Device Drivers and Utilities), `SUNWatma` (SunATM Interim API Support Software), and `SUNWatmu` (SunATM Runtime Support Software).

3.3 Rebooting the System and Examining Network Interfaces

1. Reboot the system using the `boot -r` command.

The `-r` option is required by the Solaris software environment when installing new hardware. Use the `-v` option to display the boot messages, so you can see that the SunATM adapters are recognized correctly.

To start the boot process from the `ok` prompt, type:

```
ok boot -rv
```

Note – For Solaris 2.x, use `boot -r` whenever the physical configuration of the system is changed. Refer to the `boot(1M)` man page for more information.

2. Execute `ifconfig -a` and `netstat -i` commands to examine the state of all network interfaces.

You can also use `/usr/sbin/ping` or `/usr/sbin/spray` commands to see if a network interface is active.

The following are examples of `ifconfig -a`, `ping`, and `netstat -i` output. Refer to the `ifconfig(1M)`, `ping(1M)`, `spray(1M)`, and `netstat(1M)` man pages for more information.

```
example% /sbin/ifconfig -a
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
      inet 127.0.0.1 netmask ff000000
ba0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 9180
      inet 129.144.130.9 netmask ffffffff broadcast 129.144.130.255
      ether 8:0:20:75:89:ff
```

```
example% /usr/sbin/ping zelda
zelda is alive
```

```
example% netstat -i
Name      Mtu      Net/Dest      Address      Ipkts  Ierrs  Opkts  Oerrs  Collis Queue
lo0       8232     loopback     localhost    1      0      1      0      0      0
ba0       9180     umtv20-130-n zardoz      5875   0      382812 0      0      0
```



Caution – Do not change the SBus slot in which a SunATM SBus adapter is installed once the system has been booted. The Solaris 2.x software environment remembers the location of each SBus adapter that has been installed. Switching SBus slots will cause the operating system to assume that you removed your original SunATM SBus adapter and added a second adapter to the system. Refer to the on-line man page about `path_to_inst` for more information.

Configuring the SunATM Interface



This chapter describes the new features in the SunATM 2.1 software and how to configure the software using the `atmadmin` configuration program.

4.1 SunATM SBus Adapter Versions

The SunATM 2.1 software supports some earlier versions of the SunATM-155 SBus adapters. Table 4-1 shows the hardware supported with each software release.

Table 4-1 Supported SunATM SBus Adapters

Software version	SunATM-155 SBus Adapter 1.0	SunATM-155 SBus Adapter 2.0	SunATM-155 SBus Adapter 2.1	SunATM-622 SBus Adapter 2.1
SunATM 1.0	Supported			
SunATM 2.0	Supported	Supported		
SunATM 2.1		Supported	Supported	Supported

The SunATM-622/MMF SBus Adapter 2.1, which is supported by the SunATM 2.1 software, will use the `ba` driver. Specific interfaces will be referred to as `baN`, where *N* is the instance number. The SunATM-155 SBus adapters 1.0 use the `sa` driver, which is *not* supported in the SunATM 2.1 software.

4.2 *New Features in the SunATM 2.1 Software*

Besides supporting the new SunATM SBus adapters 2.1, the SunATM 2.1 software contains several new features. This section provides a brief overview of these new features.

4.2.1 *Configuring Network Parameters*

The SunATM software modules read configuration information from these files in the `/etc` directory: `atmconfig`, `aarconfig` (for Classical IP), and `laneconfig` (for LAN Emulation). In previous releases of the SunATM software, you were required either to enter the configuration information during the software package installation, or to edit the files manually after the package installation. The SunATM 2.1 software removes the package installation configuration questions, and introduces an interactive tool that will allow you to configure the software at any time. See Section 4.3, “Using the `atmadmin` Configuration Program,” for instructions on how to use the `atmadmin` configuration program.

4.2.2 *Support for the Full ATM MIBs in ILMI*

In previous versions, the SunATM ILMI daemon, `ilmid`, only supported the address registration portion of the ATM Forum-specified Management Information Base (MIB) for ATM. In SunATM 2.1, `ilmid` now supports the full MIB, and will respond appropriately to requests from the switch.

4.2.3 *SNMP Forwarding Agent*

The ATM SNMP agent is now able to act as a forwarding agent, allowing multiple SNMP agents to run on a single system. This functionality is described in more detail in Section 6.4, “ATM and SNMP.”

4.2.4 *Logical Interface Names*

This new feature applies only in the LAN Emulation environment. A LAN Emulation interface may be configured to have multiple IP addresses, each associated with a unique logical interface name. This functionality and configuration of logical interfaces are described in “Additional IP Addresses” on page 4-20 and Section 5.4, “Supporting Logical Interfaces.”

4.2.5 *Joining Multiple Emulated LANs*

A single interface can now join multiple Emulated LANs (ELANs) — up to a total of sixteen. To avoid problems that might arise in a large network using bridging, a unique MAC address must be assigned to each LAN Emulation instance which joins an ELAN. Therefore, this feature requires that the hardware be assigned multiple MAC addresses. The new SunATM-155 SBus adapters 2.1 and the SunATM-622 SBus adapter 2.1 have sixteen assigned MAC addresses. Section 5.5, “Supporting Multiple Emulated LANS on a Single Interface,” provides more information about this feature and how to create the appropriate configuration files (see Section 4.4.5, “LAN Emulation Parameter Group” for additional information).

Note – This feature requires the new SunATM 2.1 adapter boards. It will not work on the older 2.0 adapters.

4.2.6 *Mounting the /opt directory Over an ATM Interface*

In previous releases of the SunATM software, the utilities required to bring up a SunATM interface were installed in the `/opt/SUNWatm/bin` directory, thus requiring that the system’s `/opt` filesystem be mounted, either locally or over a non-ATM network interface, before the SunATM interfaces could be configured. In the SunATM 2.1 software, these utilities reside in the `/etc/opt/SUNWatm/bin` directory, which is part of the root filesystem. In addition, the SunATM start-up script has been moved to `S00sunatm`, so that the SunATM interfaces will be configured before the `/opt` and other filesystems are mounted.

4.2.7 *Per-Adapter Framing Interface Selection*

Previous releases of the SunATM software required that the same framing interface (SONET or SDH) apply to all interfaces installed in a system. The SunATM 2.1 software allows different framing interface selections on each physical interface. See Section 4.4.1, “Physical Layer Parameter Group,” for a description of how to change the framing interface. The default framing interface is SONET.

4.3 *Using the atmadmin Configuration Program*

The SunATM configuration program, `atmadmin`, is an interactive command-line interface. The program contains a hierarchy of menus, which divide the configuration into six main parameter groups: system, physical layer, signalling, ILMI, Classical IP, and LAN Emulation. All but the system parameter group are specific to individual SunATM interfaces, so you must configure the parameters in that group separately for each interface.

If you prefer, you may enter and change the SunATM configuration information by editing the SunATM configuration files directly. See Chapter 5, “Editing the SunATM Configuration Files,” for a description of the configuration files’ contents and formats.

Note – See the Glossary for descriptions of the ATM and SunATM terms used in this chapter. Chapter 5, “Editing the SunATM Configuration Files,” and Chapter 6, “Classical IP and LAN Emulation Protocols,” also provide more information about ATM protocols and the SunATM implementation of these protocols.

4.3.1 *Starting the atmadmin Configuration Program*

The `atmadmin` program is installed with the SUNWatm software package in the `/etc/opt/SUNWatm/bin` directory. The program must be run as superuser (root), and it has no command-line options. It may be run in any local or remote shell on the SunATM system.

```
# /etc/opt/SUNWatm/bin/atmadmin
```

4.3.2 atmadmin *Main Menu*

After you have started the atmadmin configuration program, you will see the atmadmin Main Menu. From this menu you can either go to the system parameter group menu (see Section 4.3.2.1, “System Parameter Group Menu”), or you can enter the SunATM interface you want to configure. Figure 4-1 displays an example of a system with one interface named ba0.

```
Welcome to the SunATM Admin Program.
The following interfaces are installed in your system:
    ba0
    [S] System
    [X] Exit
    [?] Help

Enter Selection: ba0
```

Figure 4-1 atmadmin Main Menu

After selecting an interface, you will then see the Interface Configuration menu (see Section 4.3.2.2, “Interface Configuration Menu”).

Common Navigation Commands

Some basic commands are recognized throughout the menu hierarchy, and they may be used to navigate through the various menus. Table 4-1 lists these navigational commands.

Table 4-2 Basic Navigational Commands in atmadmin

Command	Action
m	Return to the atmadmin main menu.
p	Return to the previous menu.
x	Exit atmadmin.
?	Provide more information about the options on this menu.

4.3.2.1 System Parameter Group Menu

The system parameter group contains parameters that are not interface-specific; they apply to the entire system. Figure 4-2 shows the system parameter group menu.

```
Modifying system-wide parameters;
Currently configured as an ATM SNMP agent
The SNMP agent options are:
    [A] ATM SNMP agent
    [N] not an agent

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter selection:
```

Figure 4-2 System Parameter Group Menu

ATM SNMP Agent Status

Your may configure your SunATM system as an ATM SNMP agent. However, only one SNMP agent will work on a system, because SNMP agents use a standard UDP port where only one agent can bind. If your system contains another SNMP agent, the ATM agent will either not work or it will cause the other agent not to work.

The SunATM SNMP daemon, `atmsnmpd`, will always run on an ATM host. If you choose not to run your system as an SNMP agent, the daemon will not bind to the SNMP port, port 161, allowing another SNMP agent to use that port.

Note – See Section 6.4, “ATM and SNMP,” on page 6-10 for more information about `atmsnmpd` options.

4.3.2.2 Interface Configuration Menu

Once you have selected a SunATM interface, you will see the `atmadmin` Interface Configuration menu (Figure 4-3). From this menu you will be able to proceed to the interface parameter group sub-menus, which are described in Section 4.4, “`atmadmin` Parameter Groups.” Within these sub-menus, you will be able to change the SunATM interface configuration parameters.

```
Modifying ba0
  [Y] Physical Layer
  [U] UNI Signalling
  [I] ILMI Address Registration
  [C] Classical IP
  [L] LAN Emulation

  [P] Previous Menu
  [M] Main Menu
  [X] Exit
  [?] Help

Enter selection:
```

Figure 4-3 Interface Configuration Menu

4.3.3 `atmadmin` and the SunATM Configuration Files in the `/etc` directory

The `atmadmin` program will first attempt to read the current configuration information from the `/etc/atmconfig`, `/etc/aarconfig`, and `/etc/laneconfig` files. If no configuration information is found, or if the files do not exist, the default values listed in Table 4-3 will be applied to the installed interfaces.



Caution – When saving configuration information, `atmadmin` will overwrite the existing SunATM configuration files in the `/etc` directory. Therefore, any comments or other changes you manually made to the files will be lost.

4.4 atmadmin *Parameter Groups*

The atmadmin configuration program contains a series of menus where you can input or alter the configuration of specific SunATM software parameters. These menus, or parameter groups, are described in the following sections in this chapter:

<i>Physical Layer Parameter Group</i>	<i>page 4-10</i>
<i>Signalling Parameter Group</i>	<i>page 4-11</i>
<i>ILMI Parameter Group</i>	<i>page 4-12</i>
<i>Classical IP Parameter Group</i>	<i>page 4-12</i>
<i>LAN Emulation Parameter Group</i>	<i>page 4-16</i>

Table 4-3 summarizes the configurable parameters in each parameter group. Although the parameter list appears rather lengthy, you will only need to use the default values for most standard configurations. The large number of parameters provide the flexibility to support a wide variety of special case configurations, and to allow interoperability with a broad a range of equipment from other vendors.

Note – In most cases, you will only need to configure the parameters which do not have default values.

Table 4-3 Configurable Parameters in the SunATM Software

Group	Parameters	Possible Values	Default Values	Required?
System	SNMP Agent Status	agent or not_agent	not_agent	Yes
Physical Layer	Framing Interface	SONET or SDH	SONET	Yes
Signalling	UNI Version	3.0, 3.1, or none	No default	Yes
ILMI	Use ILMI	Yes or no	Yes	Yes
Classical IP	IP Hostname/Address	Valid IP hostname and address	No default	For Classical IP
	Interface Type	Client, Server, or Standalone	No default	For Classical IP
	Local ATM Address	Valid ATM address	\$myaddress	For Classical IP Clients or Servers

Table 4-3 Configurable Parameters in the SunATM Software (Continued)

Group	Parameters	Possible Values	Default Values	Required?
	ARP Server	Valid ATM address	\$localswitch_server	For Classical IP Clients
	PVC	32 <= n <= 1024	32	For Classical IP Standalones
LAN Emulation	Instance Number	0 <= n <= 999	No default	For LAN Emulation
Per-Instance Parameters	IP Hostname/Address	Valid IP hostname and address	No default	For IP over LAN Emulation
	Local ATM Address	Valid ATM address	\$myaddress	For LAN Emulation
	LECS Indicator	no_lecs, or lecs_present	lecs_present	For LAN Emulation
	LECS ATM Address	Valid ATM address	The well-known LECS address	For LAN Emulation, lecs_present
	LES ATM Address	Valid ATM address	No default	For LAN Emulation, no_lecs
	Emulated LAN Name	Character string	No default	For additional instance on a physical interface
	Additional IP Address?	Yes or no	No	For LAN Emulation
Per-Additional IP Address	Minor Instance Number	0 <= n <= 255	None	For LAN Emulation, additional IP
	IP Hostname/Address	Valid IP hostname and address	No default	For LAN Emulation, additional IP

4.4.1 Physical Layer Parameter Group

The physical layer parameter group contains only the framing interface parameter. Figure 4-4 shows the physical layer parameter menu.

```
Modifying ba0; Current framing interface is SONET
The framing interfaces that may be configured are:
    sonet
    sdh

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter selection:
```

Figure 4-4 Physical Layer Parameter Menu

Framing Interface

The framing interface defines the encapsulation method used for ATM cells as they are sent onto the wire. The default framing interface is SONET, but the SunATM software also supports the SDH interface. Your switch product information should indicate whether your switch uses either the SONET or the SDH interface. If the switch uses the SDH interface, you will need to select SDH from this menu.

4.4.2 Signalling Parameter Group

The signalling parameter group contains only the UNI version parameter. Figure 4-5 shows the signalling parameter menu.

```
Modifying ba0; Current UNI Version is 3.0
The UNI versions that may be configured are:
    3.0
    3.1
    [N] No Signalling Enabled

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter selection:
```

Figure 4-5 Signalling Parameter Menu

UNI Version

The SunATM software supports two versions of the ATM Forum's User Network Interface (UNI) Specification: versions 3.0 and 3.1. You may choose not to enable signalling, but in order to support either Classical IP or LAN Emulation (or both), you must select one of the two UNI versions.

4.4.3 ILMI Parameter Group

If your ATM switch does not support Interim Local Management Interface (ILMI), you can turn off the ILMI address registration on your SunATM interface from the ILMI configuration menu. Figure 4-5 shows the ILMI configuration menu.

```

Modifying ba0; Currently ILMI is enabled
  [E] Enable ILMI
  [D] Disable ILMI

  [P] Previous Menu
  [M] Main Menu
  [X] Exit
  [?] Help

Enter selection:

```

Figure 4-6 ILMI Configuration Menu

4.4.4 Classical IP Parameter Group

Classical Internet Protocol (Classical IP), specified by RFC 1577, is one way of supporting the TCP/IP and UDP/IP protocols in an ATM environment. In Classical IP, an ATM ARP server is used to resolve IP addresses to ATM addresses, replacing the traditional ARP protocol. In this configuration each host must register with the ARP server when the ATM interface is brought up. For more information on the Classical IP protocols, see Section 6.2, “Classical Internet Protocol.”

One major reason for the use of ATM ARP instead of the traditional ARP is because ATM does not support broadcast (a network capability providing transmission from one point to all points on a network). Because Classical IP over ATM does not support broadcast, you cannot use the `ypbind` UNIX command with the `-broadcast` option to automatically locate the NIS server (`ypserver`) on a Classical IP ATM subnet.

If you are planning to run NIS over your ATM network, you must specify the list of NIS servers (`ybservers`) using the `ypinit -c` command. See the `ypinit(1M)` man page for details of setting up the `ypserver`. Be sure that the IP addresses of the `ybservers` are listed in the `/etc/hosts` file.

Since Classical IP does not currently support the multicast packet delivery system, hosts cannot use the `in.rdisc` command to locate routers on the ATM subnet. The `in.rdisc` command uses IP multicasting to automatically locate routers and to pick the best router among many. Classical IP Hosts cannot use Routing Information Protocol (RIP) (`in.routed`) because RIP uses the broadcast packet delivery system. You must explicitly add the routes to the routers in the ATM subnet. You may also specify one router as the default router to provide connectivity outside of the ATM subnet. See the `route(1M)` man page for information on using the `route` command to add specific router entries and to add a default router.

Several parameters define the Classical IP configuration of a SunATM interface, and all of these parameters can be configured through the Classical IP parameter group menu (Figure 4-7).

```
Modifying ba0; Current Configuration:
  Arp Client
  IP = atm_cip
  ATM = $myaddress
  ARPSRV = $localswitch_server
    [N] No Classical IP Enabled
    [C] Client
    [S] Arp Server
    [T] Standalone
    [I] IP Hostname or Address
    [L] Local ATM Address
    [A] ATM ARP Server Address

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter Selection:
```

Figure 4-7 Classical IP Parameter Menu

4.4.4.1 *Classical IP Interface Type*

The SunATM software allows you to configure your interface as either a Classical IP ARP server or a client. In addition, you can connect two systems back-to-back, in a standalone configuration, using a Permanent Virtual Circuit (PVC). Each of these three modes appear as options on the Classical IP parameter menu.

4.4.4.2 *IP Hostname and Address*

Regardless of the Classical IP Interface Type, you must assign an IP address and hostname to the interface. If you enter a hostname that appears in the `/etc/hosts` file (or if NIS is enabled and the hostname is resolvable over NIS), you will not be prompted to enter an IP address. Instead, the resolution will be performed automatically. If the hostname cannot be resolved, you will be prompted to enter an IP address. If you must enter an IP address, or if the address is only available through NIS, the SunATM software will update the `/etc/hosts` file.

A valid IP hostname must be no more than 80 characters in length. A valid IP address must be a dot-separated set of four decimal numbers in the range of 0 to 255 (for example, 149.144.130.9).

4.4.4.3 *Local ATM Address*

The local ATM address is the 20-byte ATM address associated with this Classical IP instance. You must assign an ATM address to each Classical IP client and server, but you do not need to assign an ATM address on standalone (back-to-back) configurations. The following section describes ATM address formats and some of the SunATM software defined address variables.

ATM Address Formats and Variables

ATM addresses, like Network Service Access Point (NSAP) addresses, are 20 octets long, with each octet made up of 1 or 2 hexadecimal digits. The ATM address is divided into three fields: the End System Identifier field, Selector field, and the Network Prefix field. The End System Identifier (ESI) field is a unique 6 octet value, which can be the IEEE hardware MAC address conventionally associated with every network interface. The Selector field is one octet long. The 13 octets that make up the rest of the ATM address are

called the Network Prefix. This field should be derived from the ATM switch fabric to which the interface is connected. Every ATM switch fabric is configured with a 13 octet prefix.

To simplify references to ATM addresses in the SunATM software, several system-defined variables have been built into the software. Variables are referenced with the `$` operator, as in UNIX shell scripts. Table 4-4 summarizes the system-defined SunATM ATM address variables.

Table 4-4 Predefined SunATM Variables

Variable	Description
<code>prefix</code>	The 13-byte prefix associated with the local switch.
<code>mac</code>	The 6-byte medium access control (MAC) address associated with the local host or interface.
<code>sel</code>	The default 1-byte selector for the local interface.
<code>macsel</code>	The concatenation of <code>\$mac:\$sel</code> .
<code>myaddress</code>	The concatenation of <code>\$prefix:\$mac:\$sel</code> , resulting in the default address for the local interface.
<code>sunmacselN</code>	The concatenation of one of a series of reserved MAC addresses and <code>\$sel</code> to create a block of reserved ATM ARP server addresses. <i>N</i> should be a decimal number in the range 0 - 199.
<code>localswitch_server</code>	The concatenation of <code>\$prefix</code> , a unique reserved MAC address, and <code>\$sel</code> . When used as a server address, restricts server access to clients connected to the local switch only.

Note - The `$prefix` variable, and any other variables which use it (including `$myaddress` and `$localswich_server`), may not be used on interfaces which are not running ILMI.

ATM addresses are represented by 20 colon-separated octets, with each octet made up of 1 or 2 hexadecimal digits. You may combine variables representing portions of an ATM address with other variables and/or octets to make up a complete address. For example, `$prefix:aa:bb:cc:dd:ee:ff:$sel` represents a valid ATM address.

4.4.4.4 *Server ATM Address*

If you configured the Classical IP instance as a client, you must also enter the address of the ARP server. This parameter, like the local ATM address, must be a 20-byte ATM address. See “ATM Address Formats and Variables” on page 4-14 for a discussion of ATM address formats and variables.

4.4.4.5 *Permanent Virtual Circuit*

The Permanent Virtual Circuit parameter applies only to standalone configurations. It identifies the permanent virtual circuit (PVC) which will be used to communicate between the two systems connected back-to-back. Both systems must use the same PVC value. The PVC parameter must be a decimal integer between 32 and 1024.

4.4.5 *LAN Emulation Parameter Group*

LAN Emulation, standardized by the ATM Forum’s LAN Emulation 1.0 specification, is another way of providing TCP/IP and UDP/IP support over an ATM interface. Address resolution information is provided by a series of LAN Emulation services. When a LAN Emulation interface is brought up it must register with these LAN Emulation services (known as “joining the LAN”). This registration process and the address resolution process are described in Section 6.3, “LAN Emulation,” on page 6-5.

Unlike Classical IP, the LAN Emulation protocol provides a broadcast service to the upper layer protocols. Therefore, the multicast and RIP limitations described in Section 4.4.4, “Classical IP Parameter Group,” do not affect LAN Emulation interfaces.

The SunATM 2.1 software allows a single ATM interface to join up to sixteen emulated local area networks (ELANs), provided that this action is allowed by the switch and LAN Emulation (LANE) services. Each ELAN joined will be represented by a unique lane instance (e.g. lane0 or lane1).

Note – A requirement for supporting this feature is that the adapter card be assigned multiple MAC addresses, which is supported in the SunATM-622 SBus adapters 2.1 and the new SunATM-155 SBus adapters 2.1. This feature *WILL NOT* work with the older SunATM SBus Adapters 2.0. You can find the number of MAC addresses assigned to your SunATM adapter by using the `atmgetmac(1M)` command with the `count` option.

After choosing to configure LAN Emulation parameters, you will be asked to choose an existing (previously configured) LAN Emulation (`lane`) instance, or to create a new one in the LAN Emulation Instance menu. Figure 4-8 shows an example of this menu.

```
The following lane instances are configured on ba0:
    lane0
    lane1
    [C] Create new lane instance
    [D] Delete lane instance

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter selection:
```

Figure 4-8 LAN Emulation Instance Menu

4.4.5.1 Per-Instance LAN Emulation Parameters

The next menu, the Per-Instance LAN Emulation Parameters menu (Figure 4-9), allows you to configure the per-instance LAN Emulation parameters.

```

Modifying lane0; Current Configuration:
  IP = atm_lane
  ATM = $myaddress
  LECS Present
  LECS_Address = well-known address
  no additional IP hostnames
    [I] IP Hostname or Address
    [L] Local ATM Address
    [C] LECS Present
    [N] No LECS
    [A] LECS ATM Address
    [S] LES ATM Address
    [E] Emulated LAN Name
    [H] Additional IP Hostnames

    [P] Previous Menu
    [M] Main Menu
    [X] Exit
    [?] Help

Enter Selection:
  
```

Figure 4-9 LAN Emulation Per-Instance Parameters Menu

IP Hostname and Address

If IP traffic is to run over a LAN Emulation instance, you must assign an IP hostname and corresponding address to the instance. If you enter a hostname that appears in the `/etc/hosts` file (or if NIS is enabled and the hostname is resolvable over NIS), you will not be prompted to enter an IP address. Instead, the resolution will be performed automatically. If the hostname cannot be resolved, you will be prompted to enter an IP address. If you must enter an IP address, or if the address is only available through NIS, the SunATM software will update the `/etc/hosts` file.

A valid IP hostname must be no more than 80 characters in length. A valid IP address must be a dot-separated set of four decimal numbers in the range of 0 to 255 (for example, 149.144.130.9).

Local ATM Address

The local ATM address is the 20-byte ATM address associated with this LAN Emulation instance. See “ATM Address Formats and Variables” on page 4-14 for more information about ATM address formats and variables.

Each `lane` instance must be assigned a unique ATM address. Each SunATM 2.1 board has been assigned 16 unique MAC addresses; if you use the variable `$myaddress` for each `lane` instance, the SunATM software will automatically distribute those MAC addresses to the `lane` instances associated with each physical interface.

LECS Indicator

Most LAN Emulation Services include a LAN Emulation Configuration Server (LECS) which is the first server contacted when bringing up a LAN Emulation client. The LECS will provide the ATM address of the LAN Emulation Server (LES), as well as other configuration information about the emulated LAN. However, some LAN Emulation services do not include an LECS, and the LES must be contacted directly. With the LECS Indicator parameter, you can specify which service should be contacted first in your configuration. The possible values for this parameter are displayed as individual options on the LAN Emulation Instance menu.

Note – If the value of this parameter is `no_LECS`, a value must be given for the LES ATM Address parameter.

LECS ATM Address

The ATM Forum specifies a “well-known” ATM address for the LECS. The SunATM software uses this address to contact the LECS. If your LECS uses a different ATM address, you should specify it in this parameter. If applicable, any of the ATM address variables described in “ATM Address Formats and Variables” on page 4-14, `$prefix` in particular, may be used.

LES ATM Address

This parameter is required if the value of the LECS Indicator parameter is `no_LECS`. There is no “well-known” address for the LES, so an ATM address must be specified for the LES if there is not a LECS present to provide one. This parameter is a standard ATM address. If any of the SunATM ATM address variables such as `$prefix`, described in “ATM Address Formats and Variables” on page 4-14, are applicable, they may be used.

Emulated LAN Name

If multiple Emulated LANs (ELANs) are present, you can enter a character string in the Emulated LAN Name parameter. The LAN Emulation client may use this parameter to indicate to the LAN Emulation services which ELAN it wishes to join. By default, if a SunATM LAN Emulation client does not specify an ELAN name, it tells the services to assign it to the default (or only) ELAN.

Note – If you have multiple LAN Emulation instances configured on a physical interface, only one instance may join the default (unspecified) ELAN. You must specify ELAN name for all other instances.

Additional IP Addresses

The SunATM 2.1 software supports logical interfaces in the SunATM LAN Emulation environment. Logical interfaces allow you to assign multiple IP addresses to a single LAN Emulation interface. A logical interface name consists of three parts: the device name (in the case of SunATM LAN Emulation, `lane`); the major number, which corresponds to the `lane` instance number; and the minor number, which distinguishes the logical interfaces on a single `lane` instance. The format of a LAN Emulation logical interface name is `laneN:X`, where `N` is the major number and `X` is the minor number (for example, `lane0:2`).

The SunATM software will associate each logical interface with a unique IP hostname and address. All logical interfaces on a given physical interface will be associated with the same ATM and MAC addresses.

The IP hostname displayed in the LAN Emulation instance menu corresponds to the minor instance 0. The additional IP Address parameter will indicate if any additional IP hostnames are assigned to the instance. Additional IP hostnames may be modified and/or created by selecting this parameter. You

must enter or modify each additional IP hostname in the same manner as other IP hostname and address pairs (see “ATM Address Formats and Variables” on page 4-14 for more details), and you must associate it with a minor number between 0 and 255.

Editing the SunATM Configuration Files

This chapter describes how you can configure your SunATM interfaces by editing the configuration files.

You are not required to edit these configuration files by hand. You can use the `atmadmin` configuration program, described in Section 4.3, “Using the `atmadmin` Configuration Program,” to configure the SunATM files. From the program’s command-line interface, you can change most of the SunATM parameters, with the only exception being the access list security feature described in Section 5.2, “Configuring a Classical Internet Protocol Interface” and Section 5.3, “Configuring a LAN Emulation Interface.”



Caution – When saving configuration information, `atmadmin` *will overwrite* the existing SunATM configuration files in the `/etc` directory. Therefore, any comments or other changes you manually made to the files will be lost.

5.1 Editing the `/etc/atmconfig` File

The `/etc/atmconfig` file is a generic file that must appear on every SunATM system. It provides general configuration information that is used by the SunATM startup script to bring up ATM interfaces at boot time.

The file consists of one or more entries per interface. An entry contains the following fields:

Table 5-1 `/etc/atmconfig` Field Descriptions

Field	Description
Interface	The physical interface, <code>baN</code> .
UNI/Framing	The version of the UNI specification used for signalling, 3.0 or 3.1; or, for entries with only two fields, the Framing Interface, SONET or SDH.
CIP_Host	The IP hostname used for Classical IP.
LANE_Instance	The instance number for a LAN Emulation interface; LAN Emulation interfaces will be called <code>laneN</code> where <code>N</code> is the LAN Emulation instance number. The LANE instance number must be between 0 and 999. Note: The LANE instance number is not necessarily the same as the physical instance number.
LANE_Host	The IP hostname used for LAN Emulation

The Interface and UNI fields are required for all interfaces. The CIP_Host field is required for interfaces that run Classical IP, and the LANE_Instance and LANE_Host fields are required for interfaces that run LAN Emulation. Unused fields are represented by a hyphen.

Because the `atmconfig` file contains information about how an interface is initially configured, the system must be rebooted in order for changes made in the `/etc/atmconfig` file to take effect.

5.1.1 Changing the Framing Interface in the `/etc/atmconfig` File

The framing interface defines the encapsulation method used for ATM cells as they are sent onto the wire. The default framing interface is SONET, but the SunATM software also supports the SDH interface. Your switch product information should indicate whether your switch uses either the SONET or the SDH interface.

Previous versions of the SunATM software allowed a framing interface to be chosen for the entire system (the selection was made by setting a variable in the `/etc/system` file). In the SunATM 2.1 software, the system variable may still be used to allow backwards compatibility, but the preferred method is to select the framing interface per-interface, with an entry in the `/etc/atmconfig` file. Entries in `/etc/atmconfig` will override a variable set in `/etc/system` for a particular interface. If there is no value in either `/etc/system` or `/etc/atmconfig`, the default framing interface is SONET.

Framing entries in `/etc/atmconfig` should appear on individual lines, with two fields. The first field indicates the interface, `baN`, where `N` is the instance number (for example: `ba0`). The second is either `SDH` or `SONET`, depending on the desired setting. See Figure 5-1 for an example of selecting SDH in an `/etc/atmconfig` file.

5.1.2 Example of an `/etc/atmconfig` File

The following sample `atmconfig` file creates this configuration:

- A LAN Emulation interface `lane0`, supporting UNI 3.1, on the `ba0` interface.
- An interface that supports both Classical IP and LAN Emulation on `ba1`, using UNI3.1. The LAN Emulation interface name is `lane1`.
- A Classical IP interface, supporting UNI 3.0, on `ba2`, which uses the SDH framing interface.

#Interface	UNI/Framing	CIP_Host	LANE_Instance	LANE_Host
ba0	3.1	-	0	atm0
ba1	3.1	atm1	1	atm2
ba2	3.0	atm3	-	-
ba2	SDH			

Figure 5-1 Example `/etc/atmconfig` File

5.2 *Configuring a Classical Internet Protocol Interface*

Classical Internet Protocol (Classical IP), specified by RFC 1577, is one way of supporting the TCP/IP and UDP/IP protocols in an ATM environment. In Classical IP, an ATM ARP server is used to resolve IP addresses to ATM addresses, replacing the traditional ARP protocol. In this configuration each host must register with the ARP server when the ATM interface is brought up. For more information on the Classical IP protocols, see Section 6.2, “Classical Internet Protocol.”

One major reason for the use of ATM ARP instead of the traditional ARP is because ATM does not support broadcast (a network capability providing transmission from one point to all points on a network). Because Classical IP over ATM does not support broadcast, you cannot use the `yplib` UNIX command with the `-broadcast` option to automatically locate the NIS server (`yserver`) on a Classical IP ATM subnet.

If you are planning to run NIS over your ATM network, you must specify the list of NIS servers (`yserver`s) using the `ypinit -c` command. See the `ypinit(1M)` man page for details of setting up the `yserver`. Be sure that the IP addresses of the `yserver`s are listed in the `/etc/hosts` file.

Since Classical IP does not currently support the multicast packet delivery system, hosts cannot use the `in.rdisc` command to locate routers on the ATM subnet. The `in.rdisc` command uses IP multicasting to automatically locate routers and to pick the best router among many. Classical IP Hosts cannot use Routing Information Protocol (RIP) (`in.routed`) because RIP uses the broadcast packet delivery system. You must explicitly add the routes to the routers in the ATM subnet. You may also specify one router as the default router to provide connectivity outside of the ATM subnet. See the `route(1M)` man page for information on using the `route` command to add specific router entries and to add a default router.

5.2.1 *Editing the /etc/aarconfig File*

The `/etc/aarconfig` file is a generic file that must appear on every SunATM system which is supporting Classical IP interfaces. It allows you to specify IP to ATM address translation, permanent virtual circuits (PVCs) to destinations, and specify the address of the ATM ARP server. The environment allows for a mix of PVCs and switched virtual circuits (SVCs).

Each time the `/etc/aarconfig` file is modified, you must run the ATM ARP setup program (`aarsetup`). `aarsetup` is in the `/etc/opt/SUNWatm/bin` directory.

Every node, or client, will have both an IP address and either an ATM address or a virtual circuit identifier (VCI). See Section 6.2.1, “ATM Address Resolution,” for ATM addressing scheme information.

Table 5-3 on page 5-7 lists the flags and the options they provide. All the configuration flags are described in this section.

In the IP-ATM address table shown in the `/etc/aarconfig` file:

- `Interface` is the last part of the device name in `/dev` (`ba0`, for example).
- `Hostname` is either an IP address in “dot” notation or the name of a host that should be locally available unless a non-ATM network connection also exists.
- `ATM Address` consists of 20 octets with each octet represented by a one- or two-digit hexadecimal number and separated by colons.
- The `VCI` field is a positive decimal integer.
- An unused field is denoted by a hyphen

Table 5-2 `/etc/aarconfig` File Flags

Flag	Description
l	Represents the ATM address of the local interface on ARP clients or systems not using an ARP server for ATM address resolution, and can be used to assign an ATM address to the host. <i>Hostname</i> should not appear; <i>ATM Address</i> should be provided if and only if SVCs are used. If an <i>s</i> entry is provided to use an ARP server (see below), <i>ATM Address</i> must be provided (a server is meaningful only in an SVC environment). See Table 5-3.
L	Represents the IP and ATM address of the local interface on an ARP server. <i>Hostname</i> should not appear; <i>ATM Address</i> is required. See Table 5-3.
s	Specifies a connection to the ATM ARP server. Either <i>ATM Address</i> or <i>VCI</i> (in the case of a PVC connection) should appear, but not both. <i>Hostname</i> should not appear. The <i>s</i> entry is required on all clients that need to communicate with the server for ATM address resolution. See Table 5-3

Table 5-2 /etc/aarconfig File Flags (Continued)

Flag	Description
t	<p>Represents an IP to ATM address/VCI entry. <code>aarsetup</code> adds these entries into the local table. Any <i>t</i> entries on the server must contain <i>ATM Address</i> and may also contain <i>VCI</i> if PVC communication between the server and client is desired. In addition, there are some cases when a <i>t</i> entry may be useful on an ARP client system. If a client wants to communicate with another system over PVCs, the PVC to be used is provided in a <i>t</i> entry containing <i>VCI</i>; or if a client wishes to cache frequently used addresses to avoid frequent ARP requests, a <i>t</i> entry containing <i>ATM Address</i> may be provided. See Table 5-3.</p> <p>Note: If your naming service (NIS, DNS) server is an ATM host, you must provide the IP hostname to the address resolution for the hosts included in <i>t</i> entries, either by using the IP address in the <i>Hostname</i> field of the <i>t</i> entry, or by adding an entry to the local <code>/etc/hosts</code> file.</p>
a	<p>Represents an address that may have access to this host. If no <i>a</i> entries appear in the <code>aarconfig</code> file, access to the host is unrestricted. Including <i>a</i> entries allows access to be restricted to known hosts. As an alternative to listing individual addresses, the ATM address field may contain a prefix, followed by the wildcard <code>\$anymacsel</code>, which matches any 7-byte ESI/Selector combination following the given prefix. This allows access by any host connected to the switch specified by the given prefix. <i>Hostname</i> and <i>VCI</i> should not appear; <i>ATM Address</i> is required. See Table 5-3.</p>
m	<p>Notifies the system that the entire ATM address, including the network prefix, must be configured manually on this interface. If your interface is connected to a switch that does not support ILMI, you must include this option in your <code>/etc/aarconfig</code> or <code>/etc/lanconfig</code> file. Note that the variables <code>\$myaddress</code>, <code>\$prefix</code> and <code>\$localswitch_server</code> (which use the switch prefix obtained from the switch via ILMI) may not be used if ILMI is not running.</p>

Note – Although SunATM supports PVC connections to a server for ARP traffic, RFC 1577 does not specify this case. For interoperability with other implementations, connections to the server should use SVCs.

Note – In order for two hosts to communicate over PVCs, corresponding PVC connections must also be established in the ATM switch fabric.

Table 5-3 lists the flags and the options that they support.

Table 5-3 /etc/aarconfig File Flag Options

Interface *	Host	ATM Address	VCI	Flags	*
required	illegal	SVC only	illegal	l	local information
required	illegal	required	illegal	L	local information on server
required	illegal	required	illegal	a	access list entry
required	required	or ¹	or ¹	t	permanent table entry
required	illegal	xor ²	xor ²	s	server address/PVC
required	illegal	illegal	illegal	m	manual address registration

¹or – Means one or the other required, but using both is also legal.

²xor – Means one or the other required, but both are illegal.

Note – Entries in the `aarconfig` file must be grouped in a designated order: the local (*l* or *L*) entry must be first, the table (*t*) entries next, and then the server (*s*) entries. Other flags may appear in any order. Also, the ordering need only be maintained among entries for each physical interface; for example, all of the `ba0` entries may appear first, and then all of the `ba1` entries, etc.

5.2.2 Using Variables in the `/etc/aarconfig` File

Because the prefix portion of an ATM address specifies the ATM switch, a number of hosts specified in an `aarconfig` file may have ATM addresses which share the same prefix. To simplify setting up the `aarconfig` file, you can define variables that contain part of an ATM address.

A variable's name is an identifier consisting of a collection of no more than 32 letters, digits, and underscores. The value associated with the variable is denoted by a dollar sign followed immediately by the variable name.

Note – Variables may only be used in the ATM address field. They may not be used in any of the other fields in an entry.

Multiple variables can be concatenated to represent a single ATM address expression. A colon must be used to concatenate the variables. Thus, if one variable, `v1`, is set to `11:22` and another variable, `v2`, is set to `33:44`, the sequence `$(v1):$(v2)` represents `11:22:33:44`. Hexadecimal numbers may also be included with variables in the expression. The expression `45:$(v1):$(v2)` would have the value `45:11:22:33:44`.

Variables are defined in the `aarconfig` file according to the following format:

```
set VARIABLE = EXPRESSION
```

where `VARIABLE` is the name of a variable and `EXPRESSION` is an expression concatenating one or two-digit hexadecimal numbers and/or the values of variables that have been previously defined. The equal sign is optional, but the variable and expression must be separated either by white space (spaces or tabs), an equal sign, or both.

Several predefined variables are built into the SunATM software. These variables are summarized in Table 5-4.

Table 5-4 Predefined SunATM Variables

Variable	Description
<code>prefix</code>	The 13-byte prefix associated with the local switch.
<code>mac</code>	The 6-byte medium access control (MAC) address associated with the local host or interface.
<code>sel</code>	The default 1-byte selector for the local interface.
<code>macsel</code>	The concatenation of <code>\$mac:\$sel</code> .
<code>myaddress</code>	The concatenation of <code>\$prefix:\$mac:\$sel</code> , resulting in the default address for the local interface.
<code>anymac</code>	A wild card representing any 6-byte ESI. Should only be used in <code>a</code> entries.
<code>anymacsel</code>	A wild card representing any 7-byte ESI and Selector combination. Should only be used in <code>a</code> entries.
<code>?</code>	A wild card matching one or two hexadecimal digits within any colon-separated field. For example, <code>\$prefix:\$anymac:?</code> is equivalent to both <code>\$prefix:\$anymac:??</code> and <code>\$prefix:\$anymacsel</code> . However, it is <i>not</i> the same as <code>\$prefix:\$anymacsel:0?</code> , which requires that the first digit of the selector byte is a 0. This wild card should only be used in <code>a</code> entries.
<code>sunmacselN</code>	The concatenation of one of a series of reserved MAC addresses and <code>\$sel</code> to create a block of reserved ATM ARP server addresses. <code>N</code> should be a decimal number in the range 0 - 199.
<code>localswitch_server</code>	The concatenation of <code>\$prefix</code> , a unique reserved MAC address, and <code>\$sel</code> . When used as a server address, restricts server access to clients connected to the local switch only.

Note - The `$prefix` variable, and any other variables which use it (including `$myaddress` and `$localswitch_server`), may not be used on interfaces which are not running ILMI.

In most network configurations, the ATM address assigned to the local interface will be `$myaddress`; using this variable in the `l` entry makes it possible to use identical `aarconfig` files on all Classical IP clients using a given server.

The `sunmacselN` variables can be used in conjunction with a prefix as well as known server addresses which are not bound to a particular system. As an example, consider the case where a server that supports 50 clients fails. If the ATM address of the server is specific to that particular server, the `s` entry must be changed on all 50 clients in order to switch to a backup server. However, if the ATM address used for that server is `$prefix:$sunmacsel3`, this address is not only guaranteed to be unique, since it uses reserved medium access control (MAC) addresses, it is also possible to simply assign that address to the backup server on the same switch by changing the `l` entry to an `s` entry on one system, and bring up a new server with no changes to the clients.

Note – The `sunmacselN` variables do not include a prefix since a client and server may be on different switches and thus have different local prefix values.

In the case of a single-switch network, `localswitch_server` can be used as a well-known server address. Not only does it include the prefix associated with the local switch with a unique MAC address and appropriate Selector, it also restricts server access to clients on the local switch. Thus any host with a network prefix other than that of the local switch will be refused a connection to the ARP server if the ARP server's address is `$localswitch_server`.

Several rules apply to the use of variables in the `aarconfig` file:

1. Two variables cannot follow each other in an expression without an intervening colon. Thus `$v1:$v2` is legal while `$v1$v2` is not.
2. Fields in each line in the `aarconfig` file are separated by white space. Therefore variables should not be separated from the rest of an ATM address with whitespace. For example, `$v1: $v2` is illegal.
3. Once a variable is defined by a set command, it may not be redefined later in the `aarconfig` file.
4. The reserved variable names can not be set. These names include `prefix`, `mac`, `sel`, `macsel`, `myaddress`, `anymac`, `anymacsel`, `sunmacselN` (where `N` is a number between 0 and 199), and `localswitch_server`.

Note – The ESI portion of `localswitch_server` and the `sunmacselN` variables is a reserved MAC address. The hexadecimal values of the reserved addresses are:

```
localswitch_server    08:00:20:75:48:08
sunmacselN base      08:00:20:75:48:10
```

To calculate the ESI portion for a `sunmacselN` address, simply add the value of `N` (converted to a hexadecimal number) to the `sunmacselN` base address. For example, the ESI portion of `sunmacsel20` would be `08:00:20:75:48:10 + 0x14 = 08:00:20:75:48:24`.

5.2.3 Sample Classical IP Configurations

The following examples demonstrate entries in the `/etc/aarconfig` file for several typical network configurations.

Although some of the examples show only one sample `aarconfig` file, similarly configured files must appear on each system. Example 2 shows the files for each of the three systems in the configuration.

1. SVC-only: Clients use the default address and access to the ARP server is restricted to clients on the local switch only.

a. The `/etc/aarconfig` file on a client:

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$myaddress	-	l
ba0	-	\$localswitch_server	-	s

b. The `/etc/aarconfig` file on the server:

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$localswitch_server	-	L

2. PVC-only: *hosta* is connected to *hostb* and *hostc* over PVCs. There is no ARP server.

a. `/etc/aarconfig` on *hosta*:

Interface	Host	ATM Address	VCI	Flag
ba0	-	-	-	l
ba0	hostb	-	100	t
ba0	hostc	-	101	t

b. on *hostb*:

Interface	Host	ATM Address	VCI	Flag
ba0	-	-	-	l
ba0	hosta	-	100	t
ba0	hostc	-	102	t

c. on *hostc*:

Interface	Host	ATM Address	VCI	Flag
ba0	-	-	-	l
ba0	hosta	-	101	t
ba0	hostb	-	102	t

3. SVC-only: *hosta* uses SVCs to connect to *hostb* and *hostc*. All hosts are connected to the same switch; there is no ARP server.

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$myaddress	-	l
ba0	hostb	\$prefix:08:00:20:d5:08:a8:00	-	t
ba0	hostc	\$prefix:08:00:20:21:20:c3:00	-	t

4. PVC/SVC mix: *hosta* uses a SVC to connect to *hostb*, and a PVC to connect to *hostc*. *hostb* is not on the local switch; there is no ARP server.

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$myaddress	-	l
ba0	hostb	45:00:00:00:00:00:00:0f:00:00:00:00::08:00:20:d5:08:a8:00-	-	t
ba0	hostc	-	100	t

5. ARP server: Hosts are connected to an ATM ARP server that resolves addresses. Access is restricted to the local switch subnet and one additional switch subnet.

- a. `/etc/aarconfig` on *hosta*:

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$myaddress	-	l
ba0	-	\$prefix:\$sunmacsel0	-	s

- b. `/etc/aarconfig` on server:

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$prefix:\$sunmacsel0	-	L
ba0	-	\$prefix:\$anymacsel	-	a
ba0	-	45:00:00:00:00:00:00:0f:00:00:00:00:\$anymacsel	-	a

6. Manual address configuration: Hosts are connected to a switch that does not support ILMI.

- a. `/etc/aarconfig` on server:

```
set prfx = 45:00:00:00:00:00:00:0f:00:00:00:00
```

Interface	Host	ATM Address	VCI	Flag
ba0	-	\$prfx:\$sunmacsel	-	L

b. `/etc/aarconfig` on server:

```
set prfx = 45:00:00:00:00:00:00:00:0f:00:00:00:00
Interface Host      ATM Address          VCI   Flag
ba0       -                $prfx:$macsel       -     l
ba0       -                $prfx:$sunmacsel0   -     s
```

5.3 Configuring a LAN Emulation Interface

LAN Emulation, standardized by the ATM Forum’s LAN Emulation 1.0 specification, is another way of providing TCP/IP and UDP/IP support over an ATM interface. Address resolution information is provided by a series of LAN Emulation services. When a LAN Emulation interface is brought up, it must “join the LAN,” that is, it must register with these services. This process, and the address resolution process is described in Section 6.3, “LAN Emulation.”

Unlike Classical IP, the LAN Emulation protocol provides a broadcast service to the upper layer protocols. Therefore, the multicast and RIP limitations described in Section 5.2, “Configuring a Classical Internet Protocol Interface,” do not affect LAN Emulation interfaces.

5.3.1 Editing the `/etc/laneconfig` File

The `/etc/laneconfig` file contains the required configuration information for each interface that uses LAN Emulation. One entry is required for each SunATM interface. If you choose the automatic configuration option during the software installation, this entry will be automatically created by the installation program.

Each time you modify the `/etc/laneconfig` file, you must run the LAN Emulation setup program (`lanesetup`). `lanesetup` is in the `/etc/opt/SUNWatm/bin` directory.

The entry provides the ATM and MAC addresses which will be used by the LAN Emulation software to identify the local interface.

Each `/etc/laneconfig` entry follows this format:

Interface	MAC Address/ ELAN Name	ATM Address	VCI	Flag
-----------	---------------------------	-------------	-----	------

These entry fields are described in Table 5-5.

Table 5-5 `/etc/laneconfig` Entry Descriptions

Field	Description
Interface	Refers to the LAN Emulation interface, <code>laneN</code> .
MAC Address/ELAN Name	This field is for the 6-byte MAC address of the interface, or, in the case of an <code>n</code> entry, the name of the emulated LAN to join.
ATM Address	This field is for the 20-byte ATM address. The <code>\$myaddress</code> variable assigns the local switch prefix, local MAC address, and default selector.
VCI	The VCI field is a positive decimal integer identifying a Permanent Virtual Circuit. Place a dash in this field if VCI is not used.
Flag	This field identifies whether this entry is a local address (<code>l</code>), a permanent table entry (<code>t</code>), or an LECS address entry (<code>c</code>). For a complete description of the <code>laneconfig</code> flags, see Table 5-6.

Table 5-6 /etc/laneconfig Flag Descriptions

Flag	Description
l	This flag designates a local address entry. There must be an <i>l</i> entry for each interface running LAN Emulation. The interface and the ATM address must be included.
t	This flag designates a table entry for the local MAC-ATM address resolution table. If you wish to avoid the address resolution process for a frequently accessed system, for instance, you may include a <i>t</i> entry for that system; you may also create PVCs with a <i>t</i> entry. The interface, MAC address, and either ATM address or VCI (for SVC or PVC connection, respectively) must be included.
n	This flag allows you to specify the name of an emulated LAN to join. By default, the SunATM implementation will use the name provided by the LECS. If you wish to specify a different name, or if your LECS requires that a user include a name in its requests, a name may be provided with this flag. Interface is required; the name should be entered in the second field.
a	Represents an address that may have access to this host. If no <i>a</i> entries appear in the <code>laneconfig</code> file, access to the host is unrestricted. Including <i>a</i> entries allows access to be restricted to known hosts. As an alternative to listing individual addresses, the ATM address field may contain a prefix, followed by the wildcard <code>\$anymacsel</code> , which matches any 7-byte ESI/Selector combination following the given prefix. This allows access by any host connected to the switch specified by the given prefix. <i>Mac Address</i> and <i>VCI</i> should not appear; <i>ATM Address</i> is required. See Table 5-7.

Table 5-6 /etc/laneconfig Flag Descriptions (Continued)

Flag	Description
c	This flag allows an alternate LECS address to be specified. By default, the SunATM software uses the well-known address specified in the LAN Emulation standard. If, however, your LECS has a different address, or you wish to connect to the LECS over a PVC, you may provide the alternate ATM address or VCI in a c entry. If you wish to make a PVC connection, the VCI must be 17, as required by the LAN Emulation standard. The interface and ATM address or VCI must be included.
s	This flag specifies the LES address or VCI, and instructs the system to contact the LES directly, and to use default subnet configuration information. This flag should be used if your subnet does not have an LECS. Without this entry, the system first connects to the LECS, which provides the LES address and configuration information.
m	Notifies the system that the entire ATM address, including the network prefix, must be configured manually on this interface. If your interface is connected to a switch that does not support ILMI, you must include this option in your /etc/aarconfig or /etc/laneconfig file. Note that the variables \$myaddress, \$prefix and \$localswitch_server (which use the switch prefix obtained from the switch via ILMI) may not be used if ILMI is not running.

Table 5-7 describes the required, optional, and illegal fields for each flag type.

Table 5-7 laneconfig Flag Requirements and Options

Interface	MAC Address/ELAN Name	ATM Address	VCI	Flag
required	illegal	required	illegal	l
required	required	xor ¹	xor ¹	t
required	Emulated LAN name	illegal	illegal	n
required	illegal	required	illegal	a
required	illegal	xor ¹	xor	c
required	illegal	xor ¹	xor	s
required	illegal	illegal	illegal	m

¹xor means that you can use either the ATM Address field or the VCI field, but not both.

Note – Designate unused fields in the /etc/laneconfig file with a dash.

5.3.2 Using Variables in the /etc/laneconfig File

Some of the predefined variables used in the /etc/aarconfig file, may also be used in /etc/laneconfig. The applicable variables are listed in Table 5-4. For a complete description of how to use these variables, see Section 5.2.2, “Using Variables in the /etc/aarconfig File.”

Table 5-8 Predefined SunATM Variables

Variable	Description
prefix	The 13-byte prefix associated with the local switch.
mac	The 6-byte MAC address associated with the local host or interface.
sel	The default 1-byte selector for the local interface.
macsel	The concatenation of \$mac:\$sel.
myaddress	The concatenation of \$prefix:\$mac:\$sel, resulting in the default address for the local interface.
anymac	A wild card representing any 6-byte ESI. Should only be used in a entries.
anymacsel	A wild card representing any 7-byte ESI and Selector combination. Should only be used in a entries.
?	A wild card matching one or two hexadecimal digits within any colon-separated field. For example, \$prefix:\$anymac:? is equivalent to both \$prefix:\$anymac:?? and \$prefix:\$anymacsel. However, it is <i>not</i> the same as \$prefix:\$anymacsel:0?, which requires that the first digit of the selector byte is a 0. This wild card should only be used in a entries.

Note - The \$prefix variable, and any other variables which use it (including \$myaddress), may not be used on interfaces which are not running ILMI.

5.3.3 Sample LAN Emulation Configurations

The following examples demonstrate entries in the `/etc/laneconfig` file for several typical configurations.

Although some of the examples show only one sample `laneconfig` file, similarly configured files must appear on each LAN Emulation client.

1. Basic LAN Emulation client. The ATM and MAC address of a frequently used server is provided. The LECS provides the name of the Emulated LAN.

```
set srvr_mac = 08:00:20:01:02:03
```

Interface	MAC_Address/ELAN Name	ATM_Address	VCI	Flag
lane0	-	\$myaddress	-	l
lane0	-	\$prefix:\$srvr_mac	-	t

2. LAN Emulation client. The LECS requires that the client send the Emulated LAN name in its messages.

Interface	MAC_Address/ELAN Name	ATM_Address	VCI	Flag
lane0	-	\$myaddress	-	l
lane0	elan1	-	-	n

5.4 Supporting Logical Interfaces

A new feature in the SunATM 2.1 software is the support of logical interfaces in the LAN Emulation environment. Logical interfaces allow you to assign multiple IP addresses to a single Emulated LAN interface. A logical interface name consists of three parts: the device name (in the case of SunATM LAN Emulation, `lane`); the major number, which corresponds to the lane instance number; and the minor number, which distinguishes the logical interfaces on a single physical interface. The format of a LAN Emulation logical interface name is `laneN:X`, where `N` is the major number and `X` is the minor number.

Each logical interface will be associated with a unique IP hostname and address. All logical interfaces on a given physical interface will be associated with the same ATM and MAC addresses. Logical interfaces should be configured by placing multiple entries for a given interface in the `/etc/atmconfig` file.

The following rules and notes should be considered when using logical interfaces with the SunATM 2.1 software:

- Only one signalling protocol (for example, UNI 3.0 or 3.1) is supported per interface, and must appear in the first entry for that interface.
- Only one Classical IP hostname may be assigned to an interface; it may appear in any entry in any order in `/etc/atmconfig`.
- The first `laneN` entry on an interface must be for `laneN:0`, or simply `laneN`. `laneN` and `laneN:0` are identical and interchangeable.
- IP limits the number of logical interfaces on a physical interface to 256 (the minor number `X` must be in the range 0 - 255).

The following example shows the `atmconfig` and `laneconfig` files and the `ifconfig -a` output for a system with one physical interface, `ba0`. That interface runs both Classical IP and LAN Emulation under UNI 3.1, and has 3 different IP addresses. The IP hostnames, `cip0`, `atm0`, `atm1`, and `atm2`, should be configured appropriately in `/etc/hosts`.

The example `/etc/atmconfig` file:

Interface	UNI	CIP Hostname	LANE Instance	LANE Hostname
<code>ba0</code>	3.1	<code>cip0</code>	0	<code>atm0</code>
<code>ba0</code>	-	-	0:1	<code>atm1</code>
<code>ba0</code>	-	-	0:2	<code>atm2</code>

The corresponding example `/etc/laneconfig` file:

Interface	MAC Address/ ELAN Name	ATM Address	VC	Flag
lane0	-	\$myaddress	-	1

The resulting `ifconfig -a` output:

```
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
    inet 127.0.0.1 netmask ff000000
ba0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 9180
    inet 192.29.235.36 netmask ffffffff broadcast 192.29.235.255
    ether 8:0:20:7a:37:af
lane0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.240.36 netmask ffffffff broadcast 192.29.240.255
    ether 8:0:20:7a:37:af
lane0:1: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.241.36 netmask ffffffff broadcast 192.29.241.255
lane0:2: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.242.36 netmask ffffffff broadcast 192.29.242.255
```

5.5 Supporting Multiple Emulated LANS on a Single Interface

The SunATM 2.1 software allows a single ATM interface to join up to sixteen emulated local area networks (ELANs), provided that this is allowed by the switch and LAN Emulation (LANE) services. Each ELAN joined will be represented by a unique lane instance (e.g. lane0 or lane1).

Note – A requirement for supporting this feature is that the adapter card be assigned multiple MAC addresses, which is supported in the SunATM-622 SBus adapters and the new SunATM-155 SBus adapters 2.1. This feature *WILL NOT* work with the older SunATM SBus Adapters 2.0. You can find the number of MAC addresses assigned to your SunATM adapter by using the `atmgetmac(1M)` command with the `count` option.

The joining of multiple ELANs is configured by placing multiple entries in the `/etc/atmconfig` and `/etc/laneconfig` files. Each lane instance will have a unique IP hostname and address, ATM address, and MAC address associated with it. In addition, an ELAN name should be assigned to the instance if any

ELAN other than the default is to be joined. This information, with the exception of the MAC address, which is retrieved from the board itself, should be provided in the `/etc/atmconfig` and `/etc/laneconfig` configuration files.

Note – Only one signalling protocol (e.g. UNI 3.0 or 3.1) and one Classical IP instance are supported per physical interface. The UNI version must be specified in the first `/etc/atmconfig` entry for a given interface; the Classical IP instance may be specified in any entry.

The following example shows the `/etc/atmconfig` and `/etc/laneconfig` files and the `ifconfig -a` output for a system with one SunATM interface, `ba0`. The interface is using UNI 3.0 for signalling, and is not running Classical IP. It will join 4 emulated LANs: the default, `elan1`, `elan2`, and `elan3`.

The example `/etc/atmconfig` file:

Interface	UNI	CIP	Hostname	LANE	Instance	LANE	Hostname
ba0	3.0	-		0		atm0	
ba0	-	-		1		atm1	
ba0	-	-		2		atm2	
ba0	-	-		3		atm3	

The corresponding example `/etc/laneconfig` file:

Interface	MAC Address/ ELAN Name	ATM Address	VC	Flag
lane0	-	\$myaddress	-	l
lane1	-	\$myaddress	-	l
lane1	elan1	-	-	n
lane2	-	\$myaddress	-	l
lane2	elan2	-	-	n
lane3	-	\$myaddress	-	l
lane3	elan3	-	-	n

The resulting `ifconfig -a` output:

```
lo0: flags=849<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
    inet 127.0.0.1 netmask ff000000
lane0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.240.36 netmask ffffffff broadcast 192.29.240.255
    ether 8:0:20:7a:37:af
lane1: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.241.36 netmask ffffffff broadcast 192.29.241.255
    ether 8:0:20:7a:37:b0
lane2: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.242.36 netmask ffffffff broadcast 192.29.242.255
    ether 8:0:20:7a:37:b1
lane3: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING,MULTICAST> mtu 1500
    inet 192.29.243.36 netmask ffffffff broadcast 192.29.243.255
    ether 8:0:20:7a:37:b2
```

5.6 Tuning Your System for Better SunATM Performance

There are some system parameters that can greatly affect the speed and performance of your SunATM software and hardware. By adjusting these system parameters and options, you can achieve better performance out of your SunATM interfaces. For an up-to-date list of recommendations, visit the SunATM home page (<http://www.sun.com/ATM>).

Classical IP and LAN Emulation Protocols



ATM is a connection-oriented network protocol, which means that a connection must be established between two communicating entities before data transfer can begin. IP is inherently connectionless. The implementation on the host must therefore reconcile the differences in these two paradigms.

There are two standardized, commonly used ways of doing this: Classical IP, standardized in RFC 1577, and LAN Emulation standardized in the LAN Emulation 1.0 specification from the ATM Forum. The SunATM architecture supports both of these methods. Some of the key ideas of these two methods are discussed in later sections of this chapter.

Both methods allow IP to run transparently over the ATM interface. Thus IP itself sees the ATM interface just as it sees any traditional network interface. Every SunATM interface has a subnet IP address. During the process of startup of an ATM interface, appropriate modules and drivers are plumbed. All the TCP/IP and UDP/IP applications run without modifications over these modules, and all the utilities associated with the network interfaces also run without modification and display similar results (for example, `netstat`, `ifconfig` utilities, etc.), with one exception. Because of the different plumbing of the ATM modules, the `plumb` and `unplumb` options of `ifconfig` will not work on ATM interfaces; the `atmplumb(1M)` command must be used instead. IP treats the ATM interface as a subnet, choosing the interface used to send a packet out based on the IP address of the destination and on the IP address and netmask of the interface itself.

The transparency to IP is enabled in different ways in Classical IP and LAN Emulation. Those differences will be discussed in later sections of this chapter.

SunATM signalling conforms to the UNI specification of the ATM Forum. Both versions 3.0 and 3.1 of that specification are supported. This signalling, called Q.2931, runs on top of QSAAL and uses VC 5 for signalling as specified in the Forum specification.

6.1 ATM Addresses and Address Registration

UNI signalling uses ATM addresses for signalling. Every ATM interface will have an ATM address in addition to its IP address.

ATM addresses, like NSAP addresses, are 20 octets long. The End System Identifier (ESI) field within the ATM address is a unique 6 octet value; this can be the IEEE hardware MAC address conventionally associated with every network interface. The Selector field is one octet long. The 13 octets that make up the rest of the ATM address are called the Network Prefix, and should be derived from the ATM switch fabric to which the interface is connected. Every ATM switch fabric is configured with a 13 octet prefix.

On a SunATM host, the prefix associated with the local switch fabric is represented by the variable prefix. Its value will be obtained by the system at configuration time.

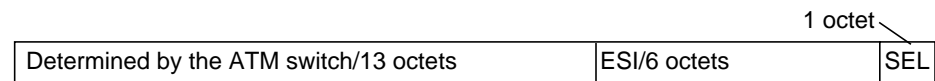


Figure 6-1 ATM Address Fields

The UNI specification specifies the Interim Local Management Interface (ILMI) service interface for a client to learn and register its ATM address. The ILMI service interface is based on the use of SNMP over AAL5. In the SunATM software package, ILMI service is provided by an address registration daemon, `ilmid`.

6.1.1 ATM Address Registration Daemon (ILMID)

Address registration with a switch is controlled by `ilmid`. When an ATM interface is brought up at boot time, `ilmid` is also started. `ilmid` then begins an exchange of messages with the switch: relaying local address information (the 7 octet ESI and selector) to the switch, and receiving the 13 octet network prefix information from the switch.

The default local address that is registered with the switch at boot time consists of the network prefix provided by the switch, the MAC address assigned to the local interface, and the default selector for that interface (usually 0). Additional addresses may be registered in two different ways. `aarsetup(1M)` and `lanesetup(1M)` register additional local addresses that may appear in `aarconfig(4)` and `laneconfig(4)`, respectively. There is also a user program, `atmreg(1M)`, that may be used to register addresses, to de-register addresses, and to check the status of any address.

6.2 *Classical Internet Protocol*

The major task required for ATM to work transparently under IP is resolving an IP address to an ATM address and establishing the connection to that destination. Classical IP does this via a database of IP/ATM address pairs that is either provided by an ATM ARP server which is accessible to all hosts on the subnet, or maintained locally in each host.

6.2.1 *ATM Address Resolution*

Traditional TCP/IP and UDP/IP applications use IP addresses for communicating to a destination. In order for these applications to run as before, there is a need to resolve these IP addresses into ATM addresses. The ATM address is then used in signalling to establish an ATM connection to the destination. An ATM connection in turn is represented by a VPI/VCI. The host must use this returned VPI/VCI to send packets to the destination representing the ATM connection.

ATM address resolution, also called ATM ARP, follows RFC 1577, the classic draft that describes the ATM ARP process.

RFC 1577 is based upon the existence of an ATM ARP server on every subnet. Every client of the subnet communicates with the ATM ARP server to derive an ATM address of the destination from the IP address of the destination. The ATM ARP server holds the IP to ATM address information for all hosts in the ATM subnet. It is likely that initial ATM configurations will not rely on dynamic ATM address resolution since it requires the presence of an ATM ARP server on every subnet. Also, there are no specified standards for providing redundant ATM ARP servers for a subnet. As specified, the ATM ARP server would constitute a single point of failure in the system. From a practical standpoint, however, early configurations may take the course of having the IP

to ATM address database in every system, thus avoiding the IP to ATM address resolution step altogether. The RFC requires the use of a router to pass data between subnets.

SunATM software facilitates this by providing ATM utilities that will allow configurations to specify IP to ATM addresses in `/etc/aarconfig` files. The `aarsetup` program uses the information in `/etc/aarconfig` to create IP to ATM address resolution tables. Dynamic entries into a server's resolution table are also supported.

Table 5-1 on page 5-2 shows the format of the `/etc/aarconfig` file for specifying the IP to ATM address. It is important for the file to be consistent on all systems in the subnet. See Section 5.2.1, "Editing the `/etc/aarconfig` File," on page 5-4.

6.2.2 ATM ARP Address Resolution Tables

Depending on the `aarconfig` file, the Classical IP software will run as either a server or a client. As a server, the Classical IP software is responsible for handling ATM ARP requests originating from its clients. An ATM server has to be configured for each subnet. The ATM ARP server code conforms to RFC 1577: clients send ATM ARP requests to the server to resolve a destination IP address to an ATM address. The server then replies to ATM ARP requests by sending an ATM ARP response. If the server does not have the IP to ATM address entry, then it replies with NAK.

The file `/etc/aarconfig` is also used by the ATM ARP server. All the IP to ATM address entries specified in the file will be entered into a kernel resident table by the ATM ARP setup program, `aarsetup`. Additional entries in the kernel table will be added dynamically using the inverse ARP process. When a client connects to the server, the server will send an inverse ARP request back to the client to obtain the client's IP address. When a response is received, an entry will be created for that client. The Classical IP software will also respond to client ARP requests. The software looks up a kernel IP to ATM address entry and responds to an ATM ARP request with either an ATM ARP reply or ATM ARP NAK (if there is no entry in the table). Note that an ATM ARP client uses the virtual channel (VC) specified in the `/etc/aarconfig` file to communicate with the server; or, if an ATM address is specified, it establishes an switched virtual circuit (SVC) connection to communicate with the server.

While dynamic entries in the ARP server's table make network administration less complex, it also creates a security problem. Any host may register with the ARP server and, therefore, gains access to the subnet. To resolve this issue, a list of hosts or networks may optionally be provided with a entries in the server's `/etc/aarconfig` file. If no `a` entries appear, any host will be allowed to connect to the server. If any `a` entries exist, only those hosts whose addresses match those specified will be allowed to connect.

Although the `a` entry requires a complete ATM address, multiple addresses can be referenced in a single entry using the provided wildcards. See Section 5.2.2, "Using Variables in the `/etc/aarconfig` File," on page 5-8, for more information about this feature.

The advantage to having an ATM ARP server in the subnet is that it represents a known source for all address resolutions. It is the only host which a client must know about to have IP addresses resolved to ATM connections, and it allows for access control in the ATM network.

When the `/etc/aarconfig` file has been modified on a system, it is necessary to rerun `aarsetup`.

Note – For better caching, all clients have the option of adding to their configuration file the IP to ATM address information for other clients. This can benefit clients who communicate frequently because it eliminates having to go through the ATM ARP server for IP to ATM address resolution.

If a host has multiple SunATM cards, the host may be a server for one IP subnet and a client for another. This is handled transparently by `aarsetup`.

6.3 LAN Emulation

As described in previous sections, Classical IP provides its own (IP to ATM) address resolution mechanism which corresponds to and replaces ARP, thus allowing IP-based applications to run transparently over ATM. A shortcoming of Classical IP, and a primary reason it must replace the traditional ARP, is that it does not support broadcast messages.

Because ATM is a connection-oriented protocol (unlike ethernet), implementing broadcast is much more difficult. The only host that receives a message is the host to which the message is addressed, and a call must be established to that host before the message can be sent.

Local Area Network (LAN) Emulation, as standardized by the ATM Forum, provides mechanisms to send broadcast messages in an ATM environment. Given this capability, LAN Emulation is also able to work transparently with ARP, as well as IP. IP and ARP may send broadcast messages over the ATM interface, and thus resolve IP addresses to MAC addresses; messages are then sent to the LAN Emulation driver, which has its own address resolution protocol (similar to that of Classical IP) to resolve the medium access control (MAC) address to an ATM address and connection.

The SunATM 2.1 software implements the client side of the LAN Emulation standard. In order to use LAN Emulation in an environment, several LAN Emulation services must also exist in the emulated LAN. These services, called the LAN Emulation Configuration Server (LECS), the LAN Emulation Server (LES), and the Broadcast and Unknown Address Server (BUS), are generally provided in an ATM switch. An overview of the functions of these servers is provided in the following sections.

6.3.1 LAN Emulation Services

6.3.1.1 LAN Emulation Configuration Server

This server is contacted first by a host interface when the host is brought up on the emulated LAN. Its address is generally a well-known address specified by the LAN Emulation standard which is coded into the host software; thus no input from the user is required to establish this connection. When contacted by a host wishing to join its emulated LAN, the LECS replies with configuration parameters for the emulated LAN, as well as the address of the LES.

6.3.1.2 LAN Emulation Server

The second step in joining an emulated LAN is to make a connection to the LAN Emulation Server. After receiving the LES address from the LECS, a host will establish a connection to the LES. The LES may add the host to a point-to-multipoint call which is maintained by the LES with connections to every host in the emulated LAN. This point-to-multipoint connection, if created by the LES, is used to send control information to each host on the emulated LAN.

The LES acts as the ATM ARP server. Since IP and ARP work with MAC addresses, an additional address resolution step is required to convert a MAC address to the corresponding ATM address, which is used to make a connection to the target host; this resolution step is provided by the LES.

6.3.1.3 Broadcast and Unknown Address Server

The final step in joining an emulated LAN is to make a connection to the BUS. The ATM address of the BUS is obtained by sending a LAN Emulation ARP request to the LES for the broadcast address. Once established, this connection is used to send broadcast messages to the BUS, which will add the client to a point-to-multipoint call including all hosts on the emulated LAN. Thus when a broadcast message (such as an IP ARP request) is received by the LAN Emulation host from its upper layers, it sends that message to the BUS, which forwards it to all hosts in the emulated LAN. Just as in the case of ethernet, the correct host responds to the sender, and thus the IP address is resolved to a MAC address.

6.3.2 Resolving an IP Address to an ATM Connection

The entire process from the time IP sends a message addressed to an IP address to the arrival of that message at the appropriate destination was hinted at in the above descriptions of the LAN Emulation servers. To demonstrate how those pieces work together during the actual transmission of a message, the process is described below, assuming that none of the needed addresses have been previously resolved and cached. The two hosts involved are referred to as the source (the system who wishes to send a message) and the target (the system to which the message is addressed).

1. IP has a message to transmit, and only knows the IP address of the target system. It first sends a message to ARP, to resolve the IP address to a MAC address.
2. ARP creates a broadcast request for the MAC address corresponding to the given IP address, which it sends to the LAN Emulation driver.
3. The LAN Emulation driver recognizes that this message has a broadcast address, and sends it to the BUS, which forwards the message to every host on the emulated LAN.

4. The message is received on each host, and sent up to ARP by the LAN Emulation driver.
5. On the target, ARP recognizes the IP address as its own and sends a response with its MAC address (addressed to the source's MAC address) down to the LAN Emulation driver.
6. The LAN Emulation driver sends an LE ARP request to the LES to resolve the source's MAC address to its ATM address.
7. The LES responds with the requested ATM address, and the target host sets up an ATM connection to the source host, over which it sends the IP ARP response.
8. The LAN Emulation driver on the source receives the IP ARP response message and sends it up to ARP. ARP then inserts the MAC address into the original message and sends it back down to the LAN Emulation driver.
9. The LAN Emulation driver then must send an LE ARP request to the LES to resolve the MAC address in the message from ARP to an ATM address. When it receives an LE ARP response, it then sees that it has a connection to that address (established by the target to return the IP ARP response), and sends the original IP message to the target over that connection.

6.3.3 LAN Emulation Connections

As should be somewhat obvious from the preceding discussion, there will be several connections established at all times when a host is a member of an emulated LAN. The following table outlines the various LAN Emulation-related connections that should be expected on a LAN Emulation client (LEC).

Note – The command `qccstat(1M)` may be used to view all existing connections for a given interface.

Table 6-1 LAN Emulation Connections

Connection	Comments
LEC → LECS	This connection is not required to remain open after the initial join of the emulated LAN, and thus may time out after a host has joined the LAN.
LEC → LES	Point-to-point connection over which the host may send LE ARP requests and receive responses from the LES.
LES → LEC	Point-to-multipoint connection over which the LES may send administrative information to all hosts. Hosts may not send on this connection.
LEC → BUS	Point-to-point connection over which the host may send broadcast messages to the BUS. A limited amount of data is also allowed on this connection.
BUS → LEC	Point-to-multipoint connection over which the BUS sends broadcast messages. Hosts may not send on this connection.

6.4 ATM and SNMP

Two of the ATM standards supported by the SunATM software, the User Network Interface (UNI) and LAN Emulation (LANE) specifications, include definitions of SNMP-style Management Information Bases (MIBs) relevant to those standards. These MIBs are referred to as the ATM Forum (ATMF) and LAN Emulation (LANE) MIBs, respectively.

The ATM SNMP daemon (`atmsnmpd`) handles requests for information in both MIBs, as well as the system MIBs, from SNMP-based network management systems (such as the SunNet Manager program), and from `ilmid`, when it is required, for SNMP requests coming from the switch.

If you configured your system (using the `atmadmin` configuration program) to start `atmsnmpd` as the default SNMP agent in the system then `atmsnmpd` will listen to UDP port 161 for SNMP traffic. Otherwise, `atmsnmpd` will be started with the `-n` option, meaning that `atmsnmpd` will not listen on any UDP port for SNMP traffic, but it can still be used by `ilmid`.

There is one caveat associated with the use of `atmsnmpd` running as an agent on a system. Since the SNMP protocol is defined to use a single UDP port number, only one SNMP agent, such as `atmsnmpd`, may run on a system at a time. Most SNMP agent daemons, including `atmsnmpd`, allow an alternate port number to be specified, but this will limit the accessibility of that agent to network managers such as the SunNet Manager program. Depending on your requirements, you may wish to run `atmsnmpd` on an alternate port. The `atmsnmpd` daemon is started in the `/etc/rc2.d/S00sunatm` script; the `-p` option with an alternate port number may be added to this script.

A new feature of the SunATM 2.1 software is the use of `atmsnmpd` as a forwarding agent. If configured as a forwarding agent, `atmsnmpd` will forward SNMP requests for unknown MIBs to the port specified with the `forward` option, `-f`. This allows a system to have two SNMP agents respond to requests received over the SNMP port. Figure 6-2 illustrates the required configuration. In order to set up this example configuration, `atmsnmpd` must be started with the parameter `-f 1000` and `other_snmpd` must be started so that it listens on port 1000.

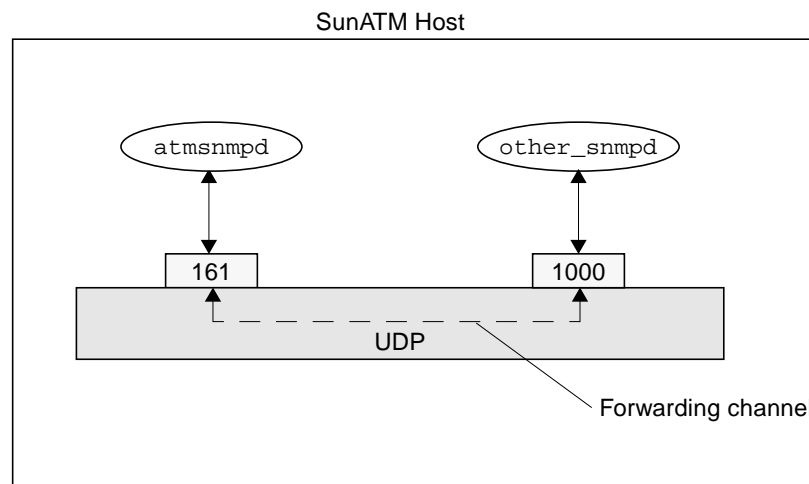


Figure 6-2 Using atmsnmpd as a Forwarding Agent

Note – If no port is specified to forward unknown requests, `atmsnmpd` will respond with a “No Such Name” error to requests for MIBs which it does not support. If a forwarding port is specified, `atmsnmpd` will instead forward the request to the specified port. Responses received from the agent running on the forwarding port will be sent to the requesting SNMP manager with no modification. If the agent does not respond, then `atmsnmpd` will not send any response back.

Appendix D, “Managing SunATM Interfaces with SNMP,” provides more information about using `atmsnmpd` to manage and monitor the SunATM interfaces with a network manager such as the SunNet Manager program.

Wiring Scheme and Pin Descriptions

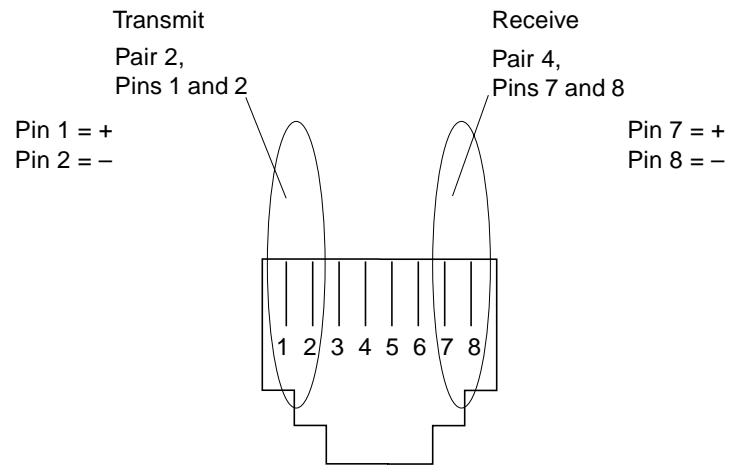


Figure A-1 Designation T568B

Table A-1 Pin Descriptions for the 96-Pin SBus Connector

Pin	Description	Pin	Description	Pin	Description	Pin	Description
1	GND	2	BR\	49	Clk	50	BG\
3	Sel\	4	IntReq1\	51	AS\	52	GND
5	D00	6	D02	53	D01	54	D03
7	D04	8	IntReq2\	55	D05	56	+5V
9	D06	10	D08	57	D07	58	D09
11	D10	12	IntReq3\	59	D11	60	GND
13	D12	14	D14	61	D13	62	D15
15	D16	16	IntReq4\	63	D17	64	+5V
17	D19	18	D21	65	D18	66	D20
19	D23	20	IntReq5\	67	D22	68	GND
21	D25	22	D27	69	D24	70	D26
23	D29	24	IntReq6\	71	D28	72	+5V
25	D31	26	Siz0	73	D30	74	Siz1
27	Siz2	28	IntReq7\	75	Rd	76	GND
29	PA00	30	PA02	77	PA01	78	PA03
31	PA04	32	LErr\	79	PA05	80	+5V
33	PA06	34	PA08	81	PA07	82	PA09
35	PA10	36	Ack0\	83	PA11	84	GND
37	PA12	38	PA14	85	PA13	86	PA15
39	PA16	40	Ack1\	87	PA17	88	+5V
41	PA18	42	PA20	89	PA19	90	PA21
43	PA22	44	Ack2\	91	PA23	92	GND
45	PA24	46	PA26	93	PA25	94	PA27
47	DtaPar	48	-12V	95	Reset\	96	+12V

SunATM SBus Adapters Specifications



B.1 SunATM-155 SBus Adapters 2.1 Specifications

B.1.1 Performance Specifications

Table B-1 Performance Specifications

Feature	Specification
SBus Clock	25 MHz max., 12.5 MHz min.
Max SBus Burst Transfer Rate	34 Mbytes/sec (approximately)
Steady State SBus Transfer Rate	5 Mbytes/sec
SBus Data/Address Lines	D (31:0)/PA (27:0)
SBus Modes	Master/Slave
Capacitance per SBus Signal Line	≤20 pF
SBus Parity	Yes
SBus Version	IEEE 1496
SBus Burst Sizes	16/32/64

B.1.2 Power Specifications

Table B-2 Power Specifications

Specification	Measurement
Power Dissipation	9.5 Watt max.
Voltage Tolerance	+/- 5%
Ripple	≤ 100 mV
Operational Current	5V, 2.0 Amps

B.1.3 Physical Dimensions

Table B-3 Physical Dimensions

Dimension	Measurement
Length	5.78 in. (146.70 mm)
Width	3.3 in. (83.82 mm)

B.1.4 Environmental Specifications

Table B-4 Environmental Specifications

Condition	Operating Specification	Storage Specification
Temperature	0 to 70°C (+32 to +131°F)	-25 to 70°C (-25 to +131°F)
Relative Humidity	5 to 85% non-condensing (40°C, wet bulb temperature)	0 to 95% non-condensing 40°C /hour
Altitude	-1000 to +15,000 ft.	-1000 to +50,000 ft.
Shock	5g, 1/2 sine wave, 11 msec	30g, 1/2 sine wave, 11 msec
Vibration, pk to pk displacement	0.005 in. max. (5 to 32 Hz)	0.1 in. max (5 to 17 Hz)
Vibration, peak acceleration	0.25g (5 to 500 Hz) (Sweep Rate = 1 octave/min.)	1.0g (5 to 500 Hz) (Sweep Rate = 1 octave/min.)

B.2 SunATM-622/MMF SBus Adapter 2.1 Specifications

B.2.1 Performance Specifications

Table B-5 Performance Specifications

Feature	Specification
SBus Clock	25 MHz max., 12.5 MHz min.
Max SBus Burst Transfer Rate	100 Mbytes/sec (approximately)
Steady State SBus Transfer Rate	100 Mbytes/sec
SBus Data/Address Lines	D (31:0)/PA (27:0)
SBus Modes	Master/Slave
Capacitance per SBus Signal Line	≤20 pF
SBus Parity	Yes
SBus Version	IEEE 1496
SBus Burst Sizes	16/32/64

B.2.2 Power Specifications

Table B-6 Power Specifications

Specification	Measurement
Power Dissipation	13 Watt max.
Voltage Tolerance	+/- 5%
Ripple	≤ 100 mV
Operational Current	5V, 2.0 Amps

B.2.3 Physical Dimensions

Table B-7 Physical Dimensions

Dimension	Measurement
Length	5.78 in. (146.70 mm)
Width	6.6 in. (167.64 mm)

B.2.4 Environmental Specifications

Table B-8 Environmental Specifications

Condition	Operating Specification	Storage Specification
Temperature	0 to 70°C (+32 to +131°F)	-25 to 70°C (-25 to +131°F)
Relative Humidity	5 to 85% non-condensing (40°C, wet bulb temperature)	0 to 95% non-condensing 40°C /hour
Altitude	-1000 to +15,000 ft.	-1000 to +50,000 ft.
Shock	5g, 1/2 sine wave, 11 msec	30g, 1/2 sine wave, 11 msec
Vibration, pk to pk displacement	0.005 in. max. (5 to 32 Hz)	0.1 in. max (5 to 17 Hz)
Vibration, peak acceleration	0.25g (5 to 500 Hz) (Sweep Rate = 1 octave/min.)	1.0g (5 to 500 Hz) (Sweep Rate = 1 octave/min.)

C.1 Troubleshooting While Starting a SunATM Interface

There are many steps involved in making an interface active on an ATM network. Once you have configured the interface properly, these steps should be transparent to you. However, problems in your configuration may cause a failure at any number of points along the way. The following list contains checks you can make to determine where in the process your system failed, and what to do to remedy the situation. If you continue to experience problems, information gathered from these checks will help your service provider diagnose the problem.

This section is divided into three sub-sections:

Section	Description
Section C.1.1, "Generic Configuration"	Refers to all ATM configurations, regardless of the type of IP support involved, if any.
Section C.1.2, "Classical IP Configuration"	Refers only to interfaces configured to support Classical IP.
Section C.1.3, "LAN Emulation Configuration"	Refers to interfaces configured to support LAN Emulation.

C.1.1 Generic Configuration

- Make sure that there is an entry for the interface in `/etc/atmconfig`.

Configuration of an interface begins during system boot. Configuration will be attempted for all interfaces listed in `/etc/atmconfig`. For information about the format of this file, see Section 5.1, “Editing the `/etc/atmconfig` File,” on page 5-2, and the `atmconfig(4)` man page.

- Check to see if any error messages were printed during the boot process.

If there were error messages, see Section C.2, “Error Messages.”

- Verify linkstate in `qccstat(1M)`.

This command will indicate the signalling status of your interface. The linkstate should be `DL_ACTIVE`. If it is not, your interface is not communicating properly with your switch.

1. Make sure that your switch and interface are both configured to run the same version of UNI signalling.

The SunATM software supports UNI versions 3.0 and 3.1; the version is set per-interface in the `/etc/atmconfig` file.

2. Verify that your interface is physically connected to the switch, and that the switch sees the physical connection (most switches have a physical link LED for each port).

If your interface is a multimode fiber interface, one possible cause for a bad physical connection is that transmit and receive are swapped. “transmit” on your interface should be connected to “receive” on the switch, and “receive” on your interface to “transmit” on the switch. There is generally writing on one of the cables in a transmit-receive pair so that the two cables are distinct.

- Verify that an address has been registered with the switch.

The `qccstat(1M)` command also lists all addresses registered to the interface with the switch. See Section 6.1, “ATM Addresses and Address Registration,” on page 6-2, for more information about address registration. If there are no addresses registered, the `ilmid` daemon on your system is not communicating properly with the switch.

1. Verify that there are incoming packets on VC 16 using `atmstat(1M)`. If there aren't any incoming packets, the switch is not responding to ILMI requests, and you should check its ILMI configuration.
 2. Verify that there are outgoing packets on VC 16 using `atmstat(1M)`. If you do not see any outgoing packets on VC 16, your interface is not transmitting ILMI packets. Verify that `ilmid` is running on your system, and if necessary, start it in the background. Starting `ilmid` with the `-v` flag causes it to print a notice for every message received or transmitted, along with other diagnostic information.
- Interfaces that are not running Classical IP or LAN Emulation will not appear in the output of the `ifconfig` command.

`ifconfig(1M)` displays interfaces that have been configured for IP. In order to support IP, ATM interfaces must run either Classical IP or LAN Emulation. Therefore, an ATM interface that is not configured to support IP by running one of these two protocols will not be displayed by `ifconfig`.

C.1.2 Classical IP Configuration

- Check all of the generic configuration points.

These are issues that apply to all SunATM interfaces, so they must all be working in order for Classical IP to work.

- Verify the output of `ifconfig(1M)`.

Executing the command `ifconfig -a` should display the SunATM interface, `baN`, where `N` is the instance number.

1. If your interface does not appear, an error probably occurred during the boot process.

Check for error messages during the boot process. The meanings and possible solutions for error messages can be found in Section C.2, "Error Messages."

2. If your interface appears, but has incorrect information, verify your configuration files.

The information given to `ifconfig` comes from the `/etc/atmconfig` and `/etc/aarconfig` files. Check the entries in those files that apply to this interface and verify their contents. For descriptions of the file formats, see

Section 5.1, “Editing the `/etc/atmconfig` File,” and Section 5.3.1, “Editing the `/etc/lanconfig` File,” or the `atmconfig(4)` and `aarconfig(4)` man pages.

- Check the `setup_state` with `aarstat(1M)`.

This command will provide information about the Classical IP status on your interface. The `setup_state` refers to the completion of the `aarsetup` program.

1. If the `setup_state` is `setup-started`.

This indicates that the `aarsetup` program has not completed; it may be delayed by slow switch responses, or failed attempts to register ATM addresses in `/etc/aarconfig`. Make sure that the local address given for your interface in `/etc/aarconfig` is unique to this switch. Using `$myaddress` and the reserved server addresses is a good way to guarantee that all addresses are unique. After making any changes to `/etc/aarconfig`, run `aarsetup` again.

2. If the state is not `setup-started` or `setup-finished`.

Verify that the addresses and interfaces in `/etc/aarconfig` are valid, and run `aarsetup` again. If you see any error messages, check their meanings in Section C.2, “Error Messages.”

- Verify the `interface_state` in `aarstat(1M)`.

The `interface_state` is either up or down, and reflects the linkstate given in the output of `qccstat`. If the linkstate is `DL_ACTIVE`, the `interface_state` is up; otherwise, the `interface_state` is down. If `aarstat` indicates that the `interface_state` is down, try the suggestions for a linkstate that is not `DL_ACTIVE` given in Section C.1.1, “Generic Configuration.”

- Make sure Classical IP is configured correctly.

The `aarstat(1M)` output lists several parameters for Classical IP. The field `arpcsmode` lists whether Classical IP is running as a client, a server, or stand-alone (a client with no server configured). Verify that this is correct; if it is not, check your `/etc/aarconfig` file entries.

- If the system is a Classical IP client, verify the server connection.

On systems running in client mode, `aarstat` also provides information about the server. Verify the server address, and that the `server_state` is connected.

- If the `server_state` is no-connection or connecting.

The system is likely having a problem establishing a connection to the server. Verify that the server address is correct, and that there is a system on the network which has registered that address. The server and applicable switch ports must also be configured to support UNI signalling, also called Q.2931 or Q.93b.

- Verify that addresses are resolved and connections are made with the `ping(1M)` command.

Once you have two systems configured and running to this point, they should be able to ping each other. On `client1`, type: `ping client2`. You should receive a response, after a small delay, of `client2 is alive`.

If the ping is not successful:

1. Check that ARP requests are being sent to the server.
Find the `server_vci` in the output of `aarstat`. Then run `atmstat`, and verify that there are outgoing packets on that VC. If not, make sure that your interface is up and configured properly.
2. Make sure that you are receiving ARP responses from the server.
In the `atmstat` output, check the output packets for the server VC (found in the `aarstat` information). If none are being received, your server is not responding to ARP requests from the client. If it is a SunATM server, verify its Classical IP status with the suggestions given here. If not, verify that it is up and running as a server.
3. Make sure the address is resolved correctly.
Run the `atmarp` command for the system you are trying to ping, and verify that its IP address has been resolved to the correct ATM address. If not, make sure that the remote system is registering the correct address with the ATM ARP server. If the address has not been resolved at all, make sure that the remote system has a connection to the server.

4. Verify that a connection has been established between the two systems. The output of `qccstat` lists the source and destination addresses of all open connections. You should have at least one connection to the server, and you should also see a connection to the remote host you are trying to ping. If not, make sure both interfaces are up and registered with the switch, and that both interfaces and the switch are running UNI signalling (Q.2931 or Q.93b).
5. Check for IP problems. If the address has been resolved correctly, and a connection has been established between the two systems, but they still cannot ping, the problem is likely outside the scope of ATM.

C.1.3 LAN Emulation Configuration

- Check all of the generic configuration points.

These are issues that apply to all SunATM interfaces, so they must all be working in order for LAN Emulation to work.

- Verify the output of `ifconfig(1M)`.

Executing the command `ifconfig -a` should display the ATM LAN Emulation interface, `laneN`, where *N* is the instance number.

1. If your interface does not appear, an error probably occurred during the boot process.

Check for error messages during the boot process. The meanings and possible solutions for error messages can be found in Section C.2, “Error Messages.”

2. If your interface appears, but has incorrect information, verify your configuration files.

The information given to `ifconfig` comes from the `/etc/atmconfig` and `/etc/laneconfig` files. Check the entries in those files that apply to this interface and verify their contents. For descriptions of the file formats, see Section 5.1, “Editing the `/etc/atmconfig` File” and Section 5.3.1, “Editing the `/etc/laneconfig` File,” or the `atmconfig(4)` and `laneconfig(4)` man pages.

- Check the `setup_state` with `lanestat(1M)`.

This command will provide information about the LAN Emulation status on your interface. The `setup_state` refers to the completion of the `lanesetup` program.

1. If the `setup_state` is `setup-started`.

This indicates that the `lanesetup` program has not completed; it may be delayed by slow switch responses, or failed attempts to register ATM addresses in `/etc/laneconfig`. Make sure that the local address given for you interface in `/etc/laneconfig` is unique to this switch. Using the variable `$myaddress` for all systems is a good way to guarantee that all addresses are unique. After making any changes to `/etc/laneconfig`, run `lanesetup` again.

2. If the state is not `setup-started` or `setup-finished`.

Verify that the addresses and interfaces in `/etc/laneconfig` are valid, and re-run `lanesetup`. If you see any error messages, check their meanings in Section C.2, “Error Messages.”

- Verify that a connection has been made to the LAN Emulation server (LES).

A LAN Emulation client must establish and maintain a connection to the LES. In most cases, the LES will also establish and maintain a second connection to the client. Find the LES address in the output of `lanestat`, and then look for connections with that address as the destination or source in the output of `qccstat`.

If you do not see any connections with that address:

1. If you have an LAN Emulation configuration server (LECS):

Make sure that the correct address is configured for the LECS. By default, the ATM Forum well-known address will be used by the SunATM software. If your LECS uses a different address, you should enter the alternate address in the `/etc/laneconfig` file. See Section 5.3.1, “Editing the `/etc/laneconfig` File,” for information on editing `/etc/laneconfig`. You can check the address currently being used in the output of `lanestat`.

2. If you do not have an LECS:

One of the LECS functions is to provide the LES address, so if you do not have an LECS, you must provide the address. This is accomplished with an entry in `/etc/lanconfig`. See Section 5.3.1, “Editing the `/etc/lanconfig` File.” You can check the LES address currently being used in the output of `lanestat`.

3. Verify that the LECS, if present, and LES are configured properly.

- Verify that a connection has been made to the BUS.

In addition to the LES connection(s), a LAN Emulation client must also establish and maintain a connection to the BUS, and the BUS will typically establish and maintain a second connection to the client. You can find the BUS ATM address in the output of `lanestat`, and then verify that there is a connection with that address as the destination, and probably a second with that address as source, in the output of `qccstat`. If there are not any connections, verify that the BUS is configured properly.

- Verify that the host has joined the Emulated LAN

The `lanestate` field in the output of `lanestat` should indicate that the client is in the active state.

If your system is not able to join the emulated LAN, there may be a problem with the way in which your LAN Emulation Services are configured. If the Emulated LAN uses an MTU size larger than 9 kilobytes, the SunATM host will not join (9 Kbytes is the largest MTU size supported by the SunATM product). If the host is not able to join, an error message will be printed with an explanation.

- Verify that addresses are resolved and connections are made with the `ping` command.

Once you have two systems configured and running to this point, they should be able to `ping` each other. On `client1`, type `ping client2`. You should receive a response, after a small delay, of `client2 is alive`.

If the `ping` is not successful:

1. Check that the IP hostname or address is resolved to a MAC address.

LAN Emulation requires two address resolution steps to make a call. The first is to resolve an IP address to a MAC address. From the perspective of IP and ARP, this works exactly like it does on an ethernet interface; using

the `arp` command, you can verify that this resolution has been made correctly. If it has not, verify the connections to the BUS, and make sure data is being transmitted and received on the connection(s) to the BUS by finding the VC in the output of `qccstat`, and looking at the statistics for that VC in `atmstat`.

2. Check that the MAC address has been resolved to an ATM address. This is the second address resolution step, and is accomplished by the LAN Emulation software and communication with the LES. You can use the `lanearp` command to verify that MAC addresses have been properly resolved to ATM addresses. If it has not, verify the connections to the LES, and make sure data is being transmitted and received on the connection(s) to the LES by finding the VC in the output of `qccstat`, and looking at the statistics for that VC in `atmstat`.
3. Verify that a connection has been established between the two systems. The output of `qccstat` lists the source and destination addresses of all open connections. There you should see a connection to the remote host you are trying to `ping`. If not, make sure both interfaces are up and registered with the switch, and that both interfaces and the switch are running UNI signalling (Q.2931 or Q.93b).
4. Check for IP problems. If the address has been resolved correctly, and a connection has been established between the two systems, but they still cannot `ping`, the problem is likely outside the scope of ATM.

C.2 Error Messages

This section includes a list of some of the most common error messages you might see while configuring and bringing up your SunATM interface. For each message, there is a brief explanation of the problem and a possible solution.

C.2.1 Error Messages from S00sunatm

```
Cannot find ATM utilities in /etc/opt/SUNWatm/bin;  
exiting S00sunatm.
```

The SunATM utility directory `/etc/opt/SUNWatm/bin` does not exist. Make sure that the SUNWatm package installation completed successfully (see Section 3.2.4, “Checking the SunATM Software Installation Using `pkginfo`,” on page 3-6, for more information). If necessary, the package may need to be re-installed.

```
Cannot find atmconfig file in /etc; exiting S00sunatm.
```

The `/etc/atmconfig` file provides configuration information to the S00sunatm script so that it can bring up the SunATM interfaces during system boot. If the `/etc/atmconfig` file is not present, S00sunatm will print this warning message and exit. The `/etc/atmconfig` file is installed with the SUNWatm package as `/etc/atmconfig.template`; if you choose autoconfiguration or if no previous `/etc/atmconfig` file exists, `pkgadd` will copy this template file to `/etc/atmconfig`. If a previous `/etc/atmconfig` file exists, it will not be overwritten. See Section 5.1, “Editing the `/etc/atmconfig` File,” for more information on this file.

```
warning: can't plumb <device>; no UNI version provided
```

The first entry in `/etc/atmconfig` for a physical interface must include a UNI value in the second field.

```
warning: can't plumb <uni version> on <device>; <uni  
version> already plumbed
```

This message is printed when an entry is encountered which attempts to plumb a signalling version on an interface which has already been plumbed with a different signalling version. The script will ignore the new UNI version and continue processing the entry and the remaining entries in the file.

```
warning: can't plumb <lane instance>: too many lane
instances on <device>
```

A physical interface will support up to n lane instances, where n is the number of MAC addresses on the board (or 1 if the board has no MAC address). The number of MAC addresses on a board may be checked using the count option to the `atmgetmac(1m)` command. If an entry is encountered which attempts to plumb more LANE instances than allowed, this message will be printed; processing will continue with the next entry in the file.

```
warning: can't plumb signalling on <device>
warning: can't plumb classical IP interface <device>
warning: can't plumb <lane instance> on <device>
```

An error occurred when the script attempted to run `atmplumb(1m)`, either to plumb signalling, classical IP, or LAN Emulation on an interface, with information specified in `/etc/atmconfig`. The `atmplumb` program will generally print out an error message indicating why it failed; use that information to check your values in the `/etc/atmconfig` entry for device. The script will proceed to read and process the remaining entries in `/etc/atmconfig`, although further entries for the failed interface will not be processed correctly.

```
warning: invalid interface <lane instance>
```

The minor number provided in a logical interface name was not in the range 0 - 255. The script will proceed without attempting to configure the invalid lane device.

```
warning: only one classical ip hostname is allowed on
<device>
```

An additional entry was found containing a Classical IP hostname after an initial Classical IP hostname was already plumbed for device. Multiple Classical IP instances are not supported on a single physical interface. The script will ignore additional Classical IP information for a physical interface.

```
warning: <laneN> entry must appear before <laneN:X> entry
```

When using logical interface names, the first entry in `/etc/atmconfig` must always be either `laneN` or `laneN:0`, which are equivalent. All entries that appear before the `laneN` or `laneN:0` entry will be ignored.

Please install <SUNWatm>

A required software package is not installed on the system. Install the package, and reboot the system.

warning: extra fields for <device> will be ignored

There were additional fields in the `/etc/atmconfig` entry for the given device name. The script will proceed, ignoring the additional fields.

warning: duplicate entry <lane device>

There were multiple entries in `/etc/atmconfig` using the same LAN Emulation instance number. This is not a fatal error; the script will continue to run. However, only the first entry for each LAN Emulation instance number will be configured for LAN Emulation.

warning: not enough fields to configure <device>

The `/etc/atmconfig` entry for the given device did not have all the required fields. You must edit `/etc/atmconfig` file (see Section 5.1, “Editing the `/etc/atmconfig` File”), filling in all the appropriate information, and reboot the system. Empty fields should be indicated with a hyphen (-).

warning: ifconfig failed for classical IP
interface <device>

warning: ifconfig failed for <lane instance>

An error occurred when the script attempted to run `ifconfig` for the specified interface. You should see error messages printed by `ifconfig` indicating why it failed; use that information to check your values in `/etc/atmconfig`. In particular, verify that the hostname you provide in `/etc/atmconfig` appears in the `/etc/hosts` file on your system.

warning: invalid lane instance (<lane instance>) for
<device>

The lane instance number provided in `/etc/atmconfig` was not in the range 0 to 999. The script will proceed without attempting to configure the invalid lane instance.

```
warning: aarsetup failed; could not configure classical IP
interfaces
```

```
warning: lanesetup failed; could not configure LAN
Emulation interfaces
```

Either the LAN Emulation or the Classical IP startup script failed and exited with an error value. Check the error messages that were printed by `aarsetup` or `lanesetup`, and verify the values you have entered in `/etc/aarconfig` and/or `/etc/laneconfig`.

C.2.2 Error Messages from aarsetup and lanesetup

```
aarsetup: could not become control process
```

```
lanesetup: could not become control process
```

An instance of the setup program was running when another instance was started up. The second instance exits with this error message. Make sure that there is not a previous instance of the program still running. The setup program might take a while to complete if the switch is slow to respond.

```
aarsetup: could not open stream to Q93B
```

```
lanesetup: could not open stream to Q93B
```

The program was not able to communicate with the Q93B driver. Make sure that you run `aarsetup` or `lanesetup` as root, and that the SUNWatm package has been properly installed.

```
aarsetup: could not scan input file
```

```
lanesetup: could not scan input file
```

The program was unable to open the `/etc/aarconfig` or `/etc/laneconfig` file (or the file specified on the command line). Verify that the appropriate file exists, and has the proper permissions. Also make sure you run `aarsetup` or `lanesetup` as root.

```
aarsetup: exiting because of errors
```

lanesetup: exiting because of errors

Errors were encountered while parsing the `/etc/aarconfig` or `/etc/laneconfig` file, so the setup program cannot successfully complete. The error condition should be corrected, and then you should execute either `aarsetup` or `lanesetup`.

`aarsetup: <interface> running as a server, but PVC-only 't' entries exist`

The `aarsetup` program has found an *L* entry in `/etc/aarconfig`, meaning that this interface will be running as a server; however, there are table entries (*t* entries) containing only PVCs, which cannot be entered into the server's ATM ARP table. Verify your interface's status (server, client, or stand-alone), make sure all *t* entries include ATM addresses, and execute `aarsetup`. See Section 5.2.1, "Editing the `/etc/aarconfig` File," on page 5-4, for more information.

`aarsetup: waiting for ilmid to provide prefix`

`lanesetup: waiting for ilmid to provide prefix`

In some cases, the address registration process may take several minutes. In this case, `aarsetup` or `lanesetup` will print out this message to notify the user that it cannot complete until address registration completes. If the messages continue for more than a minute or two, verify your connection to the switch and that the switch and interface are both supporting ILMI.

`undefined variable`

A variable was used in a configuration file without being assigned a value with a `set` statement. Add a `set` statement, or correct the variable name, and run `aarsetup` or `lanesetup` again. See Section 5.2.2, "Using Variables in the `/etc/aarconfig` File," on page 5-8, and Section 5.3.2, "Using Variables in the `/etc/laneconfig` File," on page 5-18, for more information.

`variable already defined`

An attempt was made to set a variable which had been previously set in the same configuration file. Remove the second assignment and run `aarsetup` or `lanesetup` again.

variable name ill-formed

An attempt was made to create a variable in `/etc/aarconfig` or `/etc/lanconfig`, but the variable name was syntactically invalid. Variable names should be a combination of letters, digits, and underscores (`_`). Choose a conforming variable name and run `aarsetup` or `lanesetup` again.

variable name too long

An attempt was made to create a variable in `/etc/aarconfig` or `/etc/lanconfig`, but the variable name was greater than the maximum length (32 characters). Choose a variable name of less than 32 characters and run `aarsetup` or `lanesetup` again.

variable value too long

The value assigned to a variable in a configuration file was longer than the maximum value length of 128 characters. If a longer value is desired, try using a combination of variable names, with each value less than 128 characters. After correcting the variable value lengths, run `aarsetup` or `lanesetup` again.

```
ifname: cannot join ELAN (frame size too large; please use  
a different ELAN and rerun lanesetup)
```

The largest MTU size supported by the SunATM software is 9 kilobytes. If the LAN Emulation Services try to set a size larger than 9 Kbytes, the SunATM client will not be able to join the emulated LAN. Reset your LAN Emulation services to use an MTU size less than or equal to 9 kB, and rerun `lanesetup` to join the emulated LAN.

```
ifname: frame-size change (please rerun lanesetup)
```

The MTU size was changed by the LAN Emulation Services, and `lanesetup` must be rerun to notify IP of the change. There is a slight chance that TCP connections will remain open during this change, and if that is the case, performance on those connections will be impacted by the change. You should either restart the affected applications, or reboot the system if this becomes a problem.

<ifname> could not download the MAC address

This message indicates that an error occurred while `lanesetup` was attempting to retrieve a MAC address for the indicated interface. The most likely causes of such an error are that the kernel is out of memory, or that `atmplumb` has not been run for the specified interface.

Could not find driver for <ifname>

Each LAN Emulation interface is associated with an ATM driver when LAN Emulation is set up by `atmplumb`. This message indicates that this interface/driver association has not been made, most likely because `atmplumb` has not been run for the specified interface.

Not enough MAC addresses on <ATM interface>

The number of Emulated LANs which may be joined over a single physical interface is limited by the number of MAC addresses on the ATM interface board. This message indicates that an attempt was made to join more Emulated LANs than allowed by the number of MAC addresses on the specified interface. The number of MAC addresses on an interface may be found using the `count` option on the `atmgetmac(1M)` command; the number of Emulated LANs and lane instances indicated in `/etc/atmconfig` and `/etc/laneconfig` should not exceed this number. See Section 5.5, “Supporting Multiple Emulated LANS on a Single Interface.”

C.2.3 Error Messages from the Kernel Drivers

```
q93b: warning: link coming back up on <interface>, but  
ilmid is not running
```

The link has gone down and come back up on an interface, but `ilmid` is not running at this time. This is a problem because addresses must be registered with the switch again, since both the interface and switch must clear out their address tables when the link goes down. Start `ilmid`; if the interface does not seem to be running properly after doing this, you may need to reboot the system. It is likely that the interface was in an unusual or unknown state when the link came back up, and may need to be taken down completely by rebooting.

Managing SunATM Interfaces with SNMP



The SunATM software package provides an SNMP (Simple Network Management Protocol) agent which supports the ATM UNI and LAN Emulation Management Information Bases (MIBs) defined in the User Network Interface and LAN Emulation Specifications. This agent will provide information to a network management system, such as the SunNet Manager™ system.

D.1 Installing the SunATM SNMP Software

The SunATM SNMP software is made up of three parts: the SunATM SNMP daemon, `atmsnmpd`; the SNMP management console configuration files; and the agent configuration files. The management configuration files, which are installed in the `/opt/SUNWatm/snmp` directory, are part of the SUNWatmu package. The ATM SNMP daemon and its configuration files, which are part of the SUNWatm package, are installed in the `/etc/opt/SUNWatm/bin` and `/etc/opt/SUNWatm/snmp` directories, respectively.

D.2 Setting Up the Management Console

The schema and oid files containing the required ATM MIB definitions for SunNet Manager are installed in `/opt/SUNWatm/snmp`. In addition, the MIB files in abstract syntax notation (ASN.1) format are included if you are running a network manager that does not use schema files. Refer to the documentation for your network manager for information on how to generate the appropriate configuration files from the MIB files provided.

To configure your SunNet Manager console system to recognize SunATM agents:

1. Start the `snm` console program and save your management database.

This can be done using File -> Save -> Management database from the `snm` console menus.

2. Copy the SunATM schema and oid files which were installed on the SunATM hosts to the schema directory on the manager.

The files are called `atmf.mib.schema`, `lane.mib.schema`, `atmf.mib.oid`, and `lane.mib.oid`, and are installed in the `/opt/SUNWatm/snmp` directory on SunATM hosts. They should be copied to the `/opt/SUNWconn/snm/agents` directory on the management system.

3. Build the object identifier database to include the SunATM object identifiers.

Do this by executing the following command on the management console system:

```
# /opt/SUNWconn/bin/build_oid /opt/SUNWconn/snm/agents
```

4. Start the `snm` console with the `-i` flag:

```
# snm -i
```

5. Load your management database using File -> Load -> Management database from the menus.

The SunATM MIBs, `atmf.mib` and `lane.mib`, should now be available when you create or update a component.

Note – For further information on using SunNet Manager to monitor `snmp` agents, refer to the SunNet Manager documentation.

D.3 Setting Up Agent Systems

To configure a SunATM host system to run as an SNMP agent, select the System Parameters option on the `atmadmin` main menu. From this option, you will be given the choice of setting your system to run as a SNMP agent or not (see Section 4.3, “Using the `atmadmin` Configuration Program,” for more information about the `atmadmin` configuration program).

Note – This option applies to the entire system, and not for each SunATM interface.

Whether the system is running as an agent or not, the daemon must be running, since it communicates with other parts of the SunATM software. If the system is configured to run as an ATM SNMP agent, the daemon will bind to the UDP port used by SNMP network managers (port 161). If the system is not configured as an agent, the agent will not bind to this port, and it will not respond to requests from network management software.

The default community values for the SunATM agent are public for read and private for write. If you wish to change these values, they should be changed in the `/etc/opt/SUNWatm/snmp/agent.cnf` file. This file contains SNMP agent configuration information, and you may customize these values as needed. The `atmsnmpd` daemon must be restarted after any changes to any of its configuration files, including the `agent.cnf` file.

Application Programmers' Interface



The Application Programmers' Interface (API) that is provided with this software release is an interim API to be used until the ATM Forum standardizes an API.

Note – Be aware that since this is an interim API, it can be changed at any time.

Note – For historical reasons, Q.93B and Q.2931 are used interchangeably.

The signalling API, called Q.2931 Call Control (qcc), consists of two sets of similar functions: one for applications running in the kernel, and one for applications running in user space. Each set provides functions to build and parse Q.2931 signalling messages, which are required to set up and tear down connections.

One additional function is provided to assist applications in establishing appropriate connections to the q93b driver. `q_ioc_bind` associates a Service Access Point (sap) with the specified connection to the q93b driver. The sap is used by the driver to direct incoming messages to applications.

An additional set of functions is provided to facilitate communication with the ATM device driver (the `ba` driver in SunATM software). These functions are referred to collectively as the `atm_util` functions.

For examples of user applications that use the SunATM API, see the sample programs installed in `/opt/SUNWatm/examples`.

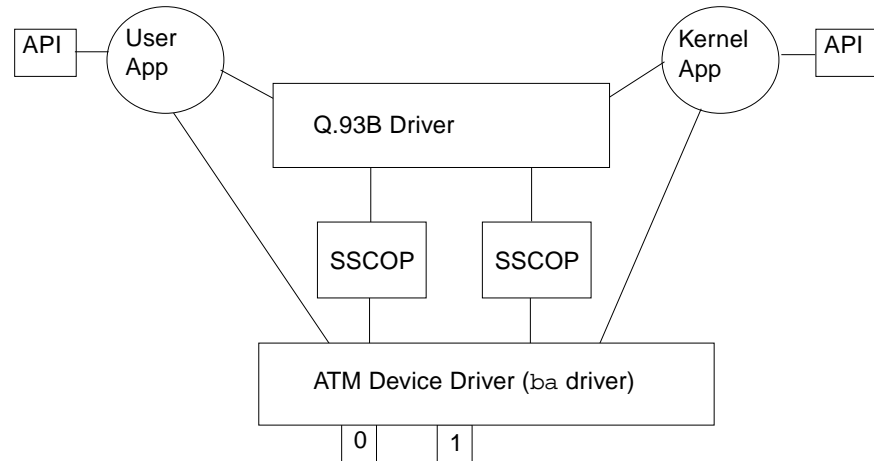


Figure E-1 ATM Signalling

E.1 Using the SunATM API with the q93b and the ATM Device Drivers

The architecture illustrated in Figure E-1 must be established on a SunATM system in order to perform Q.2931 signalling and send data over established connections. The ATM device driver, SSCOP modules, and q93b driver are “plumbed” at boot time. The task remaining for application developers is to create the connections between their application and the q93b and ATM device drivers.

Both the q93b and ATM device driver are STREAMS drivers; connecting to them is for the most part no different than connecting to other STREAMS drivers. The following sections describe the steps required to connect to each driver, use the drivers to establish ATM connections, and send data over those connections.

E.1.1 Establishing a Connection to the q93b Driver

The `open(2)` system call should be used first to obtain a file descriptor to the driver. After opening the driver, `q_ioc_bind` should be called, associating in the q93b driver a service access point (sap) with this application. Finally, if the application is a kernel driver, it should be linked above the q93b driver, using the `I_LINK` or `I_PLINK` ioctl (refer to the `streamio(7)` man page for information about this ioctl).

E.1.2 Setting up an ATM Connection Over a Switched Virtual Circuit (SVC)

After connecting to the q93b driver, either by directly calling the functions as a user application, or by having a setup program connect your application driver as described in the preceding section, the q93b driver is available to your application to establish Switched Virtual Circuits (SVCs) using the Q.2931 signalling protocol. The Q.2931 message set is displayed in Table E-1.

Table E-1 Messages Between the User and the q93b Driver

Message Type	Direction
SETUP	BOTH
SETUP_ACK	UP
CALL_PROCEEDING	BOTH
CONNECT	BOTH
CONNECT_ACK	UP
RELEASE	DOWN
RELEASE_COMPLETE	BOTH
STATUS_ENQUIRY	DOWN
STATUS	UP
RESTART	BOTH
RESTART_ACK	BOTH

UP is from q93b to user;
DOWN is from user to q93b

The q93b driver is an M-to-N mux STREAMS driver. Multiple application programs may be plumbed above the driver, and multiple physical interfaces may be connected below q93b. Applications may access any or all of the physical interfaces, and messages received on the physical interfaces may be directed to any of the applications. In order to direct messages through the q93b driver, messages from applications must include a physical interface name to identify the outgoing interface, and a sap to identify the application to which the message should be directed on the receiving host.

Messages sent to q93b by applications should be sent in the format illustrated in Figure E-2; kernel applications should use `put(9F)` to send the mblocks shown, and user applications should send two corresponding strbufs using `putmsg(2)`.

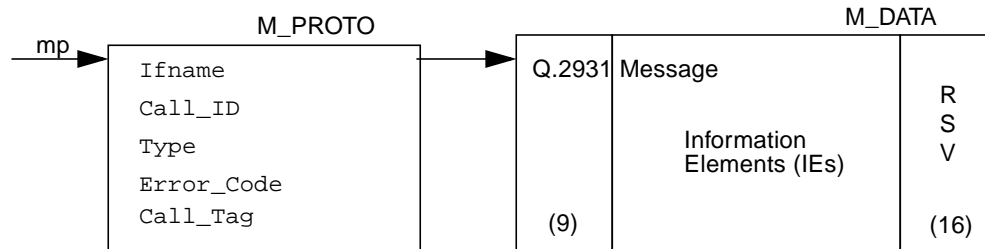


Figure E-2 Message Format

Table E-2 Fields in the M_PROTO mblock

Message	Explanation
<code>Ifname</code>	A null-terminated string containing the device name
<code>Call_ID</code>	A unique number from q93b per interface.
<code>Type</code>	The same as the Q.2931 message type except there is a local Non-Q.2931 message type <code>SETUP_ACK</code> . The <code>SETUP_ACK</code> message is used to provide the <code>Call_ID</code> to the user.
<code>Error_Code</code>	The error returned from q93b when an erroneous message is received from the user. The exact same mblock chain shall be returned to the user with the <code>Error_Code</code> field set. The user must always clear this field
<code>Call_Tag</code>	A number assigned by the calling application layer to a <code>SETUP</code> message. When a <code>SETUP_ACK</code> is received from q93b, the <code>Call_ID</code> has been set; the <code>Call_Tag</code> field may be used to identify the ack with the original request. From that point on, the <code>Call_ID</code> value should be used to identify the call.

The structure that is included in the `M_PROTO` mblock is defined as the `qcc_hdr_t` structure in `<atm/qccotypes.h>`. In the second mblock, the application shall leave the Q.2931 header portion (9 bytes) of the Q.2931 message blank; this information is filled in by the q93b driver. The application should also reserve 16 bytes at the end of the second mblock for the layer 2 (Q.SAAL) protocol performance. The `qcc` functions may be used to create messages in this format.

The following sections give a brief overview of Q.2931 signalling procedures, from the perspective of an application using the SunATM API. For more details on the procedures, refer to the ATM Forum's User Network Interface Specification, version 3.0 or 3.1. For further information on the qcc functions, which are outlined in Table E-3, see the appropriate man pages in section 3 (for user applications) or section 9F (for kernel applications). The man pages can be accessed under the function group name, or any specific function name. For example, the man page which documents the `qcc_bld_*` function group may be accessed by typing: `man qcc_bld`, `man qcc_bld`, `man qcc_bld_setup`, or `man qcc_bld_connect`, etc. The message flow during typical call setup and tear down is diagrammed in Figure E-3 on page E-10.

Table E-3 qcc Functions

Name	Functionality	Input	Output
<code>qcc_bld_*</code>	Creates and encodes a message; allows customization of a limited set of values, depending on the message type. Configurable values are passed in as parameters.	Parameter values	Encoded Q.2931 message (in the format shown in Figure E-2)
<code>qcc_parse_*</code>	Extracts a defined set of values from an encoded message	Encoded Q.2931 message (in the format shown in Figure E-2)	Parameter values
<code>qcc_len_*</code>	Returns the maximum length of the buffer that should be allocated for the second strbuf in a Q.2931 message. Only applicable to user space applications; the kernel API allocates the buffers inside the <code>qcc_bld/qcc_pack</code> functions.	none	Maximum length of the message.
<code>qcc_create_*</code>	Create a message structure with the required values set. The structure may then be further customized using <code>qcc_set_ie</code> .	Default parameter values	Message structure (defined in <code><atm/qcctypes.h></code>)
<code>qcc_set_ie</code>	Updates or inserts values for an information element into a message structure.	Message structure and IE structure (defined in <code><atm/qcctypes.h></code>)	Updated message structure

Table E-3 qcc Functions (Continued)

Name	Functionality	Input	Output
qcc_pack_*	Takes a message structure and encodes it into an actual Q.2931 message, consisting of the two mblks (or strbufs) illustrated in Figure E-2.	Message structure (defined in <atm/qccotypes.h>)	Encoded Q.2931 message (in the format shown in Figure E-2)
qcc_unpack_*	The reverse of qcc_pack_*: takes an encoded message and decodes the data into a message structure.	Encoded Q.2931 message (in the format shown in Figure E-2)	Message structure (defined in <atm/qccotypes.h>)
qcc_get_ie	Extracts a single information element structure from a message structure.	Message structure and empty IE structure (defined in <atm/qccotypes.h>)	Updated IE structure

E.1.2.1 Call Setup

When the user decides to make a call, the user sends a SETUP message down to q93b and waits for a SETUP_ACK from q93b. The SETUP message should include a Broadband Higher Layer Information (BHLL) information element which contains a four-octet sap identified as User Specific Information. The sap is used to identify the application to which the message should be directed by q93b on the receiving host. After receiving a SETUP_ACK with a 0 error field, the user waits for either a CALL_PROCEEDING, CONNECT, or RELEASE_COMPLETE message from q93b (all other messages are ignored by q93b). After the CONNECT message is received, the user may use the virtual channel.

When the user receives a SETUP message from q93b, the user shall respond with either a CALL_PROCEEDING, CONNECT, or RELEASE_COMPLETE message to q93b. After the CONNECT_ACK message is received, the user may use the virtual channel.

E.1.2.2 Release Procedure

To clear an active call or a call in progress, the user should send a RELEASE message down to q93b and wait for a RELEASE_COMPLETE from q93b. Any time the user receives a RELEASE_COMPLETE message from q93b, the user shall release the virtual channel if the call is active or in progress.

q93b never sends a RELEASE message to the user; it will always send a RELEASE_COMPLETE. The user only sends the RELEASE_COMPLETE message when rejecting a call in response to a SETUP message from q93b. At any other time, to reject or tear down a call, the user shall send a RELEASE message to q93b.

E.1.2.3 Exception Conditions

If for any reason q93b cannot process a SETUP message received from a user, the SETUP_ACK is returned with an error value set, and call setup is not continued. The error value will be one of the cause codes specified in section 5.4.5.15 of the ATM Forum UNI standard.

E.1.3 Connecting, Sending, and Receiving Data with the ATM Device Driver

Connecting to the ATM device driver involves several steps, which include several IOCTL calls. In order to create a more standardized interface for user space applications, a set of `atm_util` functions is available to application writers. An overview of those functions is provided in Table E-4. For more detailed information, refer to the `atm_util(3)` man page. The `ba(7)` man page contains a more detailed discussion of the driver-supported IOCTLs.

Table E-4 `atm_util` Function Overview

Name	Functionality	Kernel Equivalent
<code>atm_open</code>	Open a stream to the ATM device driver	must be done by a user space setup program
<code>atm_close</code>	Close a stream to the ATM device driver	must be done by a user space setup program
<code>atm_attach</code>	Attach to a physical interface	must be done by a user space setup program
<code>atm_detach</code>	Detach from a physical interface	must be done by a user space setup program
<code>atm_bind</code>	Bind to a Service Access Point	send DL_BIND_REQ
<code>atm_unbind</code>	Unbind from a Service Access Point	send DL_UNBIND_REQ

Table E-4 atm_util Function Overview (Continued)

Name	Functionality	Kernel Equivalent
atm_setraw	Set the encapsulation mode to raw	send DLIOCRAW
atm_add_vpci	Associate a vpci with this connection	A_ADDVC ioctl
atm_delete_vpci	Dissociate a vpci from this connection	A_DELVC ioctl
atm_allocate_bw	Allocate constant bit rate bandwidth for this connection	A_ALLOCBW ioctl
atm_allocate_cbr_bw	Allocate constant bit rate bandwidth with more granularity than atm_allocate_bw	A_ALLOCBW_CBR ioctl
atm_allocate_vbr_bw	Allocate variable bit rate bandwidth	A_ALLOCBW_VBR ioctl
atm_release_bw	Release previously allocated bandwidth	A_RELSE_BW ioctl

Note – In the following discussion, the user space function names are used. Refer to Table E-4 for the corresponding kernel space function or IOCTL.

To establish a data path, the application must first open the ATM driver and attach to a specific physical interface using `atm_open()` and `atm_attach()`. Next, the connection should be associated with one or more VC(s), using `atm_add_vpci()`. If a call has been established using Q.2931 signalling, the vpci provided to `atm_add_vpci()` should be the vpci that was included in the Q.2931 signalling messages received while establishing the call.

An encapsulation method must also be selected. The SunATM device driver supports raw (null) and DLPI encapsulation. Messages sent in raw mode are sent as data only, with just a four-byte vpci as a header; DLPI mode messages are LLC-encapsulated. By default, a connection is in DLPI mode; to change the encapsulation to raw, DLIOCRAW should be set using `atm_setraw()`. The remaining steps depend on the encapsulation mode selected.

E.1.3.1 Raw Mode Connections

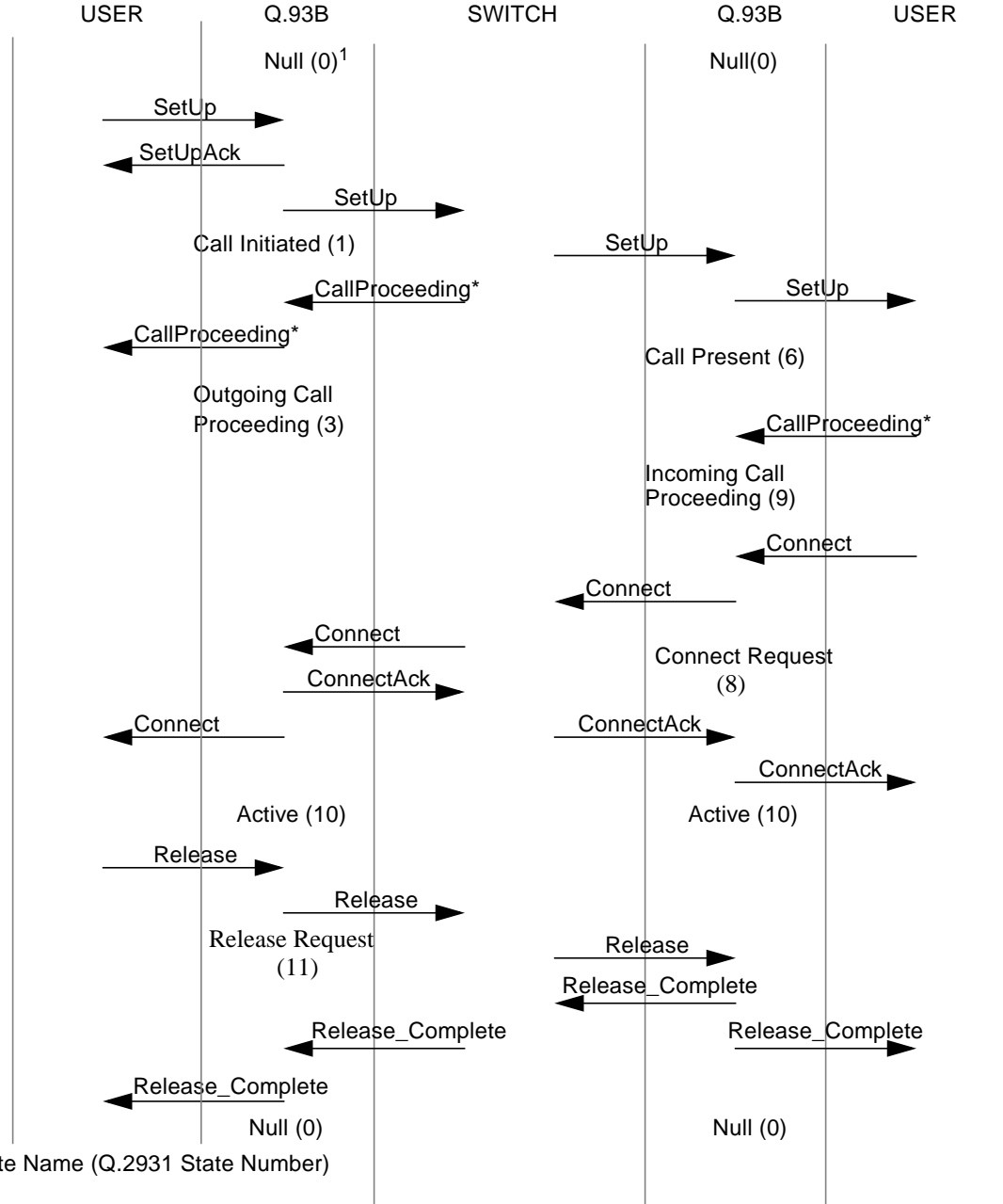
If raw mode is chosen, the only remaining configuration step is to allocate an amount of bandwidth for the use of this connection, using `atm_allocate_bw()`, `atm_allocate_cbr_bw()`, or `atm_allocate_vbr_bw()`.

From the perspective of the application/driver interface, raw mode implies that only a single message buffer (pointed to by `dataptr` in `putmsg(2)`) should be sent to the driver, containing a four-byte `vpci` followed by the data. When a message is received on a `vpci` running in raw mode, it will be directed to an application based on the `vpci`. When sending a received message up to the application, the driver will strip the four-byte `vpci` from the message if the application has not set `DLIOCRAW` with a call to `atm_setraw`; if `DLIOCRAW` has been set, the four-byte `vpci` will be included in the message sent up to the application.

E.1.3.2 DLPI Encapsulated Connections

If DLPI mode is chosen, a `sap` must be associated with the connection using `atm_bind()`. Optionally, a specific amount of bandwidth may be allocated for the connection using `atm_allocate_bw()`, `atm_allocate_cbr_bw()`, or `atm_allocate_vbr_bw()`. If bandwidth is not explicitly allocated, IP's bandwidth (which includes all available unallocated bandwidth) will be shared by the connection.

DLPI mode implies that two message buffers will be sent to the driver. The first, pointed to by `ctlptr` in `putmsg(3)`, contains the `dlpi` message type, which is `dl_unitdata_req` for transmit and `dl_unitdata_ind` for receive. The `vpci` is included in this buffer as well; the format for the buffer is defined in the header file `<sys/dlpi.h>`. The second buffer, pointed to by `dataptr` in `putmsg(3)`, contains the data. When the driver receives the two buffers from the application, it will remove the first buffer, add a LLC header containing the `sap` which has been bound to this stream to the data buffer, and transmit it. On receive, the LLC header is stripped, the control buffer is added with the DLPI header, and the two buffers are sent up to the application indicated by the `sap` in the LLC header.



¹ XX(n): Q.2931 State Name (Q.2931 State Number)
 * Optional

Figure E-3 Normal Call Set up and Tear Down

Running Diagnostic Tests



F.1 SunVTS Validation and Test Suite

The `nettest` diagnostic test checks all the networking interfaces on a system, including the SunATM SBus adapter.

In order to use the `nettest` diagnostic, you must have the SunVTS™ Validation and Test Suite installed on your system. The SunVTS software runs diagnostic tests on Sun systems.

Note – You must have Classical IP up and running on an interface for the `nettest` to work.

Refer to the *SunVTS User's Guide* for more information on how to run the `nettest` diagnostic.

F.2 Using the OpenBoot PROM Selftest

The SunATM SBus adapter's selftest verifies correct operation of the SBus adapter. The selftest consists of a suite of tests that reside in the FCode PROM on the adapter. The code is written in Forth programming language and can only be run under OpenBoot PROM (OBP) version 2.x or later.

The SunATM SBus adapter's selftest does not automatically run after power on or reset, but you can use selftest any time you want to determine the status of the hardware.

Note – Selftest does not require connection to the network. The selftest will test the internal loopback up to the Saturn User Network Interface (SUNI) ASIC.

Running the SunATM SBus Adapter's Selftest

1. As a Sun system is powered up, the following banner is displayed:

```
4-slot Ultra Enterprise 3000, No Keyboard
OpenBoot 3.2.3, 64 MB memory installed, Serial #7715685.
Ethernet address 8:0:20:75:bb:65, Host ID: 8075bb65.

Type b (boot), c (command), n (new command)
>n
ok
```

- Check that the OpenBoot PROM version is 2.x or later.
- If the system is set up to automatically boot, press key combination Stop(L1)-A to stop it.
- If the system is not already at the `ok` prompt, type `n` for the new command mode to get to the `ok` prompt.

2. Type the `show-devs` command to display all of the devices on the system. To find the path to the SunATM SBus adapter, look for the `SUNW,ba` device in the list of devices. In the following example, the `SUNW,ba` device is in bold type.

```
ok show-devs
/TI,TMS390Z50@f,f8ffffc
/eccmemctl@f,0
/virtual-memory@0,0
/memory@0,0
/obio
/iommu@f,e0000000
/openprom
/aliases
/options
/packages
/obio/power@0,a01000
/obio/auxio@0,800000
/obio/SUNW,fdtwo@0,700000
/obio/interrupt@0,400000
/obio/counter@0,300000
/obio/eprom@0,200000
/obio/zs@0,0
/obio/zs@0,100000
/iommu@f,e0000000/sbus@f,e0001000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,ba@3,0
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,bpp@f,4800000
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000
/iommu@f,e0000000/sbus@f,e0001000/SUNW,DBRIe@f,8010000/mmcdec
/iommu@f,e0000000/sbus@f,e0001000/ledma@f,400010/le@f,c00000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/st
/iommu@f,e0000000/sbus@f,e0001000/espdma@f,400000/esp@f,800000/sd
/packages/obp-tftp
/packages/deblocker
/packages/disk-label
<#0> ok
```

3. To run the SunATM SBus adapter's selftest, type `test` and the pathname to the SUNW,ba device:

```
ok test /iommu@f,e0000000/sbus@f,e0001000/SUNW,ba@3,0
Register Test -- succeeded.
Memory Test -- succeeded.
622 SAHI Internal Loopback Test -- succeeded.
622 SUNI Internal Loopback Test -- succeeded.
ok
```

Note – If the `test` command fails, verify that the SBus adapter hardware is installed correctly. If necessary, replace the SBus adapter and/or contact your service provider.

For more information on using the OpenBoot PROM commands, refer to the *Open Boot Command Reference Manual*, which was shipped with the Solaris documentation.

Glossary

anymac

A predefined SunATM wild card variable which represents any 6-byte ESI. This variable should only be used with *a* entries in the `/etc/aarconfig` or the `/etc/laneconfig` configuration files.

anymacsel

A predefined SunATM wild card variable which represents any 7-byte ESI and Selector combination. This variable should only be used with *a* entries in the `/etc/aarconfig` or the `/etc/laneconfig` configuration files.

ATM ARP

ATM Address Resolution Protocol. Both *Classical IP* and *LAN Emulation* provide a type of ATM ARP.

ATM Address

A 20-byte (the bytes are often referred to as octets) number which uniquely identifies an ATM endpoint. The first 13 bytes are assigned by the switch, and are called the *switch prefix*; the remaining 7 bytes are made up of a 6-byte *end system identifier* (*esi*) and a 1-byte *selector*, and are assigned by the local host. This documentation refers to those 7 bytes as the local portion of the ATM address.

BUS

Broadcast and Unknown Address Server. This is one of the servers required to support an Emulated LAN environment.

Classical IP

A specification to provide support for the Internet protocol over an ATM network.

ELAN

The Emulated LAN created in an ATM environment in which the systems are using LAN Emulation to provide ATM support for IP.

ESI

See *End System Identifier*

Emulated LAN Name

The character string which identifies a particular emulated LAN. Some LAN Emulation Services require that the LAN Emulation client provide an Emulated LAN Name when attempting to join the Emulated LAN.

End System Identifier

The 6-byte portion of an *ATM address* that uniquely identifies the end system. The local *MAC address* is often used as the End System Identifier, since it is a value unique to the host. The ESI and 1-byte *selector* make up the local portion of an *ATM address*.

ILMI

Interim Local Management Interface. This protocol is used to exchange address information between an ATM switch and an ATM endpoint.

ITU-TS

International Telecommunications Union - Telecommunication Sector. A branch of the ITU which develops telephony standards. Formerly known as the International Consultative Committee for Telegraph and Telephone (CCITT).

LAN Emulation

A specification to provide support for the IP protocol over an ATM network.

LECS

The LAN Emulation Configuration Server. This is one of the servers required to support an Emulated LAN environment.

LES

The LAN Emulation Server. This is one of the servers required to support an Emulated LAN environment.

LIS

Logical IP Subnetwork.

Local Address

The 7-byte portion of an *ATM address* made up of the 6-byte *End System Identifier* and the 1-byte *Selector*. The Local Address is the part of the *ATM address* that is assigned by the local system.

localswitch_server

A predefined SunATM configuration file variable which is the concatenation of `$prefix`, a unique reserved MAC address, and `$sel`. When used as a server address, restricts server access to clients connected to the local switch only. See *prefix* and *sel*.

mac

A predefined SunATM configuration file variable which is the 6-byte MAC address associated with the local host or interface.

MAC Address

The hardware address assigned to a system or interface board when it is manufactured. This address is guaranteed to be unique to the hardware.

macsel

A predefined SunATM configuration file variable which is the concatenation of `$mac:$sel`. See *mac* and *sel*.

myaddress

A predefined SunATM configuration file variable which is the concatenation of `$prefix:$mac:$sel`, resulting in the default address for the local interface. See *prefix*, *mac*, and *sel*.

Octet

A single byte. The individual bytes of an *ATM address* are often referred to as octets.

PVC

Permanent Virtual Channel. This is an ATM connection that is established by manual configuration on the two endpoints and switch, rather than with signalling.

Prefix

See *Switch Prefix*

prefix

A predefined SunATM configuration file variable which is the 13-byte prefix associated with the local switch.

Q.2931

The signalling protocol used in an ATM environment to establish connections between systems. Historically known as *Q93b*.

Q93b

See *Q.2931*

Q.SAAL

The specification for the Signalling ATM Adaptation Layer, which operates at the data link layer in an ATM protocol stack. This layer is often referred to as the Service Specific Connection Oriented Protocol, or *SSCOP*.

SSCOP

See *Q.SAAL*.

SVC

Switched Virtual Channel. This is an ATM connection that is established by the signalling protocol.

sel

A predefined SunATM configuration file variable which is the default 1-byte selector for the local interface.

Selector

The 1-byte portion of an *ATM address* that may be used for routing internal to an end system's implementation. Currently, for SunATM products, the selector should always be 00. The Selector and 6-byte *end system identifier* (*esi*) make up the local portion of an *ATM address*.

sunmacselN

A predefined SunATM configuration file variable which is the concatenation of one of a series of reserved MAC addresses and `$sel` to create a block of reserved ATM ARP server addresses. *N* should be a decimal number in the range 0 - 199. See *sel*.

Switch Prefix

The 13-byte portion of an *ATM address* that is assigned by the switch and used by the network to route messages to the proper destination.

VC

See *VCI*.

VCI

Virtual Channel Identifier. This is the number used to identify an ATM connection; a unique VCI is assigned to all connections, both *PVC* and *SVC*. The VCI is also often referred to simply as the VC. There are several reserved VCs used by the ATM protocols; signalling uses VC 5; *ILMI* uses VC 16, and *LAN Emulation* connections to the *LECS* use VC 17.

Index

Symbols

? wildcard, 5-9, 5-18

Numerics

96-Pin SBus Connector
pin descriptions, A-2

A

a configuration flag, 5-6, 5-7, 5-16, 5-17,
6-5

aarconfig file, 6-3, 6-4, 6-5
editing, 5-4
file flags, 5-5
flag options, 5-7

sample configurations, 5-11
using variables, 5-8

aarsetup program, 5-5, 6-3, 6-4, 6-5
error messages, C-13

anymac variable, 5-9, 5-18

anymacsel variable, 5-9, 5-18

API, E-1

atm_util functions, E-7
device driver
connecting, E-7
receiving data, E-7
sending data, E-7

DLPI encapsulated connections, E-9
message formats, E-4
q93b and device drivers, E-2
raw mode connections, E-9

Application Programmers Interface
See API

ARP address resolution tables, 6-4

ATM

address, 4-14, 4-19, 5-5, 5-15, 6-2, 6-5
aarconfig field, 5-5
laneconfig field, 5-15
registration, 6-2
resolution, 6-3
resolution tables, 6-4

ARP address resolution tables, 6-4

ARP server, 4-12, 4-14, 5-4, 5-5, 5-11,
5-13, 6-3, 6-4, 6-5
address, 4-16

caching, 6-5

M_PROTO mblock fields, E-4

q93b driver, E-2

qcc functions, E-5

router, 4-13, 5-4, 6-4

switch, 1-2, 1-6, 2-5, 2-8, 2-9, 6-2

switched virtual circuit, E-3

ATM Address field, 5-5, 5-8, 5-15

atmadmin program, 4-2

-
- Classical IP parameter group
 - menu, 4-12
 - common commands, 4-5
 - ILMI parameter group menu, 4-12
 - interface configuration menu, 4-7
 - LAN Emulation instance menu, 4-17
 - LAN Emulation per-interface
 - parameters menu, 4-18
 - main menu, 4-5
 - parameters, 4-8
 - physical layer parameter group
 - menu, 4-10
 - signalling parameter group
 - menu, 4-11
 - starting, 4-4
 - system parameter group menu, 4-6
 - using, 4-4 to 4-21
- atmconfig file
- editing, 5-2 to 5-3
 - example, 5-3
- atmf.mib MIB file, D-2
- atmf.mib.oid file, D-2
- atmf.mib.schema file, D-2
- atmreg program, 6-3
- atmsnmpd SNMP daemon, D-1, D-3
- atmstat command, C-3
- B**
- ba device, 1-7, 2-10, 5-3, 5-7
- broadcast and unknown address
 - server, 6-7
- broadcast messages, 6-5
- C**
- c configuration flag, 5-17
- caching, 6-5
- Call_ID message, E-4
- Call_Tag message, E-4
- checking
 - installation of a package, 3-5, 3-6
 - the network, 3-7
- CIP_Host field, 5-5
- Classical IP, 4-12, 5-4, 6-1, 6-3, 6-4
- configuring, 4-12, 5-4 to 5-14
 - no broadcast support, 4-12
 - sample configurations, 5-11
 - troubleshooting, C-3
- configuration variables
- in the aarconfig file, 5-8
 - in the laneconfig file, 5-18
- rules, 5-10
- setting, 5-8
- E**
- ejecting the CD-ROM, 3-4
- emulated LAN name, 4-20
- end system identifier field, 6-2
- Error Messages, C-10 to C-17
- Error_Code message, E-4
- examining network interfaces, 3-7
- extender plate, 2-7
- F**
- Flag field, 5-15
- framing interface
 - SDH, 4-3, 4-10
 - setting, 4-10
 - SONET, 4-3, 4-10
- H**
- hardware
- SunATM-155
 - installation, 2-6 to 2-9
 - requirements, 2-5
 - specifications, B-1 to B-2
 - supported platforms, 2-5
 - verifying the installation, 2-10
 - SunATM-622
 - installation, 1-3 to 1-6
 - requirements, 1-2
 - specifications, B-3 to B-4
 - supported platforms, 1-2
 - verifying the installation, 1-7
- Hostname field, 5-5

I

`ifconfig` command, 3-7, 6-1
Ifname message, E-4
ILMI service interface, 4-12, 6-2
ilmid daemon, 6-2, C-3
Interface field, 5-2, 5-5, 5-15
IP hostname, 4-14, 4-19
IP to ATM resolution, 6-3

K

kernel drivers,
 error messages, C-17

L

L configuration flag, 5-5, 5-7
l configuration flag, 5-5, 5-7, 5-16, 5-17
LAN Emulation, 5-14, 6-1, 6-5
 configuration server, 4-19, 6-6
 ATM address, 4-19
 configuring interface, 4-16 to 4-21,
 5-14
 connections, 6-9
 driver, 6-6, 6-7, 6-8
 instance number, 5-2
 IP address to an ATM
 connection, 4-20, 6-7
 multiple Emulated LANs, 4-20
 multiple emulated LANs, 4-3
 sample configurations, 5-19
 server, 4-19, 6-6
 ATM address, 4-20
 troubleshooting, C-6
lane# interface, 4-17, 5-3, 5-15
lane#:# interface, 4-20
lane.mib MIB file, D-2
lane.mib.oid file, D-2
lane.mib.schema file, D-2
LANE_Host field, 5-2
LANE_Instance field, 5-2
laneconfig file, 6-3
 editing, 5-14

 entry descriptions, 5-15
 flag descriptions, 5-16
 using variables, 5-18

lanesetup program, 5-14, 6-3
 error messages, C-13
localswitch_server variable, 4-15,
 5-9, 5-10

M

m configuration flag, 5-6, 5-7, 5-17
MAC address, 4-15, 5-9, 5-10, 5-15, 5-18,
 6-2, 6-6, 6-7, 6-8
MAC Address/Emulated LAN field, 5-15
mac variable, 4-15, 5-9, 5-18
macsel variable, 4-15, 5-9, 5-18
man pages, 3-3
myaddress variable, 4-15, 5-9, 5-18

N

n configuration flag, 5-16, 5-17
netstat command, 3-7, 6-1
nettest diagnostic, F-1
network prefix, 6-2

O

OpenBoot PROM, 2-2, F-2

P

permanent virtual circuit, 4-16
ping command, 3-7, C-5
 troubleshooting, C-5, C-8
pkgadd
 adding software packages, 3-2
 using, 3-5
pkgchk
 checking package installation, 3-5
pkginfo
 checking package installation, 3-6
 finding packages, 3-2
pkgrm

removing packages, 3-2, 3-6
prefix variable, 4-15, 5-9, 5-18

Q

Q.2931, 6-2, E-1
Q.93B, E-1

R

reconfiguration boot, 3-6
removing a package, 3-6
removing older software packages, 3-2
requirements
 SunATM-155
 hardware, 2-5
 software, 2-6
 supported systems, 2-5
 SunATM-622
 hardware, 1-2
 software, 1-2
 supported systems, 1-2
RJ45 connector, 2-9

S

s configuration flag, 5-5, 5-7, 5-17
S00sunatm boot script, 4-3
 error messages, C-10
SC receptacle, 1-6, 2-8
SDH, 4-3, 4-10
sel variable, 4-15, 5-9, 5-18
selector field, 6-2
selftest, F-2
show-devs command, 1-7, 2-10, F-3
SNMP
 installing the software, D-1
 setting agent status, 4-6
 setting up agent systems, D-3
 setting up management console, D-1
software
 checking the network, 3-7
 configuration, 4-4 to 4-21
 troubleshooting, C-1

 installation, 3-2 to 3-4
 requirements, 1-2, 2-6
 troubleshooting, 3-5
SONET, 4-3, 4-10
specifications, B-1 to B-4
spray command, 3-7
SunATM software
 CD-ROM
 ejecting, 3-4
 mounting, 3-2
 checking the network, 3-7
 configuration, 4-4 to 4-21
 troubleshooting, C-1
 variables, 4-15, 5-9
 installation, 3-2 to 3-4
 predefined variables, 5-18
 requirements, 2-6
 supported hardware versions, 4-1
SunATM-155 SBus adapters
 environmental specifications, B-2
 extender plate, 2-7
 hardware requirements, 2-5
 highlights, 2-1
 installation, 2-6 to 2-9
 performance specifications, B-1
 physical dimensions, B-2
 power specifications, B-2
 specifications, B-1 to B-2
 supported platforms, 2-5
SunATM-155/MMF SBus Adapter 2.1
 connecting to network, 2-8
 illustration, 2-3
 introduction, 2-1
 SC receptacle, 2-8
 wiring configuration, 2-8
SunATM-155/UTP5 SBus Adapter 2.1
 connecting to network, 2-9
 illustration, 2-4
 introduction, 2-1
 RJ45 connector, 2-9
 wiring configuration, 2-9
SunATM-622 SBus adapter
 connecting to network, 1-6
 environmental specifications, B-4

- hardware requirements, 1-2
- highlights, 1-1
- installation, 1-3 to 1-8
- introduction, 1-1
- performance specifications, B-3
- physical dimensions, B-3
- power specifications, B-3
- SC receptacle, 1-6
- specifications, B-3 to B-4
- supported platforms, 1-2
- wiring configuration, 1-6
- `sunmacselN` variable, 4-15, 5-9, 5-10
- SunVTS Validation and Test Suite, F-1
- SUNWatm
 - device drivers package, 3-3
- SUNWatma
 - interim API support package, 3-3
- SUNWatmu
 - man pages, 3-3
 - runtime support package, 3-3

T

- `t` configuration flag, 5-6, 5-7, 5-16, 5-17
- T568B pin designation, A-1
- test command, F-4
- troubleshooting, C-1 to C-9
- Type message, E-4

U

- UNI field, 5-2
- UNI specification, 5-2, 6-2
- UNI version, 4-11

V

- VCI field, 5-5, 5-15
- virtual circuit identifier, 4-16, 5-5

W

- wrist strap,
 - attaching, 1-3, 2-7

Reader Comment Card

Your comments and suggestions are important to us. Please let us know what you think about the *SunATM SBus Adapters Manual*, part number 802-6504-10.

1. Were the procedures well documented? Yes No

Please explain: _____

2. Were the tasks easy to follow? Yes No

Please explain: _____

3. Were the illustrations clear? Yes No

Please explain: _____

4. Was the information complete and easy to find? Yes No

Please explain: _____

5. Do you have additional comments about the SunATM SBus Adapters Manual?

You can send detailed comments via email to smcc-docs@sun.com, or send a fax to *SMCC Doc Feedback* at (415) 786-6443.

Your Name: _____

Title: _____

Company Name: _____

Address: _____

City: _____ State/Province: _____

Country: _____ Zip/Postal Code: _____

Email Address: _____

Telephone: _____

Part No.: 802-6504-10

Revision A, September 1996

Thank you.

**AIR MAIL
PAR AVION**

IBRS/CCRI No. 808



INTERNATIONAL BUSINESS REPLY MAIL/REPOSE PAYEE

PERMIT NO. 808 MOUNTAIN VIEW CA

POSTAGE WILL BE PAID BY ADDRESSEE

**INFORMATION PRODUCTS
M/S MPK14-108
SUN MICROSYSTEMS INC
2550 GARCIA AVE
MOUNTAIN VIEW CA 94043-9551
UNITED STATES OF AMERICA**

NE PAS AFFRANCHIR

**NO POSTAGE
NECESSARY
IF MAILED
TO THE
UNITED STATES**

